IMPROVING THE SECURITY OF CRITICAL INFRASTRUCTURE: METRICS, MEASUREMENTS, AND ANALYSIS

by

JEMAN PARK
B.S. Korea University, South Korea, 2016
M.S. University of Central Florida, US, 2019

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2020

Major Professor: David Mohaisen

ABSTRACT

In this work, we propose three important concepts needed in the process of improving the security of the critical infrastructure: metrics, measurement, and analysis. In the improvement of security technology, metrics are key to ensure the accuracy of the assessment and evaluation. Measurement is the core of the process of identifying the causality and effectiveness of various phenomena, and accurate measurement with the right assumptions is the cornerstone for accurate analysis. Lastly, analysis is a step of finding meaning from data collected through measurement. Different results can be derived for the same data according to the analysis method, and it can serve as a basis for the necessity of advanced research. In this dissertation, we look at whether these key concepts are well demonstrated in existing systems and research products. In the first thrust, we verified the validity of volume-based contribution evaluation metrics used in threat information sharing systems. In addition, we proposed qualitative evaluation as an alternative to supplement the shortcomings of the volume-based evaluation method. In the second thrust, we measured the effectiveness of the low-rate DDoS attacks in a realistic environment to highlight the importance of establishing realistic assumptions in measurements. Further, we theoretically analyzed the low-rate DDoS attacks and conducted additional experiments to validate them. In the last thrust, we conducted a large-scale measurement and analyzed the behaviors of open resolvers, to figure out the potential threats of them. We then went beyond just figuring out the number of open resolvers and explored the new implications the behavioral analysis could provide. We also experimentally proved the existence of forwarding resolvers and their behavior by precisely analyzing DNS packets of resolvers.

This dissertation is dedicated to my dear parents and sister who encouraged me all the time.

ACKNOWLEDGMENTS

This work would not have been possible without the support of many individuals over the years, to whom I am grateful. First, I would like to thank my advisor and doctoral committee chair, Dr. David Mohaisen, who found potential in me as a researcher, trained me to develop my skills and knowledge in the domain, offered the flexibility for me to pursue my interests, and helped me publish quality publications. I would like also to thank the members of my dissertation committee for their help and support: Dr. Wei Zhang, Dr. Sung Choi, and Dr. Yanjie Fu. I want to thank my teammates in the Security Analytics Lab (SEAL) for their great cooperation and encouragement.

I am also grateful to Dr. DaeHun Nyang for his valuable advice on my research and future plans. Last but not least, I would like to express my deep gratitude to Dr. Hyogon Kim, my undergraduate advisor, for guiding me when I wanted to pursue my doctoral study in the US.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

In the world of Internet, there is a constant battle between attackers and defenders. On one hand, attackers have developed new types of malicious software to steal secret information from users or infected devices to launch DDoS attacks on critical facilities. On the other hands, defenders have made a huge efforts to gather information on emerging threats and to devise new defense techniques. Attack and defense techniques on computers or networks have been continuously studied by numerous researchers over the world, which leads to great advances in both knowledge and practical application.

In general, the progress of computer/network security technology can be summarized as follows: 1) First, a new type of attack is proposed or detected by researchers or security vendors, and the vulnerabilities that enable the attack are discovered. 2) Measurements are made on how many systems expose these vulnerabilities. 3) Various candidates are suggested for how to deal with the vulnerabilities and defend against a new type of attacks. 4) Measurements of the effectiveness of the new defenses are made, and threat information sharing system is established to monitor the prevalence of threats. For example, when a new type of malware launching a Distributed Denial of Service (DDoS) attack emerges, the abnormal activity of the infected hosts is detected. Then, the malware sample is collected by researchers or security vendors and analyzed through various approaches (e.g., reverse engineering, sandbox testing, etc.). Based on the analysis, associated vulnerabilities are discovered and supplemented by security patches, while the characteristics of DDoS attacks which the malware generated are analyzed at the targeted server. At the same time, the malware sample is collected and shared by an information threat sharing system such as Virus-Total to alert new threats and track their evolution.

In the process of improvement in computer security, there are three key concepts to successfully improve the security of critical infrastructure: metrics, measurement, and analysis. First, **metrics** are a set of tools used to evaluate quantitative or qualitative aspects of security. Metrics are a key

concept in deciding what and how to evaluate an artifact in measurements and analysis. For example, the attack duration or peak rate of attack streams can be a metric to evaluate the effectiveness of DDoS attacks. In other words, if metrics are inappropriately set, measurements and analysis based on them are also inaccurate. Secondly, **measurements** are activities to characterize an artifact or a phenomenon. In computer security, it primarily means the act of identifying the magnitude of potential threats, the effectiveness of attacks or defenses. However, measurements are meaningful only if the correct parameters, assumptions, and circumstances (settings) are supported; otherwise, the results of the measurements are likely to be distorted. Finally, the **analysis** is concerned with extracting meaningful information from the data resulting from the measurements. This is the process of identifying the relationship between the variables set in the measurement by correlating the results. The analysis gives us insight into what can weaken the effectiveness of an attack, or what can strengthen defense against such an attack, leading to sustainable efforts around networks and systems security.

1.1 Motivation

In the above, we elaborated on three key concepts in the advance of computer security. It is undeniable that if these three pillars are not well established, it is difficult to respond appropriately to emerging threats. Our work took the first step in this regard by examining various foundational questions, including: 1) Is the security of critical infrastructures being well improved by information sharing and new defenses? 2) Are the three components above demonstrated in current system security research practice? 3) What can we do to bridge the gap between the suboptimal reality and what is anticipated?

In this study, we question existing research and systems. To be specific, based on three key concepts proposed, we evaluate the security of critical infrastructure. In the first thrust (Chapter 2), we verified the validity of volume-based **contribution evaluation metrics** used in threat infor-

mation sharing systems. In addition, we proposed qualitative evaluation as a supplement to the shortcomings of the volume-based evaluation method. In the second thrust (Chapter 3), we measured the effectiveness of the low-rate DDoS attacks in a realistic environment to highlight the importance of establishing **realistic assumptions in measurements**. Further, we theoretically analyzed the low-rate DDoS attacks and conducted additional experiments to validate them. In the last thrust (Chapter 4), we conducted a large-scale measurement and analyzed the behaviors of open resolvers, to figure out the potential threats of them. We then went beyond just figuring out the number of open resolvers and explored the new implications the **behavioral analysis** could provide. We also experimentally proved the existence of forwarding resolvers and their behavior by precisely analyzing DNS packets of resolvers.

1.2 Statement of Research

Quality of Indicator (Chapter 2). Our goal in the first thrust in my dissertation is to examine the appropriateness of evaluation metrics in the current threat information sharing systems. Given in the insufficiency of volume-based contribution metrics, we suggest a way that each contribution can be evaluated in terms of the quality rather than the traditional method of quantitative evaluation. For that, we propose four new qualitative evaluation metrics; accuracy, relevance, utility, and uniqueness, and use them to demonstrate how different results from the existing quantitative methods are achieved. By re-evaluating the quality of contribution from each anti-virus vendor, in a malware artifacts and labels sharing system of information sharing, we highlight the effectiveness of qualitative metrics and discuss how such metrics alleviate the impact of free-riding.

Measurement of the Low-Rate DDoS Attack (Chapter 3). The second thrust of the dissertation is to validate the assumptions in the measurement space. More specifically, we evaluate and measure the effectiveness of low-rate DDoS attacks. In the literature [56] which examines the threat of the Very Short Intermittent DDoS attack (VSI-DDoS attack, a kind of low-rate DDoS attack),

researchers measured the effectiveness of the attack assuming tight botnet synchronization. They assumed that bots can concentrate attack streams in a very short time (e.g., 10ms), which is not a trivial assumption in current distributed systems. Through their evaluation, researchers highlight the potential threat of the VSI-DDoS attack as a practical attack.

In this thrust, we question the assumption used in those measurements, explore measuring the degree of synchronization of botnets in realistic environments, and apply the results in context to figure out the actual effectiveness of the VSI-DDoS attacks. By evaluating the attacks with multiple synchronization scenarios, we highlight that they can be overestimated by a small flaw in assumptions. As an extended work, we conduct a thorough theoretical analysis of the relationship between the level of synchronization and effectiveness.

Threats of Open Resolvers (Chapter 4). As a final thrust of the dissertation, we conduct a large-scale measurement and an in-depth analysis of Internet threats. We focus on the Domain Name System (DNS), which is one of the core critical infrastructures in today's Internet. Our key pursuit is to examine how certain components in the DNS ecosystem affect the security of the Internet as a whole; namely the open resolvers. Although existing studies have already pointed out the security risks of open resolvers, there are still a large number of them in the wild, with changing behaviors. Our research begins with an attempt to find out the current status of these open resolvers. Unlike previous studies that were limited to understanding the spatial distribution of open resolvers, we analyze the behavior of these resolvers in detail using two datasets collected through active scans and probing using a pre-configured network setup. This setup allowed us to investigate not only the fact that there are millions of open resolvers, but also the behavior of open resolvers confirmed again a ground-truth: they are shown to not conform to standards, DNS manipulation attacks are quite prevalent. Furthermore, we further study whether DNS forwarders are used and their behaviors.

1.3 Dissertation Organization

This dissertation includes contents and material from three published papers by the author. Chapter 2 incorporates the material from Reference [44], which presents qualitative metrics for evaluating contribution in the threat information system as a supplement to an existing quantitative evaluation. Chapter 3 is based on References [47] and [46], which measures and theoretically models the effectiveness of pulsing DDoS attacks under the realistic synchronization assumption. Finally, Chapter 4 which is based on Reference [45] demonstrates the abnormal behaviors of open resolvers and forwarders through a large-scale measurement and in-depth analysis. This introductory Chapter and Chapter 5 also use some material from the above papers.

CHAPTER 2: ASSESSING PARTICIPATION IN THREAT INFORMATION SHARING[1]

In this chapter, we investigate the appropriateness of the **metrics** used in the most of the current threat information systems. To demonstrate the limitations of metrics that only consider a quantitative aspect, we introduce the notion of Quality of Indicator (QoI) to assess the level of contribution by participants in threat intelligence sharing. We exemplify QoI by metrics of the correctness, relevance, utility, and uniqueness of indicators. We build a system that extrapolates the metrics using a machine learning process over a reference set of indicators. We compare these results against a model that only considers the volume of information as a metric for contribution, and unveil various observations, including the ability to spot low-quality contributions that are synonymous to free-riding.

2.1 Motivation

In threat intelligence sharing, participants exchange patterns of threats, in the form of threat indicators or signals, with each other [38, 8, 60, 58, 20, 30]. Participants are defined over a community of trust and ideally collaborate towards a common mission: to understand and respond to emerging threats [65]. For such intelligence sharing to happen, standards for representation, exchange, and consumption of indicators are used [7, 23, 32, 49, 69, 22, 15]. Communities of trust are established, and systems and initiatives for sharing are built. However, participants need to contribute information in those systems to be consumed by other community members. Given the coinciding benefits and risks of threat information sharing, some community members have adopted an elusive behavior of "free-riding" [18] so that they can achieve utility of the sharing paradigms without

[1]This content was reproduced from the following article: J. Park, H. Alasmary, O. Al-Ibrahim, C. Kamhoua, K. Kwiat, L. Njilla, and D. Mohaisen, "QoI: Assessing Participation in Threat Information Sharing," *In Proceedings of the 43rd International Conference on Acoustics, Speech and Signal Processing*, IEEE ICASSP 2018, Calgary, AB, Canada, 2018. The copyright form for this article is included in the appendix.

contributing much to the community.

Understanding the effectiveness of sharing has been viewed from the point of view of whether participants contribute or not, and using the volume of contributed indicators. Therefore, a community member who does not contribute a volume of data is considered a free-riding community member [63]. The state-of-the-art on the problem did not include other metrics beyond simple measures of volume-based contribution, particularly metrics that evaluate qualitatively the indicator, which means the threat information contributed by participants.

Contributions. In this work, we propose a new approach to qualitatively assess the contribution to threat information sharing. By estimating the quality of contribution and comparing it to volume-based evaluation, we highlight the difference between the two methods. Our main contributions are as follows:

1. We identify the need for QoI to capture contributions by community members in an information sharing paradigm.
2. We develop and formulate four metrics which are correctness, relevance, utility, and uniqueness, to capture the notion of quality implemented in a system.
3. We experimentally demonstrate those measures and metrics and show how they differ in identifying a contributor's behavior from the simple volume-based measure of contribution.

2.2 System Design

In this section, we describe the architecture and overall procedures in the proposed QoI system. We exemplify a malware information sharing system to show how each QoI is computed and what metrics can be used for qualitative evaluation of contribution.

2.2.1 System Architecture and Design

Strawman Design. Before nodes can accept indicators, they first evaluate their quality by asking a special node, an assessor, to perform rating of the indicators. While effective, this strawman design has multiple issues. Most importantly, each community member has to fully trust the assessor and the validity of its scoring of indicators. Second, the approach is prone to disruption by the failure of the assessor. Such issues could be, however, mitigated by introducing multiple assessors, where the final score of quality is obtained using a consensus over multiple scores, one per assessor.

Distributed Design. On the other end of the spectrum of the strawman design is to let every node in the system to be an assessor. In doing that, we implicitly assume the availability of reference data to each node in the system, which is implausible. For example, based on a prior study, no single community member (antivirus scanner in the case of malware detection and labeling) has a 100% coverage or accuracy [41]: namely, it takes between 6 and 18 community members to provide close to perfect coverage of detection and correctness of labeling, respectively, for a malware family like Zeus [39]. This, in turn, calls for a more "intelligent" process for the assessment of QoI. Our system assumes the reference dataset has sufficient information about every indicator presented by the different community members. A high-level description of the system implementation, incorporating both learning and assessment, is shown in Figure 2.1. Note that this system is ideally executed by each community member.

2.2.2 System Setup

We use malware indicators sharing as a working example. Among the steps described below, each process from S1 to S3 is matched with box 1 through 3 of Figure 2.1.

◇ S0: **Defining Metrics.** Quality metrics are used to ensure that community members who participate in information sharing provide threat indicators that are valuable to other members, while scoring procedures are methods that specify how these metrics are used to generate a quality score.

The main purpose of this work is defining those metrics.

◇ S1: **Defining Labels.** Annotations capture the type of threat, the level (of severity, timeliness, etc.) or quality type of an indicator. Utilizing these annotations, a weight value is assigned to each quality label, and a scoring method is used to convert the quality labels to a numeric aggregate score for the indicator and ultimately to the contributor.

◇ S2: **Building a Reference.** The reference dataset is used to evaluate QoI for a sample of indicators submitted by a sample provider. To build the initial reference dataset, data that is collected through security operations (e.g., monitoring, profiling, analyses, etc.) is vetted for their validity and applicability to the domain, perhaps using often expensive by necessary manual vetting [41, 40]. This reference dataset is used for the purpose of initializing the system.

◇ S3: **Extrapolating.** Extrapolation allows each assessor to predict the label of an indicator using its feature set and classifier model. The classifier is trained using a supervised learning process extracted from the reference dataset.

2.2.3 QoI Assessment Process

An illustration of the QoI assessment is depicted in Figure 2.1. The QoI assessment is achieved through a supervised learning process over the reference dataset, rather than the direct matching of explicit labels of indicators in the dataset.

We assume a reference labeled (training) dataset that contains a comprehensive library of artifacts, such as malware samples, incident reports, and logs, and that has been collected through operational intelligence gathering procedures. To predict a label correctly, the build of our trained model encompasses multiple components, namely a feature selection procedure, a machine learning algorithm selection procedure (e.g., SVM, logistic regression, random forest, etc.) and the corresponding parameters (e.g., procedure for regularization and linearization in case of SVM and

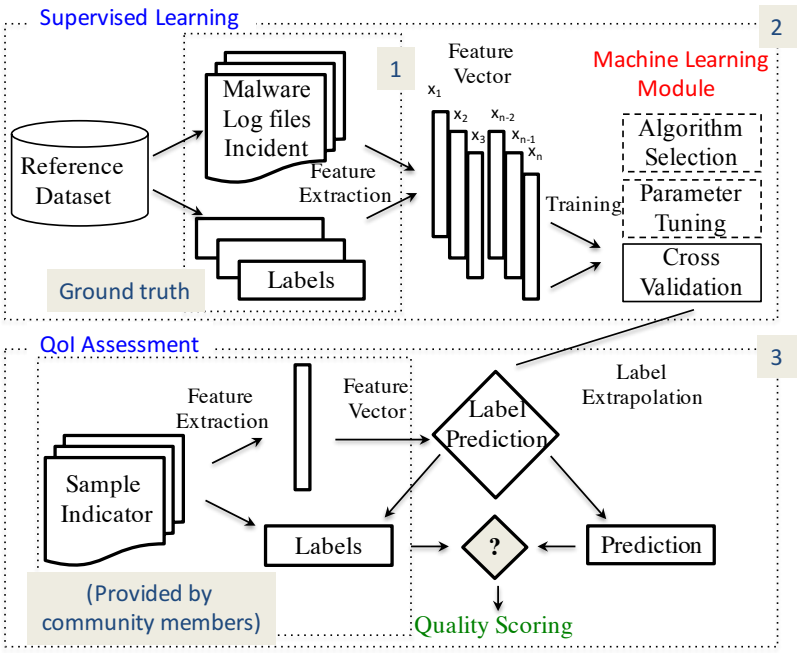


Figure 2.1: The QoI assessment process, incorporating a reference model and a learning component for extrapolation.

LR, respectively), and cross-validation procedures (e.g., fold size, validation, etc.).

The assessor uses the previously built model as a predictor and assigns a label to the indicator. The assessor decides the quality of the indicator by taking both the predicted and the self-provided labels into account (details are in the following section). The quality scoring procedure then aggregates the individual scores of the various indicators provided by each community member to assess their actual contribution.

2.2.4 Overall System Procedures

As outlined in sections 2.2.2 and 2.2.3, the QoI assessment process consists of multiple procedures for assessment initialization, learning, and extrapolation. In the following, we elaborate on some of the technical details of those procedures.

Reference Dataset and Learning. After identifying metrics for defining quality, we demonstrate the use of QoI for the assessment of the contribution level of participants. In order to initialize our system, a reference dataset is used to build a prediction model through supervised techniques of learning. Specifically, the collection of a reference dataset involves submission of sample artifacts from multiple sources, such as other community members, vendors, or own research [42].

We use various features for learning a model and their label prediction. For that, we use both static and dynamic analysis. For static analysis, we use artifacts such as file name, size, hashes, magic literals, compression artifacts, date, source, author, file type and portal executable (PE) header, among others. For dynamic analysis, we use counts that capture interactions between malware and the file system, user memory, registry, and network artifacts and features, as detailed in [42].

Modeling and Learning. In the following, we elaborate on a particular learning technique as a tool of QoI assessment. To build our classifier, we obtain r samples for training from the reference dataset consisting of r_i training samples per class, with d features per sample. For each training sample y, we observe a label $\ell \in \Lambda$ and a sample vector $\vec{y}$, where $\vec{y}$ is a vector of length m. We refer to the classes labels by their indices $i = 1, 2, \ldots, \lambda$. We assume that samples labeled by i are distributed as $\mathcal{N}(\mu_i, \Sigma)$, the multivariate normal distribution with mean vector μ_i and standard deviation matrix Σ. We denote by $\mathscr{L}(x, \mu_i, \Sigma)$ the corresponding probability density function and by π_i the prior probability that an unknown sample comes from class labeled by i. Bayes' Theorem states that the probability an observed sample x comes from class i is proportional to the product of the class density and prior probability; namely $P(Z = i|X = x) \propto \mathscr{L}(x, \mu_i, \Sigma) \times \pi_i$, where $P(Z = i|X = x)$ is the posterior probability that sample x comes from class i. The classifier assigns the sample to the class with the largest posterior probability to minimize the misclassification error. This can be written as a rule: $\hat{z}(x) = \arg\min_i \{(x-\mu_i)^T \Sigma^{-1}(x-\mu_i) - 2\log(\pi_i)\}$ A sample is assigned nearest class with the distance being $||x-\mu_i||^2_\Sigma - 2\log(\pi_i)$, where $||x-\mu||^2 = (x-\mu)^T\Sigma^{-1}(x-\mu)$; square the Mahalanobis distance between x and μ.

noend 1 Relevance of X_i (R)

1: Define weight values: $w_{r_1}, w_{r_2}, w_{r_3}, \ldots w_{r_{|\Lambda|}} \in \mathbb{R}$
2: Define $w_R(.)$ to assign weights to labels: $w_R(l_i) = w_{r_i}$
3: Compute the relevance of X_i as the average of the weighted sum: $R(X_i) = (\sum_{(\vec{x}_{ij}, l_{ij}) \in X_i} w_R(l_{ij}))/(\sum_{k=1}^{|\Lambda|} w_{r_k})$

Misclassification rate. A misclassification occurs when an indicator is assigned to an incorrect label, which done with $P(\varepsilon)$, where: $P(\varepsilon) = \sum_{j=1}^{\lambda}[P(\hat{Z} \neq j | Z = j) \times \pi_j]$.

Labeling and Quality Scoring. Denote by n the number of community members. Each (user) u_i provides a set of samples $X_i = \{(\vec{x}_{i1}, l_{i1}), (\vec{x}_{i2}, l_{i2}), (\vec{x}_{i3}, l_{i3}), \ldots, (\vec{x}_{ik}, l_{ik})\}$ with feature vector $\vec{x}_{ij}$ and sample label $l_{ij} \in \Lambda$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, k$.

Correctness. The reference dataset is used as the benchmark for determining the correct label for an arbitrary sample. Each sample consists of a feature vector and an associated label. We first build a classifier by utilizing the reference dataset Y as the training set and forming a prediction on the label of $\vec{x}$ to obtain l'. Then, the assigned label of $\vec{x}$ is compared against the predicted label l' and a positive score is given if labels match. The correctness is computed as the average of the sum of scores for all samples in X_i.

Relevance. The algorithm for computing the relevance is shown in Procedure 1. In this procedure, the weight values $w_{r_1}, w_{r_2}, \ldots w_{r_{|\Lambda|}}$ are chosen based on the context of use, and a mapping function $w_R(.)$ is defined to assign weights labels such that higher weight values are assigned to labels of greater interest to the community members. For each sample x, the corresponding label is evaluated using the mapping function $w_R(.)$ to obtain the weight value as the sample score. The relevance score of X_i, denoted by $R(X_i)$, is calculated as the average of the weighted sum of the scores.

Utility. Proc. 2 defines the utility. In this procedure we note that the utility of an indicator is determined by the sum of the utility weights of the samples. The weights $w_{t_1}, w_{t_2} \ldots w_{t_d}$ and weight function $w_T(.)$ are defined by the application.

Uniqueness. Procedure 3 outlines the steps used for calculating the uniqueness of a set of indica-

noend 2 Utility of X_i (U)

1: Define $t_1, t_2, \ldots, t_d \in \mathbb{R}$.
2: Define weight values $w_{t_1}, w_{t_2} \ldots w_{t_d}$, where each weight value corresponds to a utility type.
3: Define a weight function $w_T(.)$ s.t. $\ell \in \Lambda$ maps to a utility weight, i.e. $w_T(\ell) = w_{t_m}$ where $m = \{1, 2, \ldots, d\}$.
4: **for** $x \in X_i$ **do**
5: Compute a weight of $x = (\vec{x}, l')$, using $w_T(l') = w_{t_{l'}}$
6: **end for**
7: Compute the utility score of X_i as the average of the sum of the sample weights: $U(X_i) = \frac{1}{k} \sum_{j=1}^{k} w_{t_j}$, where t_j is the corresponding label type of sample $x_{ij} \in X_i$

noend 3 Uniqueness of X_i (N)

1: Consider the set Z which is initially empty, i.e. $Z = \phi$
2: Build the set Z by considering unique samples from the sets $X_1, X_2, ...X_n$
3: **for** $i = 1, 2, \ldots n$ **do**
4: **for** $j = 1, 2, \ldots k$ **do**
5: **if** $x_{ij} \notin Z$ **then** add x_{ij} to Z.
6: **end if**
7: **end for**
8: **end for**
9: Compute $s_n(x_{ij}) = 1$ if $x_{ij} \in Z \backslash \{X_i\}$ and 0 otherwise.
10: Compute $N(X_i) = \frac{1}{k} \sum_{j=1}^{k} s_n(x_{ij})$

tors. In this procedure, we assume that samples can be uniquely identified (e.g. using hashes). In set notation, we can say that an element $x_{ij} \in X_i$ is unique if it is not an element of other sample sets, i.e. $x_{ij} \notin \bigcup X \setminus \{X_i\}$.

Quality of Indicator (QoI). QoI is a comprehensive measure and is calculated as the average of the weighted sum of the four components: correctness (C), relevance (R), utility (U) and uniqueness (N), as shown in procedure 4. The weights assigned for metrics are application-specific.

2.3 Evaluation

In this section, we introduce the dataset used in the evaluation and highlight the importance of a qualitative evaluation by comparing the results with ones from a quantitative metrics.

noend 4 QoI of X_i (QoI)

1: Define normalized weights, w_C, w_R, w_U, and w_N.
2: Calculate QoI as the weighted sum of the components: $QoI(X_i) = w_C C(X_i) + w_R R(X_i) + w_U U(X_i) + w_N N(X_i)$

2.3.1 Dataset

We use 11 malware families for our evaluation; five DDoS (Avzhan; 3458 samples, Darkness; 1878 samples, Ddoser; 502 samples, jkddos; 333 samples, N0ise; 431 samples), four targeted (ShadyRAT; 1287 samples, DNSCalc; 403 samples, Lurid; 399 samples, Getkys; 953 samples), and two mass market families (ZeroAccess; 568 samples, Zeus; 1975 samples). The dataset is collected between mid-2011 to mid-2013 [41]. Scans for labeling are done through VirusTotal around May 2013, static features are obtained from VirusTotal, and dynamic features are extracted using AMAL [42].

2.3.2 Results

We note that while there are more samples gathered for DDoS-type malware in comparison with others, the threat-intelligence community often gives more weight to identify malware or incidents that are less observable, which presents a level of sophistication. Thus, for our evaluation, we consider trojan and targeted malware more relevant than DDoS, from the point of view of community members consuming the shared information. The results are shown in Figure 2.2, and the following are observations on three of QoI metrics (since evaluating uniqueness is trivial).

◇ ***Correctness.*** As shown in Figure 2.2, the majority of other vendors have a gap in the score between the volume-based and the correctness-based contribution measures, where the correctness-based measure is significantly lower. Reasons may include labeling samples under unknown names, mislabeling due to similarities between families, or assigning generic labels like “trojan”, “virus”, and “unclassified”. Examining the correctness of AV indicators also leads to a more subtle discussion about their utility. Looking closer into the label generated by some vendors, we found

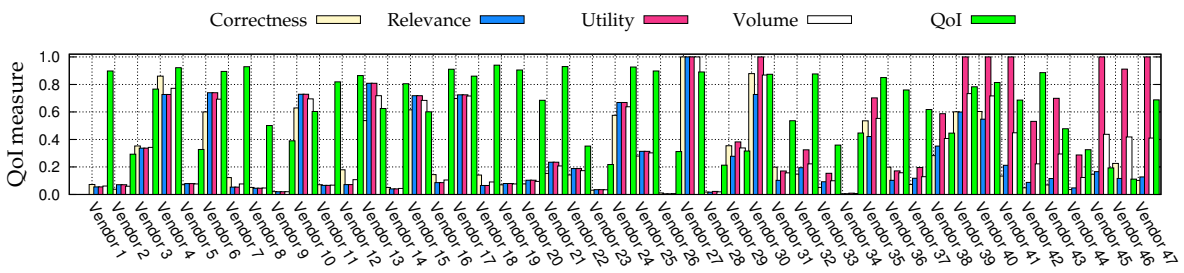


Figure 2.2: Comparison between various quality-based metrics for AV indicator assessment

out that some labels are too generic and only describe the behavior rather than the name of a known malware family type, e.g., Trojan.Win32.ServStart vs. Avzhan.

◇ ***Relevance.*** We give more weight to targeted malware and Trojans over DDoS samples: $w_{targeted} = 5$, $w_{troj} = 3$, $w_{ddos} = 1$, and "0" otherwise. Thus, we observe two behaviors. First, certain contributors with high volume-based scores have a very low score (close to "0") of relevance. In particular, with the two relevant and one less relevant family types identified, such community members have more unidentified malware samples (e.g., vendor 7, vendor 21, vendor 59, etc.). On the other hand, other contributors with small volume-based contribution, have a higher relevance score, due to very relevant family identified in their shared indicator (e.g., vendor 10, vendor 16, vendor 27, etc.).

◇ ***Utility.*** To evaluate the utility, we give weights for three classes of malware labels: complete labels (w_c) are based on industrially popular name, generic labels (w_g) are based on placeholders commonly used for labeling the family such as " generic", " worm", " trojan", " start" and " run", and incomplete labels, (w_i), including " suspicious", " malware", and " unclassified", which do not hold any meaning of a class. Similar to the strategy with relevance, we assign weights of $w_c = 5$, $w_g = 2$, and $w_i = 1$.

◇ ***Aggregated*** QoI ***score.*** We aggregate a single QoI score for each vendor based on the weighted sum of the various QoI metrics. As can be seen, vendors, such as 39 and 46, with low QoI scores are rated with higher scores in their volume-based contribution, potentially alluding to free-riding. On the other hand, vendor 1, vendor 8, and vendor 33, among others, which tended to have lower volume-based scores, tend to have higher QoI, indicating their quality of contribution.

2.4 Summary

In this work, we have the first look at the notion of the quality of indicators (QoI). As empirically analyzed, identifying the levels of contribution cannot be simply expressed in the volume-based measure of contribution. By verifying our metrics on a real-world data of antivirus scans we unveil that contribution measured by volume is not always consistent with those quality measures, and that QoI as a notion is capable of capturing forms of contribution analogous to free-riding.

CHAPTER 3: ASSESSING THE EFFECTIVENESS OF PULSING DDOS ATTACKS UNDER REALISTIC NETWORK SYNCHRONIZATION[1]

In this chapter, we examine another key concept, **measurements**. Specifically, to understand the importance of establishing realistic assumptions in measurements, we revisit the key premise of the VSI-DDoS, or the pulsing types of low-rate DDoS attack, in the wild. Specifically, we are interested in assessing the impact of the attack when evaluated under more realistic assumptions of bots' synchronization. The key motivation for this assessment is that in distributed systems in general, and in botnets (as a special case of distributed systems) in particular, time-synchronization is a non-trivial task. In other words, when multiple machines gather together to form a botnet, it can be practically difficult for any traffic, such as HTTP requests, originating from all bots to arrive at a given server in a short period of time (a few milliseconds).

3.1 Motivation

The Distributed Denial-of-Service (DDoS) attack, a representative type of attacks targeting the availability of the system, has re-emerged recently as one of the most challenging threats [71, 70]. Until recently, DDoS attacks have been performed by sending a large number of packets to target servers to deplete their resources, bandwidth or memory. An enormous number of packets generated using a set of infected machines in a botnet saturates the available resources of the targeted server [61, 54], making it unavailable to legitimate users and causing a denial of service [24, 25].

Recently, new types of low-rate DDoS attacks, which deviated from the high volume-based DDoS

[1]This content was reproduced from the following articles: J.Park, D. Nyang, and D. Mohaisen "Timing is Almost Everything: Realistic Evaluation of the Very Short Intermittent DDoS Attacks," *In Proceedings of the 16th International Conference on Privacy, Security and Trust*, PST 2018, Belfast, Northern Ireland, United Kingdom, 2018. and J. Park, M. Mohaisen, D. Nyang, and D. Mohaisen, "Assessing the Effectiveness of Pulsing Denial of Service Attacks under Realistic Network Synchronization Assumptions," *In Elsevier Computer Networks*, Vol. 173, 2020. The copyright form for these articles is included in the appendix.

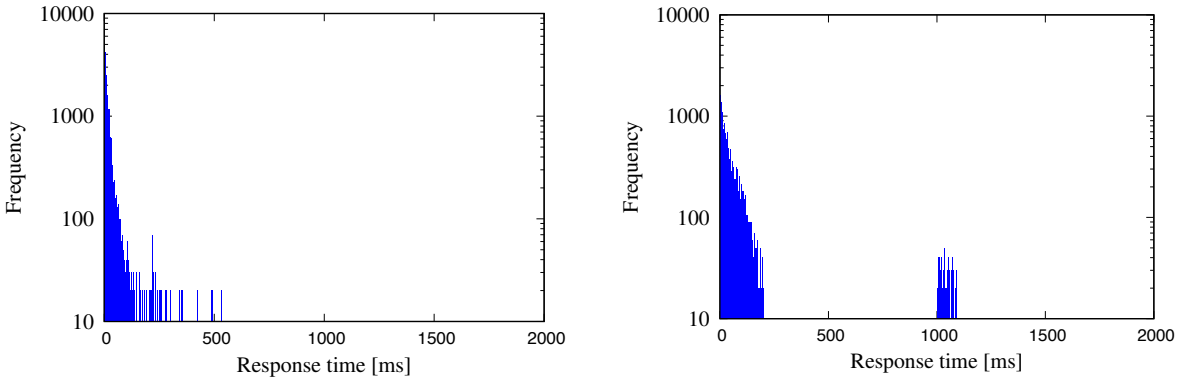
attacks, have emerged as a new threat [31, 51, 57]. The low-rate DDoS attacks significantly differ from traditional attack technique. As the name suggests, these attacks are made with low-rate or low volume. In general, the prime goal of low-rate attack is to degrade the Quality of Service (QoS) that legitimate users experience by exploiting the congestion control mechanism of TCP. At the same time, the low-rate DDoS attack is characterized by being very stealthy due to the small amount of attack volume. There have been no noticeable reports of damages caused by low-rate DDoS attacks, but there is still growing concern about this new type of attack and their potential implications in violating service level agreements (SLAs).

Perhaps the biggest advantage of low-rate DDoS attack is stealthiness. Since this type of DDoS attack does not aim at complete depletion of the resource, it only needs a relatively small amount of traffic. As such, the key factor for a successful attack is the concentration of the attack traffic. If the adversary is capable of concentrating attack packets in a shorter time, the attacker may need a smaller number bots and less traffic from each bot, which improves the effectiveness and stealthiness of the attack. Thus, in a botnet that consists of multiple bots, the synchronization of attack streams can be considered a key factor for the successful low-rate DDoS attack. In other words, if there is an excessive assumption on it, the anticipated effectiveness of the attack can be overestimated.

Contributions. In this work, we are interested in assessing the impact of the attack when evaluated under more realistic assumptions of bots' synchronization. By measuring and comparing the results from ideal and realistic environments, we highlight the importance of accurate assumptions in understanding the threat. Our main contributions are as follows:

1. We measure the Very Short Intermittent DDoS (VSI-DDoS) attack as an example of low-rate DDoS attacks in a loosely-synchronized environment of the botnet to understand its real, potential, and possible difficulty in deploying it.

2. Through the preliminary experiment, we fix the six degrees of synchronization of the botnet



(a) Response time w/o the VSI-DDoS attack (b) Response time under the VSI-DDoS attack

Figure 3.1: The distribution of the HTTP response time experienced by the legitimate users. Under the VSI-DDoS attack, some users may experience the very long response time (over than 1 second) for their requests due to the TCP retransmission (right), while the users can get the most responses within 100ms in general (left).

and apply it to quantitatively evaluate the actual effect of the low-rate DDoS attack.

3. We demonstrate that even a moderate imperfections in bots time synchronization would degrade the impact of the VSI-DDoS attack, and pronounce it ineffective.

4. We theoretically analyze the correlation between the level of synchronization and the difficulty to achieve the adversary's goal.

5. We further measure the required number of attack packets with different goals (95th and 99th percentile response time) to demonstrate the attacker's limitations in a realistic environment.

3.2 Background

3.2.1 The Objective: Latency as a Target

In web applications, the QoS is considered an important notion that measures the overall performance of the system through various easily to interpret measures [33]. Users' experience when

using Internet services, which is affected by QoS measures, has a significant impact on their behavior. For example, service providers' revenue can be severely affected by the QoS that users experience when using web services [27]. Among the various aspects that are used to compute QoS, the latency is one of the most important factors. For users surfing multiple pages in the web service, if the latency is several seconds or more, it can lead to a poor user experience. Users are unlikely to want to use such services and will eventually leave, resulting in a decrease of audience and sales—in e-commerce, for example. Thus, many web service providers, including Google and Amazon, are making a lot of efforts to reduce tail latency [2] to a level that does not inconvenience users. This nature of customer behavior opens up the possibility of a new type of attacks, the low-rate DDoS attacks. In those attacks, the adversary causes a damage to the web service provider by simply causing degradation of QoS. This kind of attack differs from the traditional DDoS attack, which completely paralyzes a link or a server by sending a huge amount of packets to the victim.

The major factor making low-rate DDoS attacks possible is the congestion control mechanism in TCP. Compared to UDP, TCP is considered reliable because it provides retransmission function for failed transmissions. When the TCP packet is dropped during transmission, the sender will retransmit after waiting for a response during Retransmission Timeout (RTO). When it comes to RTO, RFC6298 [67] introduces how to calculate RTO and indicates that if the computed RTO is less than 1 second, it should be rounded up to 1 second. It means that if the drop of a legitimate packet occurs, the latency that the sender experiences should be longer than 1 second. Low-rate DDoS attacks exploit such characteristics of TCP-based communication.

In general, the objective of the low-rate DDoS attacks is high latency itself. The attack causes some TCP-based packets to be dropped at the link or server, causing them to be retransmitted. Under this circumstance, the sender waits for the RTO (or even longer) to detect the packet loss and retransmit. This attack differs significantly from the traditional DDoS attacks, which almost

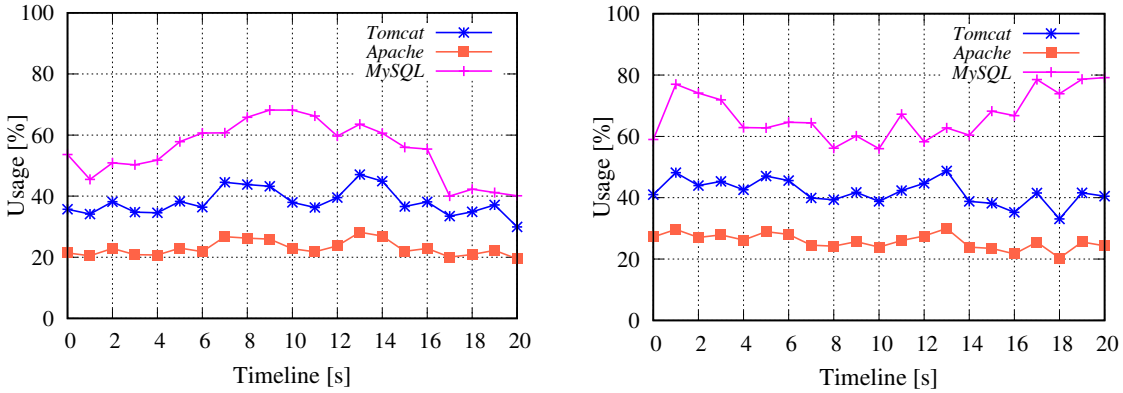
[2]The tail latency is defined as the latency of a server's 99th percentile response, which means the delay that users experience in the worst case.

completely block user access by depleting the resources of the target server, whereas in this attack users may only suffer bad experience that makes the service almost unusable due to the high latency. While traditional attacks need a huge amount of attacking traffic to deplete, low-rate DDoS attacks need a relatively small amount of packets to make some of the legitimate users' packets be dropped.

3.2.2 Example: VSI-DDoS Attack

Operation. The VSI-DDoS attack is a recently introduced low-rate DDoS attack which belongs to the type of pulsing attacks. In particular, it aims at the application layer of the target server and exploits very short bottlenecks (VSBs) in the n-tier web application system. By causing VSBs on web application servers, the VSI-DDoS attack degrades the legitimate users' QoS [56]. As we described earlier, the VSI-DDoS attack causes the transient saturation of resources and delays the response to legitimate users' requests by exploiting TCP RTO, while the traditional DDoS attacks exhaust server resources for a long period of time. For example, when a number of HTTP requests suddenly concentrate within a few milliseconds and exceed the server's queue limit, the legitimate user's request is dropped, resulting in a very long response time (VLRT), as the TCP retransmission occurs. The occurrence of TCP retransmission aggravates the user experience, since the user cannot use the service until the response of the retransmitted request arrives after the TCP RTO for the dropped packet. Fig. 3.1 shows how the VSI-DDoS attack can adversely affect the user's experience. In general, most responses for HTTP requests are quickly returned to the users within 100ms (as in Fig. 3.1a). However, under the VSI-DDoS attack, some of the responses are significantly delayed more than one second due to the attack (as in Fig. 3.1b). As such, the VSI-DDoS attack can be a major threat to web application services, affecting their QoS.

Difficulty of Detection. Since transient servers' resource exhaustion caused by VSI-DDoS occurs mostly for a few milliseconds, it is difficult to detect using the current second-level monitoring sys-



(a) CPU usage without the VSI-DDoS attack

(b) CPU usage under the VSI-DDoS attack

Figure 3.2: The CPU usage of each server with/without the VSI-DDoS attack captured by *collectl* with the sampling rate of 1Hz. Notice that even under the VSI-DDoS attack, the CPU usage of each server is not saturated, which makes it difficult to detect.

tems such as *sar, vmstat*, and *top*. This is because those monitoring systems have a low frequency (i.e., check the utilization of resources at a granularity of seconds compared to the millisecond operation realm of VSI-DDoS). As shown in Fig. 3.2, although the CPU usage of each server under the VSI-DDoS attack is slightly higher (as in Fig. 3.2b) than the case without the attack (as in Fig. 3.2a), it still seems not to be saturated from the second-level monitoring system's perspective. In other words, the VSI-DDoS attack operates in a way that is difficult to detect with the current DDoS detection mechanisms and may continue to impact the convenience of users of web application services without being detected. While there has not been a report of damages caused by low-rate DDoS, such as VSI-DDoS, many security agencies are warning about the danger of stealthy and sub-saturating attack [11].

3.2.3 Key Assumption: Synchronization

Perhaps the biggest advantage of low-rate DDoS attack is stealthiness. Since this type of DDoS attack does not aim at complete depletion of the resource, it only needs a relatively small amount

of traffic. As such, the key factor for a successful attack is the concentration of the attack traffic. If the adversary is capable of concentrating attack packets in a shorter time, the attacker may need a smaller number bots and less traffic from each bot, which improves the effectiveness and stealthiness of the attack. Thus, in a botnet that consists of multiple bots, the synchronization of attack streams can be considered a key factor for the successful low-rate DDoS attack. In other words, if there is an excessive assumption on it, the anticipated effectiveness of the attack can be overestimated.

In VSI-DDoS, for example, to validate the attack, Shan *et al.* conducted experiments showing how varying parameters of the attack affect the response latency to legitimate users' requests. With multiple bots fully synchronized, they noted that tail latency is significantly increased when generated packets arrive at the server within a very short time window (a few milliseconds). By assuming an almost complete synchronization, they experimentally demonstrated that the 95th percentile response time of the target server increases by more than 1 second, which is the TCP RTO, under the VSI-DDoS attack. As a result, they demonstrated that the VSI-DDoS attack can be a threat to many web application services, by degrading their QoS guarantees.

3.2.4 Botnet Synchronization in the Wild

As we described, the assumption of synchronization in the evaluation can lead to inaccurate results. In fact, time synchronization in distributed systems is a challenging problem, and clock skew is, by default, an intrinsic feature and an unavoidable characteristic of those systems. Particularly, given the strong synchronization assumption, it is unclear how violating this assumption, even slightly, would affect the performance of the attack.

One may say that using an off-the-shelf time synchronization, such as the Network Time Protocol (NTP) [34] can bring an order to bots, to meet the assumptions for successful pulsing attack. However, in reality, such an approach would face various shortcomings. First, while NTP works nicely

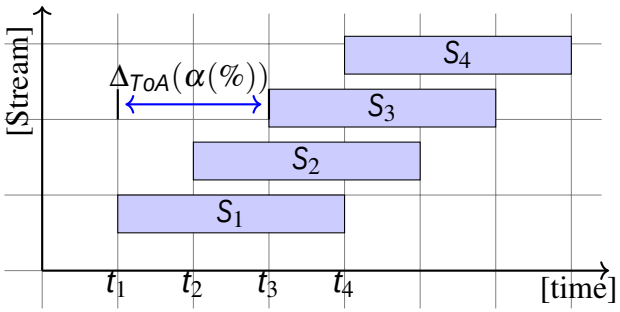


Figure 3.3: An illustration of botnet synchronization in [26]. Notice that S_i means the stream from each bot i and t_i means the arrival time of the first packet of each stream. $\Delta_{TOA}(\alpha(\%))$ corresponds to the maximum time difference between the first packets of the majority ($\alpha\%$) of attack streams. For example, the expression, $\Delta_{TOA}(90\%) = 50ms$, means that the first packets of 90% of attack streams arrive within 50ms.

in local area networks, achieving up to a millisecond level of accuracy under ideal conditions, it provides worse performance (tens to hundreds of milliseconds) on the Internet due to congestion and path asymmetry. Second, connecting to NTP servers on the Internet, while common, can be used as an indicator to trace bots back and detect them, pronouncing the approach unfeasible.

Recently, Ke *et al.* [26] introduced *CICADAS* which performs the highly sophisticated botnet synchronization in 2016. To amplify the impact of pulsing attack, a kind of low-rate DDoS attack, they designed a synchronization technique in the decentralized system. To evaluate the level of synchronization, they also introduced the concept of the differential time of arrival, denoted by Δ_{TOA}. As shown in Fig. 3.3, $\Delta_{TOA}(\alpha) = t_\alpha$ means that the first packet of $\alpha(\%)$ of the attack streams arrive at the target within t_α. From the Internet-wide experiments, they got the result that $\Delta_{TOA}(75\%)$ and $\Delta_{TOA}(90\%)$ are about 40ms and 60ms, respectively.

Despite the outstanding results, CICADAS still demonstrates that perfect synchronization among the bots is a very difficult goal to achieve. Per their approach, if there are four machines attacking the server with 50ms of burst, for example, and considering the worst-case scenario, the server is attacked from three (75%) machines at the same time for only 10ms, that is 50ms - 40ms, which values correspond to burst length and $\Delta_{TOA}(75\%)$, respectively. Furthermore, the accuracy of their

Table 3.1: The variables used in theoretical analysis of the VSI-DDoS attack.

Variable	Meanings
n	target web application server's capacity
t	ideal time of packet concentration (at target)
b	number of bots
L	length of burst time each bot sends
i	time interval between two consecutive packets
$k_1 \dots k_b$	required number of attack packets from each bot
$s_1 \dots s_b$	attack stream from each bot
$p_{s_1} \dots p_{s_b}$	number of packets arriving at the target during t

approach enforces a lower-bound on the burst time: if the burst gets shorter, the time period that the packets from multiple sources concentrate on the target will be shortened also, and even may not exist.

Even when existing approaches provide a perfect synchronization accuracy, network conditions are stochastic, where the interarrival time between packets sent from different hosts to the same server may vary depending on the network condition and paths.

3.2.5 Theoretical Analysis

In this section, we formally analyze how the concentration of attack streams arriving at the target is characterized. The main focus of this work is the bots' behavior required for a successful attack according to the degree of synchronization. Therefore, we don't take the legitimate users' behavior into account. The variables used in analysis and their meanings are described in Table 3.1.

Let n be the shortest length of queues in the web application server, which corresponds to the server's capacity. Thus, the goal of the adversary is to concentrate at least n packets at the server in a very short period of time, t, in order to cause packets to be dropped. Moreover, let b be the number of bots that the adversary utilizes to launch a pulsing attack and $k_1, k_2, \dots, k_b$ be the number of packets each bot should send to saturate the server's queue, such that $\sum_{j=1}^{b} k_j \geq n$. Since the

objective of this analysis is to compute the required average number of attack packets from each bot, for convenience we assume that all bots send the same number of packets, which means $k_1 = k_2 = \cdots = k_b \geq n/b$. Moreover, let L be the length of burst time each bot sends, where $L \geq t$. Assuming that the arrival time of packets in each attack follows a uniform distribution, the number of arriving packets from bot j within a very short time is $(k_j/L) \times t$. For example, assuming a bot j sends 150 packets (k_j) during each burst of 30 ms (L), the number of packets arriving in a very short time of 10 ms (t) is $(150/30) \times 10 = 50$.

Best-Case Scenario. First, we consider the best-case scenario from the adversary's point of view, which happens when all streams from b bots arrive at exactly the same time. In such a case, the number of packets that each bot should send can be easily computed as:

$$\left(\frac{k}{L} \times t\right) \times b \geq n, \text{ where } k \geq \frac{n \times L}{t \times b}$$

As shown, the longer the queue of the server (n) or the longer the burst (L), the more packets per bot the adversary needs to send for a successful attack. Conversely, the more bots the adversary uses (b) or the shorter the packet concentration time (t), the less traffic per bot the adversary needs.

However, this attack, which assumes a perfect synchronization, is difficult to achieve in reality, considering the aforementioned constraints. As such, in the following we will analyze the average case of the attack. Note that the worst case scenario happens when the adversary cannot fill the queue at the target (due to the huge difference in the arrival times of attack streams), and so there is no value of k that makes the target's queue full. As such, we do not deal with the worst-case in our analysis.

Average-Case Scenario. In the average-case scenario, the streams of attack packets arrive at the server within a regular interval of i, where i is assumed to be less than or equal to t for a more general analyses.

When the first packet of each stream arrives at the target at regular intervals of i, the maximum number of streams that can exist simultaneously is $L/i+1$ (refer to Fig. 3.3). In this case, it is assumed that $t_2-t_1=t_3-t_2=t_4-t_3=i$. If the short time t required to concentrate the attack packets is t_4-t_1, the stream S_1 would send packets during a time interval of length $3i$, while S_2 and S_3 would send during $2i$ and i, respectively.

Let p_{s_j} denote the number of packets arriving at the server from the stream s_j during t. We can then generalize the maximum number of packets (p) during t as:

$$\begin{aligned} p &= p_{S_{(L/i)+1}} + p_{S_{L/i}} + \cdots + p_{S_2} + p_{S_1} \\ &= \frac{k}{L} \times i + \frac{k}{L} \times 2i + \cdots + \frac{k}{L} \times (\frac{L}{i}+1) \times i \\ &= \sum_{b=1}^{(L/i)+1} \frac{k}{L} \times b \times i \end{aligned}$$

In order to saturate the server, the number of packets (p) should be equal to or greater than the length of shortest queue (n):

$$\begin{aligned} p &= \sum_{b=1}^{(L/i)+1} \frac{k}{L} \times b \times i \geq n \\ \therefore \frac{k}{L} &\times \frac{1}{2} \times (\frac{L}{i}+1) \times (\frac{L}{i}+2) \times i \geq n \end{aligned}$$

From this inequality with fixed values of L and n, we can deduce that a larger value of i increases the required value of k to satisfy the given inequality. Since i is the time difference between attack streams, it can be indirectly considered as a level of synchronization of a botnet. A large value of i implies that the bots are loosely synchronized. In this regards, the above inequality shows that each bot needs to send a larger number of attack packets for larger i. Conversely, as the value of i decreases (tight synchronization), the number of packets k is also reduced. Given that, in the average-case scenario, there may be no overlap between some of the attack streams. Thus, the number of bots or the number of attack packets per bot should increase further.

Based on the above analysis, we can conclude that such variance in synchronization results in less effectiveness of the pulsing attack. For this reason, we pose the following question: under a realistic condition, with worse synchronization inaccuracy, how effective is the pulsing attack? We answer this question in the rest of this paper by evaluating the VSI-DDoS attack.

3.3 Experiential Setting

To measure the impact of a more realistic time synchronization in botnets, taking a slight deviation from ideal scenarios into account, we build a testbed similar to the one in [56]. While it is difficult to match the exact specifications of the hardware used in their experiments (partly due to the lack of details), in our work we replicated the behavior of servers, legitimate users, and bots, which are more crucial to the reproducibility of their results. Particularly, because the work at hand is mainly concerned with understanding the effect of the VSI-DDoS attack depending on the degree of synchronization, we focus on how the change in synchronization level, rather than individual values, affects the performance of the VSI-DDoS attack. As a result, in the rest of this work, and given that other parameters (e.g., packet size) are the same in all of our experiments, we use the burst time (the difference between start and end of a VSI-DDoS attack cycle) to characterize the attack intensity. We note that given the main goal is to compare the relative latency for different burst times, the exact matching of the hardware specifications as in [56] is unnecessary.

3.3.1 System Setup

As shown in Fig. 3.4, our system consists of the RUBBoS 3-tier architecture for a server-side setup (i.e., web server, an application server, and a DB server), bots, legitimate users. In the following, we elaborate on each of those system components.

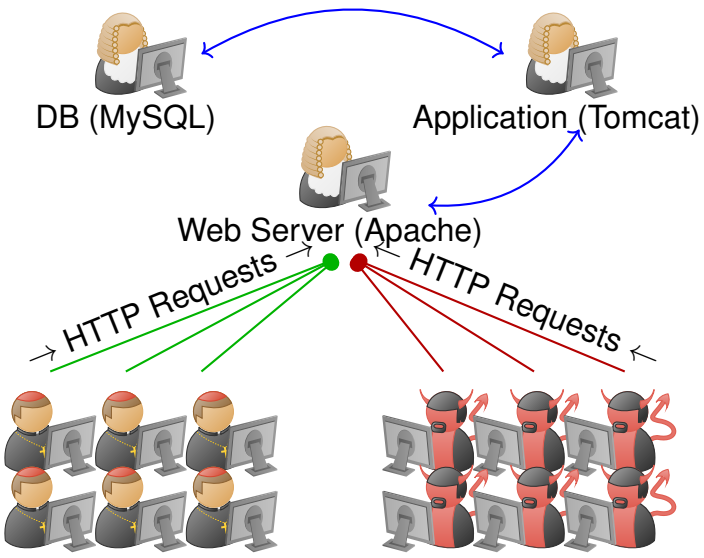


Figure 3.4: The overview of the communication among RUBBoS benchwork (consisting of RUBBoS 3-tier architecture; front-end web server, back-end DB server, and application server), legitimate users, and botnet.

Web Application Server. As shown in Fig. 3.4, we build a server system that provides web services using RUBBoS, a popular n-tier web application benchmark [52]. Following the typical 3-tier architecture, an Apache web server, a Tomcat application server, and a MySQL database server were deployed on the Vultr cloud [68]. Each server is created as an independent instance with the same specification of a 2.4GHz single core virtual CPU and a 1.8GB of Random Access Memory (RAM). The servers interconnect using 1Gbps links.

Legitimate users. The behavior of the legitimate user is imitated using the workload generator of RUBBoS. Three cloud instances mimic the behavior of a total of 1,050 legitimate users, each for 350 users. According to the configuration of RUBBoS, the users surf web pages following a Markov chain model. Behaviors such as searching the bulletin board, writing a new article, leaving a comment, etc. are continuously generated following the probability model of Markov process.

Bot for the VSI-DDoS Attack. Apache Bench (AB) [5] is used to create a bot that performs the VSI-DDoS attack. The purpose of bots is to intermittently send the given number of HTTP

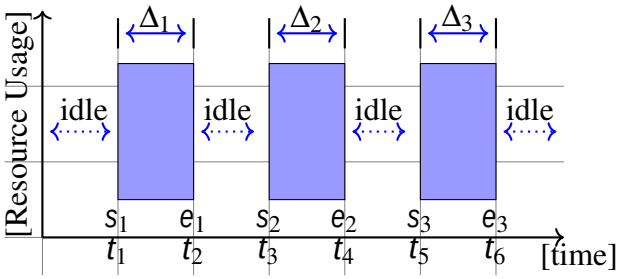


Figure 3.5: An illustration of VSI-DDoS cycles, where a large number of packets is sent over a short period of time (e.g., $e_1 - s_1$), resulting in high attack intensity. In our work, the intensity Δ is characterized by the degree of concentration of HTTP requests from bots, where each chunk is 100 HTTP requests.

requests to the server and to trigger transient saturation, thereby delaying responses by the server to legitimate users.

In the default setting, AB does not use keep-alive, which means that a new TCP connection is created for each HTTP request. Therefore, every HTTP request behaves like a different user and tries to connect to the server to use the service.

In addition, we made the botnet operate on the single machine rather than be deployed over the multiple cloud instances. In this research, since it is not our focus to implement the sophisticated synchronization of botnet itself, we implemented the bot on a single device so that the degree of synchronization can be accurately adjusted by simply modifying the configuration. The average of the round trip time (RTT) between the bot instance and the server is about 0.5 ms (measured using *ping*), which means that the network latency in one-way packet transmission is about 0.25 ms. We also measure the actual concentration of botnet packets with the latency through preliminary experiments.

3.3.2 Intensity Values

In our experiments, the concentration of HTTP requests made by bots is set as a variable parameter, incorporating a variable accuracy in time synchronization between the originating bots. As shown in Figure 3.5, the VSI-DDoS attack is made in such a way that HTTP bursts are sent from bots to the server intermittently (i.e., within a very short period of time separating the sending of the first packet from the bot and the arrival of the last packet to the server). In the figure, each chunk (blue rectangle) means the given number of HTTP requests (e.g., 100 HTTP requests in our experiments) is sent from the bot periodically (resulting in intensity Δ_i), with an ideal time between every two consecutive cycles (2 seconds).

To express the concentration of packets from the bot, in this work, the time difference between the first request and the last request in the same chunk is denoted as *intensity* for convenience. For example, if the intensity is 45ms, it means that all 100 HTTP requests are sent to the server within 45ms. A high intensity means that the attack is concentrated within a shorter period of time; i.e., the bots are well synchronized.

As described in section 3.3.1, we used AB for launching the VSI-DDoS attack. AB supports sending a given number of HTTP requests in a short time to a URL to check the performance of the server hosting that URL. As such, we can control this process by feeding the number of packets sent from the bot to the server, the concurrent level of transmission, and time limit as options [5]. However, AB does not fully respond to the input variables due to the issues such as the limited network resource. In our study, in order to analyze the correlation between the degree of synchronization and the impact of the VSI-DDoS attack, the packet transmission from bots is key variable we need to accurately control. Thus, we conducted a preliminary experiment to figure out the behavior of AB in detail.

While running the AB and changing the input option value of the concurrent level, we captured the actual transmission time of 100 HTTP packets using `tcpdump`. Splitting the entire transmis-

sion into multiple tasks (two AB instances where each sends 50 requests), we use a shell script to generate various intensity levels. We conducted the measurements with six different settings ($S_1, S_2 \ldots S_6$) and the burst length was calculated by measuring the time of the packets captured by `tcpdump` with each setting. In order to minimize the impact of the external network environment, the web server and the bot are configured to be instances located in the same locale in the Vultr (perhaps locally connected).

In the experiment of sending 200 chunks (each chunk of 100 HTTP requests), the burst length ($e_i - s_i$) has the distribution as in Fig. 3.6. In the figure, Δ_S (blue) is the length of the burst measured at the sender, a bot, and Δ_R (yellow) is the burst length measured at the receiver, the web server. Each box shows the distribution from the upper 25% to the lower 25% of the burst length, while the black line in the box means the average value of all the results (200 chunks). Two horizontal lines above and below the box represent the maximum and minimum values, respectively. In the figure, we can see that the packet distribution in the sender and the receiver is similar under the strictly controlled environment. The average burst lengths with each setting measured at the bot were 11.2ms, 45.64ms, 63.37ms, 79.05ms, 88.84ms, and 129.53ms, while the average values were 11.27ms, 45.51ms, 63.53ms, 79.15ms, 88.93ms, and 130.01ms at the web server. The maximum value of the standard deviation of all distributions was 4.22 (in the receiver with the setting S_2), the other values are less than 4, which means that the packets from the bot arrive at the server in a similar pattern under the same configuration. In the rest of the sections, for convenience, the intensity values with each setting are denoted by i_1, i_2, i_3, i_4, i_5, and i_6, respectively.

To understand the result where the degree of synchronization is more realistic, we compared the response time experienced by legitimate users for each case with the intensity value as shown above. Based on previous studies on synchronization, among these values, i_1 (about 10ms) can be considered a very difficult level of synchronization to be achieved by the attacker (e.g., impossible to achieve by NTP [34] and state-of-the-art botnet synchronization [26]), i_2 (about 45ms) and i_3 (about 65ms) can be considered difficult levels, i_4 (about 80ms) and i_5 (about 90ms) can be

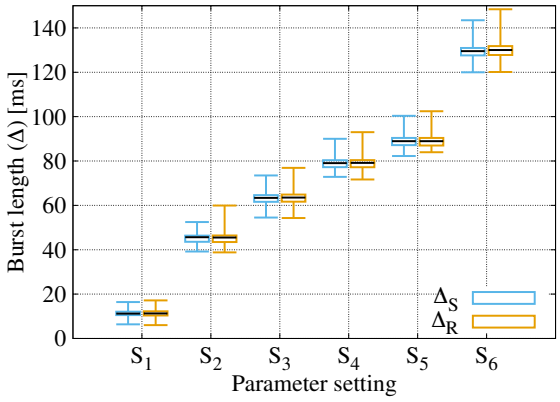


Figure 3.6: Measurements of actual burst length with different setting; 200 repeated measurements for each setting. Δ_S and Δ_R correspond to the measured burst lengths from the sender (bot) and the receiver (web server), respectively. The box represents the distribution from the upper quartile to the lower quartile, and the black bar represents the mean value.

considered achievable levels, and i_6 (about 130ms) and above can be considered more realistic levels even with the network dynamics.

3.4 Results

3.4.1 Response Latency Measurement

In order to investigate the effect of the VSI-DDoS attack on the QoS, we performed each simulation for the same time (50 seconds) in all scenarios with different intensity values. Ten experiments were repeated (totally 500 seconds) for each setting to ensure a sufficient amount of data. During each simulation, we measured the response time at the legitimate users from the web service and counted the number of packets with response time longer than 1 second, which indicates the higher latency due to the VSI-DDoS attack. The table in Fig. 3.7 represents the total number of HTTP request-response pairs between the web server and all legitimate users.

Because the simulation was conducted for the same time period, and as shown in the table, the total number of HTTP request-response pairs (c_1) remained similar except for the cases with the i_1

Intensity	c_1	c_2
i_1	69,090	9,643
i_2	71,530	6,080
i_3	73,006	2,644
i_4	72,744	1,655
i_5	73,574	1,463
i_6	73,049	1,097

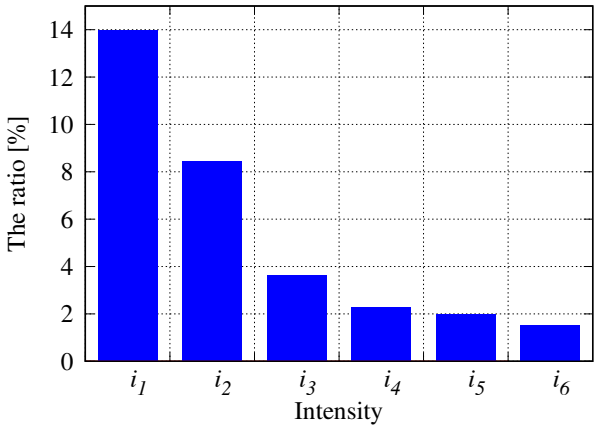


Figure 3.7: Statistics of HTTP requests and responses; c_1 corresponds to the total number of HTTP request-response pairs between all legitimate users and the server, and c_2 corresponds to the number of HTTP responses with over 1 second time (left). The percentage of legitimate users response with time over 1 second out of the total HTTP responses (right).

and i_2 that have tight intensities. The reason for the two relatively small values can be found in the delayed answer that occurs under the VSI-DDoS attack. As in the table, both i_1 and i_2 intensities lead to a larger number of delayed responses (c_2). Therefore, in the case of an HTTP request in which the response is not returned to the legitimate client during the experiment period of 50 seconds due to the delay caused by the VSI-DDoS attack, it is not included in the c_1 value shown in the table. In other words, low c_1 values with the i_1 and i_2 intensities are the result of frequent latency occurred by tightly synchronized attack not reflected in statistics.

Fig. 3.7 (right) shows the degradation of QoS provided to the legitimate user as a latency guarantee when the intensity of the bot attack is changed. In particular, the figure plots $c_2/c_1 \times 100$ for the different intensity values in the table. When the HTTP request intensity of the bot is i_1, about 14.0% of legitimate users' responses received from the server exceed 1 second, which corresponds to an ideal setup of the VSI-DDoS attack. As the intensity of the attack gradually loosened, the effect of the VSI-DDoS attack quickly decreased.

This is, the ratios of HTTP requests that have a response time over 1 second are 8.4% for the intensity of i_2,3.6% for i_3, 2.3% for i_4, 1.9% for i_5 and 1.5% for i_6. This shows that if the intensity

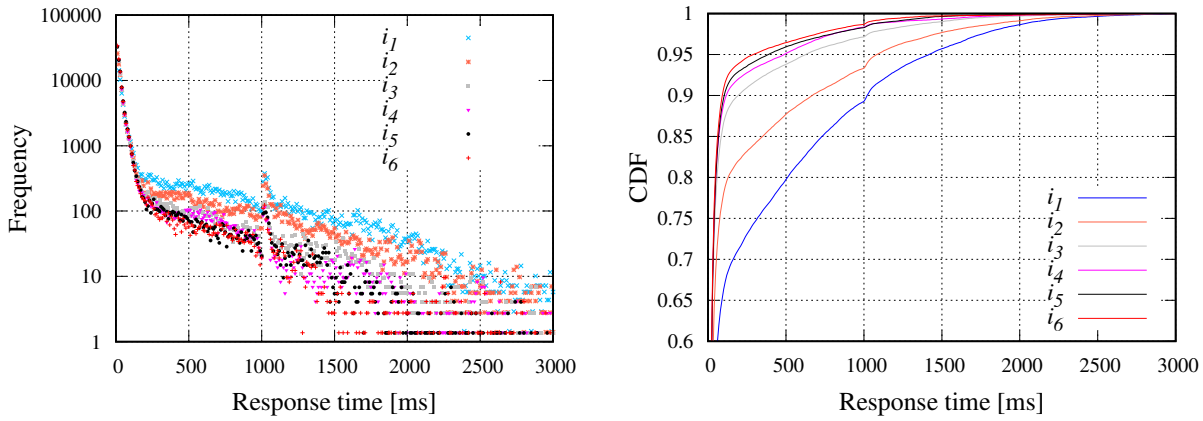
of HTTP requests sent from the bot change slight, to reflect loosely synchronized bots, it is difficult for the attacker to obtain the desired result. For example, when the intensity is i_5, the legitimate user's VLRT rate (over 1 second) is only about half the rate at i_3, which means that the attacker should use more number of machines to get the desired effect on the server.

3.4.2 HTTP Response Time Distribution

While the results above demonstrate the key insight of our experiments, Fig. 3.8a shows the HTTP response time that the legitimate users received from the server. In the graph, the x-axis represents HTTP response time (ms), and the y-axis represents the number of packets having the corresponding time. Because the total number of HTTP request-response pairs is different in each case, for an accurate comparison, the graph was created by normalizing the ratio of observed results in the experiment to a total of 100,000 HTTP packets. As shown in the graph, in all scenarios the distribution increases from 1 second of response time due to TCP retransmission under the VSI-DDoS attack. Considering that all graphs are plotted on a log scale, the actual amount of change will vary highly depending on the intensity of the bot's attack.

From the same figure, we notice that when the response time is equal to or greater than 1,000 ms, the number of HTTP request-response pairs increases more sharply as the intensity is higher (e.g., i_1, and i_2). The degradation of QoS with the various values of intensity can be seen more clearly in Fig. 3.8b. In the CDF representation of the response time experienced by legitimate users, the degradation of QoS caused with each intensity is significantly distinguished.

This means that HTTP requests that are concentrated on the server within a short period of time saturate the server more frequently and cause more frequent response delay to legitimate users. Conversely, as packets from the bot arrive over a wider interval of time, the actual effect of the VSI-DDoS attack is weakened.



(a) The distribution of response times.

(b) The CDF of response times.

Figure 3.8: The distribution and CDF of response times under the VSI-DDoS attack.

3.4.3 Requirement for Successful Attack

From the above results, we observe that the effectiveness of low-rate DDoS attack significantly decreases when the network is not tightly synchronized. Nevertheless, if an adversary still wants to negatively impact the server through a low-rate DDoS attack, the adversary should either 1) send more packets from each bot, or 2) use more bots. As such, we conduct an additional experiment to estimate the required number of packets sent by a botnet to achieve their purpose in this real scenario.

Goal. We set the adversary's goal to make the 95th and 99th percentile response longer than 1 second. Here, the "95th percentile response time more than 1 second" means that the top 5% longest response time exceeds 1 second. Similarly, 99% means that the top 1% longest response time is more than 1 second. As such, making the 95th percentile response more than 1 second is more difficult than making the 99th percentile more than 1 second (more users experience longer delay)—note that we did not invent those terms, and for consistency we used the terms utilized in the literature. We measured the number of the required attack packets to achieve this goal with various burst lengths. As we analyzed in subsection 3.2.5, we can intuitively deduce that more attack

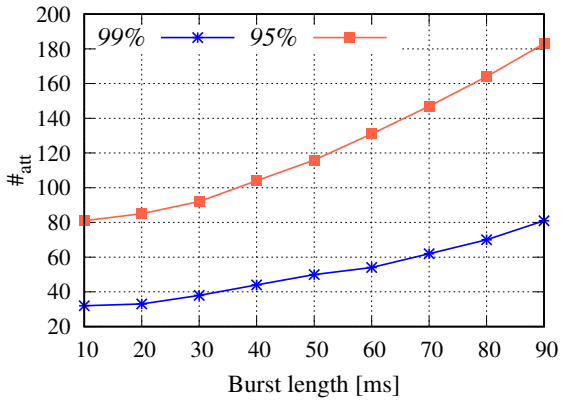


Figure 3.9: The required number of attack packets to make the 95th and 99th percentile response longer than 1 second. Notice that $\#_{att}$ corresponds to the number of attack packets to achieve the goal.

packets will be needed when the burst length is longer (loose synchronization). In the following, we demonstrate the correlation between the two factors through an empirical measurement.

Setting. We set up a similar testbed for this experiment as those used in the prior experiments. We configured the environment using the same cloud instances and kept the client's behavior the same (with the same probability of page transition). However, in this experiment we implemented the adversary using Python to more flexibly adjust the sending time of packets. By setting a uniform delay between consecutive transmissions, we adjust the length of the burst time. The delays inserted are uniform to accurately adjust the total length. We used these settings because the length of the burst is the most important parameter in our experiments, and minor differences may appear if other random generation methods are used. However, given that the length of the burst itself is very short (tens of milliseconds), it is difficult for a random-based delay to make a big difference in packet concentration. Moreover, we observed that there is a limit to controlling millisecond-level time precisely in our program, and for that we conducted various repeated measurements to account for the variability.

Results. Fig. 3.9 shows the results of our measurements. At both the 95% and 99%, we can see that as the degree of synchronization gets weaker, more attack traffic is required to reach the adversary's goal. Since the main purpose of the attack is to cause an overflow in the queue on the target server, it is important to concentrate the packets in a short time.

In our evaluation, and to make the 95% percentile response more than 1 second, we can see that the attacker needs at least 80 attack packets with a burst length of 10 ms and more than 180 packets with a length of 90 ms. This shows that at 90 ms, it takes more than twice as many packets to achieve the same goal in the most ideal case (10 ms). Even with 50 ms and 60 ms burst length intervals, which are considered relatively realistic, the adversary needs about 1.5 times as many packets as in the ideal situation. These differences shown in the figure are not small, which means that the adversary's goal is significantly harder to achieve in reality.

The trend also appears similar for the 99th percentile response. In the most realistic case (90 ms), the attack requires nearly twice as many packets as the ideal, and the cases with 50 or 60 ms of burst length require about 1.5 times as many packets. The results show that it is possible for an attacker to considerably delay the 95th percentile or 99th percentile response by launching a pulsing attack. However, it also shows that there is a big difference in the number of bots (or traffic per bot) required for a successful attack per the level of synchronization.

3.4.4 Multi Bots Synchronization

In the previous section, we performed the evaluation only using a single bot to generate attack traffic with a delay between consecutive packets. For a more realistic evaluation, we build our experimental environment with multi bots and demonstrate how the multi bots synchronization would make it even more difficult for the adversary to achieve his goal.

We built an environment where two bots were used to launch an attack for an intuitive comparison with an attack using a single bot as shown in the previous section. We configured the bot with

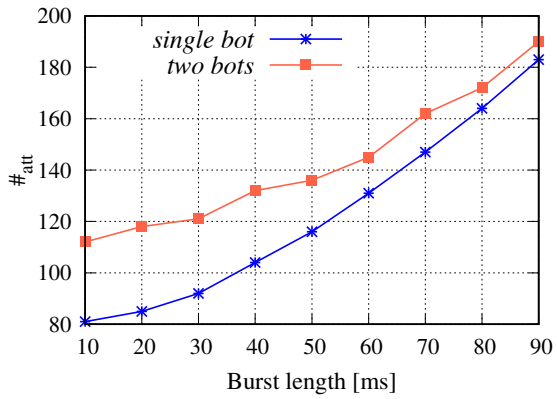


Figure 3.10: The difference in the required number of attack packets in a single and multi bot environment to make the 95th response longer than 1 second. Notice that $\#_{att}$ denotes the number of attack packets to achieve the goal.

two cloud instances. For each burst, the command is sent from a master bot to a secondary bot, and the secondary bot starts sending bursts as soon as it receives the command. The command is sent from the master bot to the secondary bot every burst, and the burst is sent immediately upon receipt. We note that while utilizing state-of-the-art technology for synchronization is possible, that is unnecessary, and one can assume a lower bound on the latency; given that the Round-Trip Time (RTT) between the two hosts is less than 10 ms, this can be similar to a very high level of synchronization. In the experiment, two bots sent the same number of attack packets to the target server during the same length of burst. Same as in the previous section, in each stream, each packet was sent with regular intervals.

Fig. 3.10 shows the number of required packets to launch a successful attack (95th response time longer than 1 second). In this figure, the blue line shows the case when the adversary uses a single bot, while the orange line is when the adversary uses two synchronized bots. Given the burst length of 10 ms, when the adversary uses two bots, we note that the adversary needs about 1.5 time as many packets to achieve his goal. On the other hand, as the length of the burst increases, the difference between the two cases decreases. This means that the longer the burst, the smaller the effect of the degree of synchronization, since the regular interval between packets increases.

3.4.5 Summary of evaluations

Notice that the above result was measured when the only bot (or bots, in the case of the realistic scenario) sends the attack packets with a uniform delay. That is, the above measurement mainly reflects the difference due to the time skew between bots, but hardly reflects the difference due to network dynamics. In real networks, factors such as the bot's physical location and routing environment can make synchronization more difficult, and in fact, an attacker may need more packets than measured in the previous sections to achieve the same goal.

As a result, we emphasize that the above experiments show that low-rate DDoS attack is possible, although difficult to achieve as previously stated. Regardless of the number of bots and attack packets required, a low-rate DDoS attack can certainly cause an inconvenience to users, which can pose a threat to web service providers. However, we state, based on the above measurements, that the threat posed by the low-rate DDoS attack may have been exaggerated under unrealistic assumptions about the behavior of bots and what is considered realistic with respect to real world synchronization capabilities. When measuring the effectiveness of an attack in a more realistic scenario, we can see that there is a lower level of risk than previously stated, and it would be desirable to find a countermeasure in place with this exact understanding.

3.5 Discussion

Our findings have shown that it is difficult to obtain sufficient performance of the VSI-DDoS attack in an environment where synchronization between bots is imperfect. To this end, we envision various approaches to cope with this limitation:

3.5.1 Improve Bots' Synchronization

Improving the bots' synchronization is perhaps the most obvious option to improve the attack. Better synchronization can be achieved by applying highly advanced techniques. As distributed systems evolve, with the deployment of large-scale and massively distributed network anchors, it is anticipated that time synchronization would witness a great improvement over the few upcoming years. However, as it stands today, the state-of-the-art techniques for time synchronization do not seem to provide the level of synchronization required for the attack, and the current techniques seem, above all, sensitive to network characteristics to provide a millisecond-level of time synchronization accuracy required for the attack. We note that other techniques; e.g., nanosecond-level synchronization, have been developed for a special purpose (e.g., datacenter), although most of them require additional hardware, making them difficult to apply to or exploit by botnets.

3.5.2 Increase HTTP Requests from Each Bot

The second approach to address the problem is to increase the HTTP requests generated by each bot, which would result in higher attack intensity. Since the packets generated on one machine are likely to arrive at the server almost at the same time, they can be effective regardless of the synchronization of the botnet. However, when the number of HTTP requests per bot significantly increases, the bot may have different pattern from legitimate users. In addition, repetitively sending a larger number of packets can expose the bot to the Intrusion Detection System (IDS) as an anomalous activity [21, 73].

3.5.3 Increase The Number of Bots Participating in Botnet

The third option the adversary can choose to launch the VSI-DDoS attack is to increase the number of bots participating in the botnet. As more bots participate in and more attack streams are created,

more packets can arrive at the server concurrently under the same synchronization level. While this approach can circumvent the anomaly detection, it would lead to two side effects: 1) increasing the cost of the attack, and 2) making it more difficult to apply any (loose) time synchronization among a large number of bots.

3.6 Summary

In this work, we analyzed the impact of loose bots synchronization on low-rate pulsing attack; an emerging type of DDoS attack that is capable of bypassing existing defenses. Because it requires a low volume of attack traffic, many concerns have been raised about these attacks and their high stealthiness. For example, the VSI-DDoS attack, one of low-rate pulsing DDoS, has showed its effectiveness and stealthiness through empirical analyses. However, in this work, we demonstrated how a more realistic evaluation of such an attack under a more realistic synchronization scenario of botnets, as the case on the Internet today, reduces the effectiveness of the pulsing attack, including VSI-DDoS. We note that while this attack, in essence, can actually degrade the user's QoS, in reality it requires a stronger adversarial assumption, characterized by tight bot synchronization for success. Through measurement under realistic assumptions, we demonstrate that even a moderate imperfections in bots time synchronization would reduce the impact of the attack, and even pronounce it ineffective. Specifically, the effect of the attack at realistic synchronization level seems to make adversary's goal difficult to achieve.

CHAPTER 4: BEHAVIORAL ANALYSIS OF OPEN DNS RESOLVERS[1]

In this chapter, we conduct a large-scale measurement and **behavioral analysis** of open resolvers. Unlike previous studies that were limited to understanding the spatial distribution of open resolvers, we analyze the behavior of these resolvers in detail using two datasets collected through active scans and probing using a pre-configured network setup. By elaborating on the new discoveries provided by an unconventional analytical approach, we underscore the need for a multifaceted analysis.

4.1 Motivation

The Domain Name System (DNS) is a hierarchical distributed naming system and is a pillar of today's Internet. The primary goal of DNS is to supply a mapping between domain names and associated IP addresses. For instance, once a user types a domain name, e.g., www.example.com, into a web browser, the domain name will be mapped, by a set of DNS servers, to the associated IP address, e.g., 1.2.3.4. Almost all Internet services depend on DNS to connect users to hosts by resolving DNS queries. However, because DNS is an open system, anyone may query publicly accessible resolvers, called open resolvers. The operation of those resolvers is required in rare cases; mainly public services such as Google DNS [2] and Open DNS [1]. However, prior studies identified millions of publicly-accessible open resolvers on the Internet [29, 43]. It is shown that open resolvers are an attractive target for attackers to launch a wide variety of attacks, such as DNS amplification [13], DNS manipulation [4], etc..

Open resolvers can be used as a stepping stone for many attacks. For example, a report by Cloud-

[1]This content was reproduced from the following article: J. Park, A. Khormali, M. Mohaisen, and D. Mohaisen, "Where Are You Taking Me? Behavioral Analysis of Open DNS Resolvers," *In Proceedings of the 49th IEEE/IFIP International Conference on Dependable Systems and Networks*, IEEE/IFIP DSN 2019, Portland, OR, US, 2019. The copyright form for this article is included in the appendix.

Flare highlights a 75Gbps DNS amplification DDoS attack in 2013 [10] using open resolvers in the wild. Takano *et al.* [62] also show the potential of DNS open resolvers for attacks by investigating the software version installed on those resolvers. Moreover, several previous studies demonstrated that DNS manipulation is widely used for malicious purpose by adversaries [28, 53], censorship by governments [48], or even monetary benefits [72]. These works showed that open resolvers in the wild expose their vulnerabilities to the adversaries and users alike.

To this end, we present in this work an up-to-date view of open resolvers' threats through an in-depth analysis. Unlike the prior work that only dealt with a small subset of accessible open resolvers [48, 3, 4, 59], we attempt to investigate all open resolvers over the Internet. Moreover, we focus on the behavioral aspects of open resolvers, which provides a deeper understanding of threats posed by them. Mutual reliability is the most important factor in DNS, where domain name is queried and a response is obtained. This reliability can be guaranteed only when a role-based behavior is performed. Observing the behavior of open resolvers is a measure of their security and DNS reliability as a whole.

Contributions.. In this work, our interest is to understand how big the threats of open resolvers actually are by measuring and analyzing the behaviors of open resolvers. Our main contributions are as follows:

1. We conducted a comprehensive measurement over the entire IPv4 address space to understand the behaviors and threats of open resolvers around the world. An Internet-wide measurement allows us to have an empirical understanding of DNS open resolvers independent of arbitrary generalization. We found that there are about 3 million recursive resolvers that do not require any authorization for domain name resolution.

2. Through quantitative analysis, we found that many open resolvers generate DNS responses in a way that deviates from the standard. More specifically, the responses from open resolvers marked fields in DNS response header, such as the Recursion Available bit, the Authoritative

Answer bit, and the response code, improperly.

3. Through measurements, we report empirical results of DNS manipulation by open resolvers. By validating the open resolvers' answers, we discovered that more than 26 thousand open resolvers redirect users to malicious destinations reported as malware, phishing, etc..

4. For a temporal contrast, we use a dataset collected in 2013. We found that the number of open resolvers has significantly decreased, while the number of resolvers manipulating responses remains the same, and the number of open resolvers providing malicious responses has rather increased. This result shows the prevalence of open resolvers as a threat, despite their decrease in number.

5. We conduct a further analysis of the packets captured at the authoritative name server. By matching each flow with the same subdomain, we uncover the behaviors of open resolvers in detail, including the use of forwarder and the manipulation of answers.

4.2 Background

4.2.1 DNS Functionality

When a user wants to access a website on the Internet, she can do that by typing the domain name corresponding to the website, such as www.example.com, into a web browser's address bar. However, computers do not communicate using domains in a string form directly, but using numerically formed addresses, e.g., Internet Protocol (IP) address. Users, on the other hand, cannot memorize numerical addresses easily. To address this issue, Mockapetris [35] introduced the basics of DNS, which enables users to easily type natural language strings instead of numeric addresses for websites. In DNS operation, a human-readable domain name is mapped into a machine-readable address, e.g., IPv4 or IPv6 address.

Due to its convenience, scalability, and resilience, DNS has become an essential component of

the Internet. Many users access websites with the help of DNS without being aware of it. Such characteristics mean that DNS components with malicious intent can be a significant threat. A well-formed DNS infrastructure can be used for malicious purposes, such as creating a command & control channel of malware that uses Domain Generation Algorithm (DGA) [64]. Moreover, a miscreant may pose as an open DNS resolver and participate in the resolution process. To clarify why it can be a threat, we briefly describe the operation of DNS in the following.

4.2.2 DNS Resolution

For efficient and stable operation, DNS was designed as a hierarchical and globally distributed system that consists of the root, Top-level Domain (TLD), and authoritative name servers. Each of these servers is partly involved in converting human-memorable domain names to machine-recognizable addresses.

The overall resolution process is shown in Fig. 4.1. DNS resolution begins once a user attempts to access a web service using its domain name. DNS uses cache for performance, and when a domain name mapping is not cached in the local cache or the host table, the local resolver initiates a DNS query to the recursive resolver to retrieve the corresponding IP address to the domain name. The recursive resolver starts by asking root, then TLD, then the authoritative name servers.

Steps ② through ⑦ show the typical resolution process. The root server is the first server that receives a query from the recursive resolver, in step ②. The root servers are globally distributed and they maintain the IP addresses and location of TLD name servers. In step ③, the root name server replies to the query with the appropriate list of TLD servers for .com. In step ④ and ⑤, the recursive server sends a query for example.com and the .com TLD server responds with the IP address of the given domain's authoritative name server. In step ⑥ and ⑦, the recursive resolver communicates with the authoritative name server of example.com to find the address of www.example.com. Finally, the translated IP address of the requested domain name is forwarded

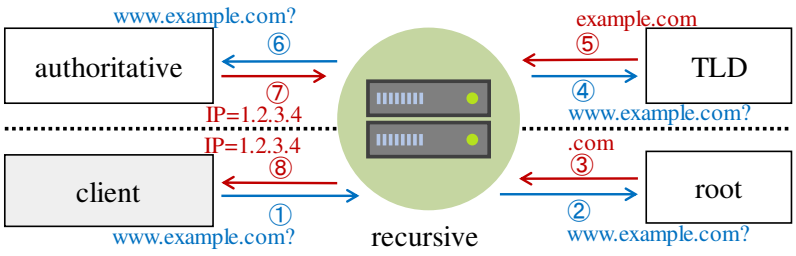


Figure 4.1: Illustration of DNS resolution over recursive, root, TLD, and authoritative name server. The texts and arrows in blue correspond to DNS queries, while those in red correspond to DNS answers.

to the local resolver. As a result, the browser can send a Hypertext Transfer Protocol (HTTP) request to the website to retrieve its contents.

4.2.3 Threat of Open Resolver

As described earlier, the recursive resolver is responsible for the recursive translation of domain names into IP addresses on behalf of clients. Among these recursive resolvers, open resolver is accessible by anyone on the Internet for resolution. Due to the role a typical recursive resolver plays in the resolution process, open resolvers are becoming a major threat to the security and resilience of the Internet. The rest of this section are details on how open resolver are exploited; e.g., for DNS amplification attack and DNS manipulation.

DNS Amplification Attack. The DNS amplification attack is a DDoS attack performed by exploiting the large difference between the size of a typical DNS query and the corresponding response. Originally, DNS had a packet size limited to 512 bytes. However, due to recent update [14], it is now possible to have more than 512 bytes in DNS responses.

In addition to the ‘A’ type query, which is commonly used to request the IP address of a webpage, there are also other types of DNS queries: ‘MX’ for requesting mail server information, ‘CNAME’ for requesting the canonical name of the server, etc.. Moreover, ‘ANY’ type DNS query requests

information about all domains managed by an authoritative name server including ‘A’, ‘MX’, and ‘CNAME’. If the authoritative name server manages a larger number of domains, the larger DNS response will be replied to the ’ANY’ type query. Moreover, the standard DNS resolution is unauthenticated, which means it is possible for an adversary to generate a DNS query with a spoofed address as a source. Because the DNS response is returned to the source of the query, IP forgery would mean that someone who did not issue a given query may receive an overwhelming number of responses.

DNS amplification attacks use the above two features of open resolvers. ‘ANY’ type DNS queries with a victim’s IP address as a source are sent to the open resolver, resulting in a concentration of DNS responses to the victim. An attacker can simply send hundreds of DNS queries to open resolvers to exhaust the victim’s bandwidth without having to create a huge amount of packets for a direct DDoS attack. That is, in such an attack, the open resolver acts as an attack amplifier.

DNS Manipulation. Another viable threat due to open resolvers is DNS manipulation. Users typically trust the results that an open resolver provides as a result of a recursive resolution. In other words, the IP address contained in the DNS response is considered as a correct address of the given domain name. However, an attacker can exploit an open resolver to provide a manipulated result to the legitimate users. Instead of the genuine page the user wants to access, a false DNS response may mislead the user to a similar phishing page created by the attacker to distribute malicious program or to steal one’s credential. Even when the attacker does not own the open resolver, he may produce the same effect by injecting the manipulated record into other existing open resolvers.

4.3 Measurement Setting

The goal of this work is to answer the following questions. 1) How many open resolvers exist over the world? 2) Do open resolvers behave correctly? 3) How do such behaviors pose a threat

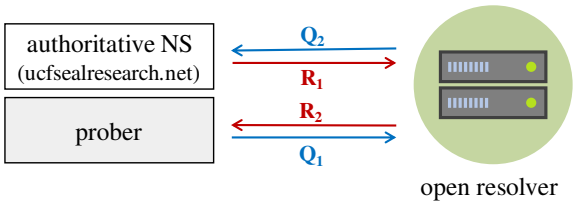


Figure 4.2: The flow of DNS request and response packets among the prober, authoritative name server and open resolver. Notice that `Q1` and `R2` are captured at the prober by modified Zmap, while `Q2` and `R1` are captured at the authoritative name server by tcpdump.

to Internet users? To answer these questions, we analyzed DNS responses obtained using an open resolver probing system, which we describe in the following.

4.3.1 Measurement System

The overall flow of open resolver probing is shown in Fig. 4.2. In this figure, the flow of `Q1`, `Q2`, `R1`, and `R2` corresponds to the DNS query from the prober to the open resolver, the DNS query from the open resolver to the authoritative name server, the DNS response from the authoritative name server, and the DNS response from the open resolver to the prober, respectively. The root name server and the TLD name server are not shown in this figure because they are out of the scope of this study. The communication with both servers takes place in the time between `Q1` and `Q2` to find the address of the authoritative name server. To gather all flows in Fig. 4.2 during our measurements, we built and controlled two components: a prober and an authoritative name server. In the following, we elaborate on the details of each component.

4.3.2 Open Resolver Prober

A prober is responsible for sending DNS queries to the entire IPv4 address space (`Q1`) and collecting responses from open resolvers (`R2`). The `Q1` messages generated by a prober include the subdomains underneath *ucfsealresearch.net*, which is under our management.

To perform DNS probing, we modified ZMap [16], an open-source fast Internet-wide scanner. In theory, ZMap is able to probe the entire IPv4 address space within an hour. To cope with our limited bandwidth, I/O constraints, etc., we performed a probing at 100k packets-per-second (pps). We implemented a systematic probing by combining the latest ZMap with the subdomain generation method described in section 4.3.4.

4.3.3 Authoritative Name Server

Upon receiving a DNS query, the open resolver starts a recursive resolution. The interpretation of the domain name proceeds in the order as shown in Fig. 4.1. Among the components that make up the whole DNS, it is impossible to build a root name server or a TLD server by ourselves, so we built an authoritative name server to observe the behavior of open resolvers. The authoritative name server is responsible for the translation of subdomains that belonged to the Second-Level Domain (SLD) in the DNS query. A prober generates DNS queries that include our SLD, which makes our authoritative name server participate in the recursive resolution process as a last step (⑥ and ⑦ in Fig. 4.1), the subdomain translation.

System Specification. We built an authoritative name server on a commercial public cloud service, Vultr [68]. The cloud instance has two 2.6 GHz virtual CPUs, with 4 GB of memory, and running CentOS 7 x64. BIND 9.9.4 was used, and the resolution of IPv6 was disabled in our configuration, since this work only focused on the IPv4 address space.

Second-Level Domain. We purchased an SLD, *ucfsealresearch.net*, from GoDaddy [19] to enable the configured authoritative name server to manage the domain name resolution of its subdomains. GoDaddy provided the option of changing responsible DNS server for the purchased SLD. We set the authoritative name server that we configured as the responsible DNS server of the purchased SLD.

4.3.4 Subdomain Generation

To understand the behavior of the open resolver, we need to keep track of `Q1, Q2, R1`, and `R2` for each open resolver. Basically, DNS matches the pair of the query and the response using the ID field in the DNS header. However, it is infeasible to assign a unique ID number to each query-response pair in our measurement. This is because the DNS ID field is only 16 bits which can represent up to 65,535 IDs, while the probing rate is about 100k pps. Therefore, we implemented and applied a subdomain generation method to deal with this issue. During the probing process, DNS queries with different subdomains (*e.g.or000.0000000.ucfsealresearch.net*, *or000.0000001.ucfsealresearch.net, etc..)* are sent to different IP addresses. Using the `qname` information contained both in the DNS request and response, we were able to easily group `Q1, Q2, R1`, and `R2` for each flow.

Subdomain Cluster. Considering the limited memory resource of the authoritative name server, it cannot load about 4 billion subdomains for all IP addresses at once. In our authoritative name server, only about 5 million subdomains could be reliably loaded. Therefore, we devised a two-tiered subdomain structure for the measurement as shown in Fig. 4.3.

We grouped the 5 million subdomains that can be provided at once by the authoritative name server into one cluster. Five million subdomains, where each has a unique number (right 7 digits in the figure), are generated as one cluster (a zone file), and each cluster is numbered (left 3 digits in the figure). Once the predetermined number of subdomains in the cluster is exhausted, the cluster is updated with a new cluster number.

Subdomain Reuse. The application of subdomain and clustering allow us to easily match `Q1, Q2, R1`, and `R2` by comparing the `qname` field in DNS packet as well as to prevent the cached response. However, creating a cluster of 5 million subdomains also increases the probing time. To be specific, it takes about one minute to load 5 million subdomains at the authoritative name server, and the time will be very long considering that 4 billion IP addresses can make up to 800

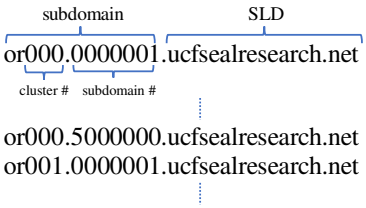


Figure 4.3: The subdomain structure for open resolver probing.

clusters in total. Moreover, the number of open resolvers found in other projects [43, 55] was less than 10 million, which means that about 99.75% of the loaded subdomains would not be used by the open resolver. As such, we added a subdomain reuse method to improve the performance. The prober parses the response packet (`R2`) after sending the packets including subdomains within one cluster and reuses the subdomain not in the collected `R2` set, indicating that the packet was sent to the IP address which is not an open resolver. Using this approach, we could reduce the number of clusters for probing from the theoretical value of 800 to only 4.

4.4 Results and Analysis

We successfully performed an Internet-wide probing that lasted approximately 10 hours and 35 minutes, where about 3.7 billion `Q1`, 13 million of each `Q2` and `R1`, and 6.5 million `R2` packets were captured at either the prober or the authoritative name server. Compared to the number of `Q1`, the number of `Q2` and `R1` is about 0.353% and those of `R2` are only about 0.176%. Table 4.1 shows a summary of the probing results.

We compare this result with results obtained from a dataset collected in 2013. In 2013, we performed an Internet-wide measurement using a C-based system, not based on ZMap as in this study, which does not affect the settings. As shown in Table 4.1, the probing took about 7 days for sending about 3.7 billion `Q1`. We collected about 38 million `Q2` and `R1`, and about 16.6 million `R2` packets. The percentages of `Q2` (`R1`) and `R2` to the number of `Q1` are about 1.0357 and 0.453,

Table 4.1: The Summary of the open resolver probing. Notice that the numbers in parentheses in the Q2 and R2 show the percentage of each number to the number of Q1.

Start time	End time	Duration	Q1	Q2, R1 (%)	R2 (%)
10/28/2013 2PM	11/04/2013 6PM	7d 5h	3,676,724,690	38,079,578 (1.0357)	16,660,123 (0.453)
04/26/2018 3PM	04/27/2018 2AM	11h	3,702,258,432	13,049,863 (0.3525)	6,506,258 (0.1757)

respectively. By observing the reduction of Q1 and R2 counts, we deduce that the number of open resolvers has declined over five years. In the following, we explore the change in the number of open resolvers and their behaviors in-depth.

R2 with Empty Question Field. In the sequel, we focus on R2 to analyze the behavior of the open resolvers. However, we remark that some of the collected R2 packets had an empty dns_question field. In general, the dns_question field is included in both the DNS query and the response [37]. As described in section 4.3.4, dns_question is used to group the set of Q1, Q2, R1, and R2. Accordingly, we excluded those 494 packets without dns_question field from our analysis in 2018. As a result, the following analysis only covered 6,505,764 R2 packets with dns_question field.

4.4.1 DNS Answer and Correctness

In this section, we describe a high-level analysis of the collected R2 packets, both presence and correctness. The presence of packets is simply measured by counting the number of R2 at the prober, while the correctness is measured by comparing the translated result in R2 with the ground truth (the IP address we set on the authoritative name server).

As shown in Table 4.2, we observed 16,660,123 R2 packets during the probing in 2013. Out of all R2 responses, 4,867,241 responses do not include dns_answer, while 11,792,882 packets contain dns_answer. Among the R2 packets which have dns_answer field, 11,671,589 packets indicate the correct IP address, but 121,293 responses include incorrect information. The rate of

Table 4.2: The presence and correctness of `dns_answer` field in `R2`. Notice that W and W/O correspond to the the number of `R2` packets with and without `dns_answer`, respectively. W_{Corr} and W_{Incorr} correspond to the number of correct and incorrect answers, which results in $W_{Corr} + W_{Incorr} = W$, and Err means the percentage of incorrect answers to the W, such that $Err = W_{Incorr}/W \times 100$.

Year	`R2`	W/O	W		$Err(\%)$
			W_{Corr}	W_{Incorr}	
2013	16,660,123	4,867,241	11,792,882		
			11,671,589	121,293	1.029
2018	6,506,258	3,642,109	2,863,655		
			2,752,562	111,093	3.879

incorrect information is about 1.029%.

In 2018, on the other hand, we found that 2,863,655 DNS responses out of total 6,505,764 `R2` collected packets had `dns_answer`, while the remaining 3,642,109 packets do not. Moreover, 2,752,562 of the 2,863,655 `dns_answer` fields contained the correct IP address result, and the other 111,093 responses had wrong results (3.879%).

From this result, we can conclude that the number of `R2` packets with `dns_answer` has greatly decreased from about 11.8 million to 2.9 million. The reduction (≈9 million) is similar to the reduction in the `R2` packets (≈10 million). Interestingly, however, the number of `R2` packets providing misleading information remains similar (≈110 thousand). Consequently, the error rate has increased from about 1% to 4%. From this result, we can infer that the number of resolvers exhibiting unusual behaviors did not significantly change, despite a significant reduction in the total number of open resolvers.

We conducted a deeper analysis of the answers from open resolvers. Specifically, we looked into how open resolvers behave according to the ideal way in section 4.4.2, and investigated the incorrect (suspicious) answers in section 4.4.3.

4.4.2 Analysis of DNS Header

The operation of DNS is mainly based on RFC1034 [36] and RFC1035 [37]. These documents elaborate the standards for DNS, such as DNS packet header structure as well as the process of DNS resolution. Considering that the open resolver is a component of the whole DNS, we expect it to follow the standard when participating in the translation process.

Through this measurement, however, we found that many resolvers don't follow the standard. To be specific, when the resolvers generate DNS answer packets, many of them fill the DNS flags and the response code fields not according to the instructions described in the standard. In the following, we investigate such behaviors of open resolvers by analyzing the collected data through a comprehensive measurement.

Recursion Available Flag. The `Recursion Desired (RD)` flag bit in the header of DNS queries sent during the probing is '1', which means that the recursive resolution is required. If the recipient of this query can perform recursive resolution (open resolver), it proceeds with recursion on behalf of the prober. Once the open resolver knows the result, it returns the translated IP address with the `Recursion Available (RA)` bit of '1' to the prober.

In this work, the investigation of the `RA` flag in `R2` started to figure out how many open resolvers exist. According to the definition of open resolver, a publicly accessible and recursion-available resolver, we expected the open resolvers to answer to our query with the `RA` bit of 1 and a correct answer. However, by looking into the collected data we found that `RA` bit does not directly mean the existence of an open resolver.

Table 4.3 shows the analysis of `RA` bit in `R2` packets. In 2013, we can see that the `RA` bit of 12,270,335 `R2` packets appeared as 1, which might imply that there were about 12 million open resolvers. The interesting observation we make is that there were 241,950 DNS responses with `dns_answer` field, even though they also have the `RA` bit of 0 (recursion unavailable). Moreover,

166,108 of them include the correct IP address information, which means that the senders of those 166,108 packets actually play the role of the open resolver, although they indicate they are not open resolvers. Conversely, there were 719,403 DNS responses without `dns_answer` field, but with the `RA` bit of 1 (recursion available), which means they do not perform the resolution. Such resolvers can be assumed to be either publicly inaccessible or unable to perform the recursive resolution. If the latter is the reason for the blank answer, it can be inferred that those resolvers do not follow the standard implementation for DNS resolution.

In the result collected in 2018, 3,503,581 `R2` packets have the `RA` bit of 0 (recursion unavailable), while 3,002,183 include `RA` bit of 1 (recursion available). Moreover, among the responses with `RA` bit of 0, 69,166 packets include `dns_answer`, 3,994 of correct answers and 65,172 of incorrect answers, which results in an error rate of 94%. On the other hand, 207,694 `R2` responses do not contain any resolved IP address, even though they claim to have recursion available.

Regardless of the value of the `RA` bit, the number of packets with the `dns_answer` field decreased to about one quarter (from 241 thousand to 69 thousand for `RA` bit of 0 and from 11.5 million to 2.8 million for `RA` bit of 1). However, the number of incorrect answers is similar, or even larger in 2018 (from 75 thousand to 65 thousand for `RA` bit of 0 and from 42 thousand to 46 thousand for `RA` bit of 1).

In terms of the accuracy of the response, when the `RA` bit is 0 and the `dns_answer` field is included, the resolved IP address is often wrong in 2018, with 94.225% of wrong `dns_answer` fields. Moreover, 31.346% of responses with `RA` bit of 0 and `dns_answer` field include incorrect information in 2013. If the `RA` bit of 0 and `dns_answer` were given together, the probability that the included IP address was inaccurate increased by more than three times. When the `RA` bit is 1, it can be seen that about 0.393% and 1.643% of the packets containing `dns_answer` include the wrong result in 2013 and 2018, respectively. Considering that only less than 6% of the cases with the `RA` bit of 0 include `dns_answer` field, the ratio of incorrect responses to the total would be low. Obviously, however, an improper combination of the `RA` bit and `dns_answer` can be a clear

Table 4.3: The statistics of the dns_answer field and the value of RA bit in R2. Notice that RA_0 and RA_1 correspond to the value of RA flag bit.

	2013					2018				
		W					W			
	W/O	W_{Corr}	W_{Incorr}	Total	Err(%)	W/O	W_{Corr}	W_{Incorr}	Total	Err(%)
RA_0	4,147,838	241,950 166,108	75,842	4,389,788	31.346	3,434,415	69,166 3,994	65,172	3,503,581	94.225
RA_1	719,403	11,550,932 11,505,481	45,451	12,270,335	0.393	207,694	2,794,489 2,748,568	45,921	3,002,183	1.643

indicator of a false result.

While investigating the RA bits in the responses, it is difficult to determine the number of open resolvers. It can be estimated that there were about 11.5 million open resolvers with the strictest criteria in 2013 (with the RA flag of 1 and correct dns_answer). Using the same criteria, it is estimated that there are about 2.74 million open resolvers in 2018. However, if we only use the RA flag as a criterion, we can also estimate that there were 12.2 and 3 million open resolvers in 2013 and 2018, respectively. On the other hand, it is also possible to conclude that there were about 11.7 and 2.75 million open resolvers by only counting the R2 packets with correct answer regardless of the RA bit.

Authoritative Answer Flag. Authoritative Answer (AA) means that the server responding to the DNS query is the authoritative name server for the given domain. In our probing, we sent DNS queries for all IPv4 addresses, and we were the only owner for the SLD, *ucfsealresearch.net*. Therefore, intuitively, it is appropriate to assume that the AA bit of all R2 is set to 0 except one response from our authoritative name server itself. However, as shown in Table 4.4, 381,124 R2 packets came back with the AA bit of 1, which is about 2.29% of all responses in 2013. Among them, 231,368 responses contained dns_answer, of which 78,279 had incorrect results. The ratio of the false answers to total answers with the AA bit of 1 is about 20.539%, which is significantly high compared to 0.372% when the AA is 0.

In 2018, we received 249,193 R2 responses with AA of 1. Among them, 119,147 packets had

Table 4.4: The statistics of the dns_answer field and the value of AA bit in R2. Notice that AA_0 and AA_1 correspond to the value of AA flag bit.

	2013					2018				
		W					W			
	W/O	W_{Corr}	W_{Incorr}	Total	Err(%)	W/O	W_{Corr}	W_{Incorr}	Total	Err(%)
AA_0	4,717,485	11,561,514 11,518,500	43,014	16,278,999	0.372	3,512,053	2,744,518 2,727,477	17,041	6,256,571	0.621
AA_1	149,756	231,368 153,089	78,279	381,124	20.539	130,046	119,147 25,095	94,052	249,193	78.938

dns_answer and 94,052 had incorrect information (79%), which is more than twice the rate of 2013, while the rate for AA bit of 0 is about 0.621%. The number of responses with AA bit of 1 is less than 4% of the total answers, while incorrect answer with AA bit of 1 account for about 84.661% of all incorrect packets.

Comparing the results of 2013 and 2018, we can see that the R2 with AA bit of 0 is greatly reduced (from 16 million to 6 million). In the case of AA bit of 1, the number of packets decreased from about 381k to 249k, which is about 61%. Nevertheless, the value of 249k is still abnormally higher than the expected value of 1. As interesting as the findings of the RA bit, however, the number of responses with inaccurate information was similar or even higher, which results in a significantly high error rate in 2018, which more than doubled from 2013.

As in the previous analysis of the RA flag, the analysis of the AA flag also shows that a large number of open resolvers do not work in a reliable manner.

Response Code. The response code (rcode) of the DNS response provides metadata about the outcome of the resolution. The rcode, which usually has a value from 0 to 15, is set to 0 for NoError, 1 for FormErr, 2 for ServFail, 3 for NXDomain, 4 for NotImp, 5 for Refused, 6 for YXDomain, 7 for YXRRSet, 8 for NXRRSet, and 9 for NotAuth [17]. All but 0 (NoError) indicate that the resolution was not successful.

Table 4.5 shows the distribution of rcode in the collected R2. As expected, most responses containing dns_answer field contained an rcode of 0, while most responses had a nonzero rcode without

Table 4.5: The rcode of the DNS reponses. Notice that each column corresponds to rcode of 0 to 7, and 9. The rcode of 8 (NXRRSet) is omitted due to the absence in our dataset.

		NoError	FormErr	ServFail	NXDomain	NotImp	Refused	YXDomain	YXRRSet	Not Auth
2013	*W*	11,780,575	0	12,723	10	0	1,272	0	0	0
	W/O	1,198,772	453	354,176	145,724	38	3,168,053	0	2	11
	Total	12,979,347	453	366,899	145,734	38	3,169,325	0	2	11
2018	*W*	2,860,940	23	2,489	10	0	193	0	0	0
	W/O	377,803	233	200,320	48,830	605	2,934,269	1	2	80,032
	Total	3,238,743	256	202,809	48,840	605	2,934,462	1	2	80,032

`dns_answer`. Except for this general tendency, however, we found some `R2` packets with abnormal combinations: 14,005 contained a nonzero (error) rcode despite having `dns_answer` field; 12,723 for ServFail, 10 for NXDomain, and 1,272 for Refused. Conversely, 1,198,772 `R2` packets without the `dns_answer` field had rcode of NoError.

In 2018, we also found 2,715 `R2` packets with a nonzero rcode, even with `dns_answer` field. In particular, 23 `R2` s have rcode of 1, 2,489 have 2, 10 have 3, and 193 have 5; 377,803 responses with rcode of 0 had no `dns_answer` field.

In analyzing response code, we found that the number of packets with most response codes decreased (NoError, FormErr, ServFail, NXDomain, and Refused). However, we also can see that the number of responses with the rcode of 1 (NotImp) and 9 (Not Auth) significantly increased, while those with the rcode of 6 (YXDomain) and 7 (YXRRSet) remained at a similar level.

4.4.3 Incorrect DNS Answers

Here we describe further analysis on the incorrect IP addresses included in `R2` packets based on the result observed in 2013 and 2018. As shown in Table 4.2, we notice that the wrong answer was provided in 110,093 packets out of 6,506,258 `R2` packets in 2018, while 121,293 packets out of 16,660,123 provided the wrond answer in 2013.

Table 4.6 shows a summary of the incorrect answers collected through the measurement. We

Table 4.6: The Summary of incorrect answers. Notice that $\#_{R2}$ means the number of R2 packets that include the answer in each form, and $\#_u$ means the number of unique values appearing in $\#_{R2}$.

Form	2013		2018		Example
	$\#_{R2}$	$\#_u$	$\#_{R2}$	$\#_u$	
IP	112,270	28,443	110,790	15,022	216.194.64.193
URL	249	175	231	80	u.dcoin.co
string	10	57	72	29	wild, OK, ff
N/A	8,764	-	-	-	<0x00>
Total	121,293	28,675	111,093	15,131	-

categorized 121,293 and 111,093 R2 packets in 2013 and 2018 with the wrong result into three types: IP address, URL, string, according to dns_answer. As a result, we found that 112,270 in 2013 and 110,790 in 2018 of the R2 packets had incorrect IP addresses, while 249 and 231 R2 packets had incorrect URLs. We also found that 10 and 72 responses include the abnormal strings such as *wild, ff, OK, 04b400000000, etc.*

Top 10 Analysis.

Table 4.7 shows the top 10 IP addresses with the most occurrences in R2 packets in 2018. The most frequently observed IP address in incorrect dns_answer was found in 23,692 responses, which is a domain and web hosting related company. The summed number of top 10 appearances is 50,669, which is about half the total number of incorrect R2 responses (111,093).

By examining the top 10 addresses, we found that four of them are private networks that belonged to 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16. However, in the case of the IP addresses, we found that 74.220.199.15, 208.91.197.91, and 141.8.225.68 are located at the second, third, and fourth places in the table, with suspicious information was found from security information vendor. For IP address 208.91.197.91, for instance, Ransomware Tracker [66] states that the address is a ransomware IP, and Cymon [12] shows that malware, phishing, and botnet activities are reported for the given address as shown in Fig. 4.4. Therefore, 22,805 R2 packets pointing to the IP addresses can be considered to have a deceptive dns_answer for malicious purpose.

Table 4.7: Top 10 IP addresses included in incorrect DNS responses in 2018. 'Reports' is whether a suspicious report was found when querying the address using Cymon API.

IP address	#	Org Name	Reports
216.194.64.193	23,692	Tera-byte Dot Com	N
74.220.199.15	13,369	Unified Layer	**Y**
208.91.197.91	8,239	Confluence Network Inc	**Y**
141.8.225.68	1,197	Rook Media GmbH	**Y**
192.168.1.1	1,014	private network	N/A
192.168.2.1	741	private network	N/A
114.44.34.86	734	Chunghwa Telecom	N
172.30.1.254	607	private network	N/A
10.0.0.1	548	private network	N/A
118.166.1.6	528	Chunghwa Telecom	N
Total	50,669	-	-

For the 2013 dataset, we inspected the top-10 frequent IP addresses. The total number of `R2` packets that include those addresses is 26,514, which is almost half of the number in 2018. Specifically, in 2013, the most frequently appeared address with 9,651 `R2` packets is 74.220.199.15, the second rank in 2018, and it is the only address reported as malicious. Moreover, there are 3 private network addresses, 192.168.1.254, 192.168.2.1, and 192.168.1.1 as a second, third, and tenth places. More than 5k packets, in third place, include the address 20.20.20.20, which is owned by Microsoft, while 173.192.59.63 appeared in 995 packets (seventh rank), 221.238.203.46 in 811 packets (eighth rank), and 68.87.91.199 in 748 packets (ninth rank). As for the unusual point, 1,032 packets include 0.0.0.0.

Suspicious IP Addresses. Based on the possibility of malicious activities performed by open resolvers, we conducted a deeper analysis to identify the open resolvers misleading users to malicious destination. For answers of IP addresses in Table 4.6, we conducted an additional analysis using Cymon API [6]. From Cymon, we gathered reported information about the given addresses and judged their maliciousness. As a result, we found that there were 335 IP addresses reported as malicious. When there are multiple reports for different categories, the most frequently reported category is selected.

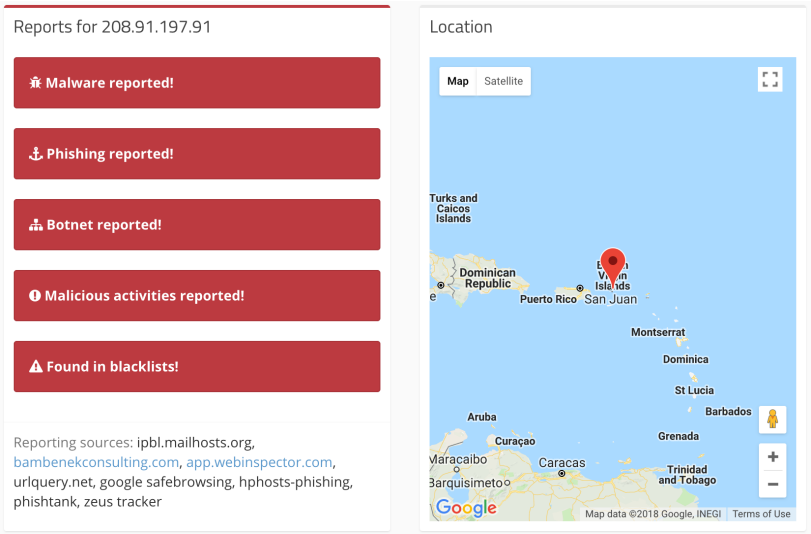


Figure 4.4: The information in Cymon about the IP address of 208.91.197.91 that ranks in the third highest reference in 2018. Note the multiple reports about malware, botnet, phishing, etc. It can be assumed that the open resolvers which redirect the users to this address are exploited by the adversaries.

As shown at the right side of Table 4.8, the most common category for the malicious IP address is malware. The number of IP addresses related to malware is 170, accounting for over half. Moreover, the number of IP addresses related to phishing is 125, accounting for more than one third, alluding to the possibilities of DNS poisoning or manipulation. Moreover, the number of IP addresses reported as spam, SSH bruteforce, scan, and botnet is about 40. When the analysis is conducted w.r.t. the number of `R2` packets, the result is different. In `R2` packets, malware addresses account for more than 85% of the total, which means that 170 malware reported addresses are observed in `R2`, on average 136 times each. On the other hand, the 125 phishing related IP addresses are observed in 2,878 `R2` packets($\approx$10% of the total; 23 occurrences for each address).

To measure changes in the malicious use of open resolvers, we also conducted the same analysis on the result in 2013. In total, there were 100 unique malicious IP addresses in 12,874 responses. Among them, 65 addresses appearing in 11,149 `R2` packets were reported as malware. For addresses reported as phishing, there were 18 unique addresses that were included in 1,092 responses.

Table 4.8: Malicious IP addresses in R2 packets. Notice that $\#_{IP}$ corresponds to the number of IP addresses reported to Cymon in each category. When the IP address is reported with multiple categories, the category with the most frequency is selected. Notice that $\#_{R2}$ means the number of R2 packets that include the IP addresses belonged to each category.

Report Category	2013				2018			
	$\#_{IP}$	$(\%_{IP})$	$\#_{R2}$	$(\%_{R2})$	$\#_{IP}$	$(\%_{IP})$	$\#_{R2}$	$(\%_{R2})$
Malware	65	65.0	11,149	86.6	170	50.7	23,189	86.1
Phishing	19	19.0	1,092	8.5	125	37.3	2,878	10.7
Spam	4	4.0	67	0.5	15	4.5	44	0.2
SSH Bruteforce	2	2.0	2	0	10	3.0	323	1.2
Scan	8	8.0	493	3.8	9	2.7	388	1.4
Botnet	1	1.0	70	0.5	4	1.2	102	0.4
Email Bruteforce	1	1.0	1	0	2	0.6	2	0
Total	100	-	12,874	-	335	-	26,926	-

In addition to the above two categories, 16 IP addresses in 633 responses were reported as Spam, SSH Bruteforce, Scan, Botnet, and Email Bruteforce.

The interesting observation we make by comparing the results of 2013 and 2018 is that the malicious behavior of open resolvers has increased from 12,874 to 26,926 in terms of the number of R2 responses. This corresponds to more than 100% of increase. From the point of view of the unique IP addresses in the R2 packets, the increase in malicious behavior is also significant: from 100 unique addresses to 335 addresses, which is more than tripled (235%).

We notice that the number of unique IP addresses reported as malware increased from 65 to 170, but the ratio to all malicious addresses decreased from 65% to 50%. The most rapid change can be found in phishing: from 19 in 2013 to 125 in 2018, which is about seven folds increase. The ratio is also doubled from about 19% to 37%, indicating that today's open resolvers are more exploitable for phishing purposes than before.

Our analysis can be considered as a lower bound of the malicious activities in that it deals only with information in Cymon. However, more malicious addresses can appear when validating using threat information from multiple vendors.

DNS Manipulation. The above analysis shows that DNS manipulation happens. Queries sent to each IP address were a subdomain instantaneously created, and subsequently manipulated. As mentioned earlier, one of the purposes of using subdomain is to prevent caching of results at the open resolver. In other words, the malicious IP address in the `R2` packets we received does not match the information stored in the cache of the open resolver, but it is likely to be the result of an actual but illegitimate response. It is unreasonable to assume that an attacker applies a cache poisoning to the legitimate open resolver, because of the short time window, but it is more plausible to say that the open resolver itself is under the adversary's control. It can be assumed that those open resolvers will work in a way that provides the predetermined answer which includes the malicious IP address for every query they receive.

4.4.4 DNS Header in Malicious Responses

In addition to the general analysis, we also provide an analysis of `R2` packets that may mislead the users to malicious IP addresses.

RA and AA Flags. Table 4.9 shows the statistics of `RA` and `AA` flags in `R2` packets that contain a malicious IP address. With regard to the `RA` bit, more than 70% of `R2` packets indicate that the senders are recursion unavailable although the responses have the `dns_answer` field. On the other hand, about 27% of `R2` packets include the `RA` bit of 1, which means that the contained `dns_answer` fields are the result from recursive resolution. However, we already know that the IP addresses in those `R2` packets are malicious and not true, which allows us to infer that `RA` bit is used improperly.

We also make several interesting observations from the `AA` bit in `R2` packets. More than 70% of the responses have a `AA` bit of 1, which means that they are from the authoritative name server. Considering that they were not directly sent to our authoritative name server, and even they included the malicious IP address and not true result, the use of `AA` flag can be assumed to be a malicious

Table 4.9: RA and AA analysis on R2 packets with the malicious IP address in 2018. Notice that $\#_R$ and $\#_A$ correspond to the number of packets with each flag and value. Also, $\%_R$ and $\%_A$ correspond to the percentage of each flag to the total R2 packets including the malicious information (26,926).

RA	$\#_R$	$\%_R$	AA	$\#_A$	$\%_A$
RA_0	19,534	72.5	AA_0	7,472	27.8
RA_1	7,392	27.5	AA_1	19,454	72.2

attempt to allude to the credibility of the response.

Response Code. In the analysis of rcode, we found that all 26,926 R2 packets with malicious IP address have the rcode of 0 (NoError). The use of rcode can also be seen as an intention to encourage the requester to trust the response and to access the IP address by claiming a reliability of the answer.

4.5 Analysis of Packets at Authoritative Server

In this section, we analyze the Q2 and R1 packets captured at the authoritative name server in 2018. By incorporating the analysis of the packets at the authoritative name server and prober, we characterize open resolvers behaviors in detail.

4.5.1 Q2 and R1 Analysis

As in Table 4.1, we captured about 13 million Q2 and R1 packets at the authoritative name server. Only about half of them were delivered to our prober back as R2 responses. Thus, we do further exploration to understand the rest of the packets.

Summary. First, we describe a summary of Q2 and R1 packets collected at the authoritative name server. As we described earlier, Q2 and R1 packets at the authoritative name server were captured using tcpdump as pcap files. In order to analyze them in detail, we implemented a C-based parser

Table 4.10: Examples of duplicated queries in `R1` packets from different IP addresses. The subdomain (or000.2543237) was originally sent to and returned by 5.188.178.199.

Subdomain	IP (Block)	#	Organization
or000.2543237	74.125.0.0/16	72	Google LLC.
	173.194.97.0/25	14	Google LLC.
	162.209.124.87	1	Rackspace Hosting
	200.32.248.1	1	Belize Telemedia LTD.
	200.32.218.132	1	Belize Telemedia LTD.

using libpcap and looked into the collected DNS packets. Since our main goal is to understand how the prober's query for subdomains would be handled by open resolvers and the authoritative name server, we excluded 'non-A' type (e.g., NS, CNAME, etc.) packets from about 13 million collected packets. As a result, we are left with 10,638,510 `R1` packets as our dataset. In the meanwhile, we observed that the number of packets in `Q2` and `R1` differs slightly, but this is probably due to missing packets by tcpdump or duplicated queries, which will discuss further below. In essence, the authoritative name server is under our control, so we safely assume no malicious behavior by the authoritative, such as packet tampering. Hereafter, we focus only on `R1` packets in the following analysis.

Duplicated Queries. The next step is to check the duplicated queries. As we explained in section 4.3.4, for probing we generated a unique subdomain for each target address, which means no DNS queries share the same `dns_question` field. However, in our dataset, we found that there exists such a significant number of duplicates. For example, the authoritative name server received the queries about one subdomain, or000.2543237.ucfsealresearch.net, from 89 different IP addresses. As shown in Table 4.10, 72 addresses belonged to 74.125.0.0/16 and 14 addresses belonged to 173.194.97.0/25, while both subnets are owned by Google LLC. On the other hand, there are three `R1` packets from addresses not owned by Google.

An interesting observation we make is that the `Q1` for this subdomain was originally sent to 5.188.178.199, which is owned by Fast Content Delivery LTD. This address does not appear to

perform recursive resolution directly for the query we sent, but it tries to utilize Google's or another organization's DNS server as forwarders to translate the given subdomain. However, what is even more interesting is that although the given subdomain was translated by our authoritative name server along with the other DNS server, such as Google, and the result was highly likely to be returned to the original resolver, the resolver did not deliver the correct answer to the prober. The `R2` response from 5.188.178.199 does not include an `dns_answer` field in the packet but includes a rcode of 5 (Refused). In summary, the DNS server refused the user's (prober's) query but performed resolution through other DNS servers (forwarders) for unknown reasons. Analysis of whether a recursive resolution is actually performed and the accuracy of `R2` packets is discussed in detail later.

Overall, we investigated the subdomains contained in 10,638,510 `R1` packets and we could see that `R1` packets for 35,320 subdomains were returned to two or more different resolvers or forwarders. Moreover, there are 6,423,321 unique domain names, while 4,215,189 subdomains are included in two or more queries.

Duplicated IP Addresses (Forwarder). While examining the `R1` packets, we observe that there is a relatively small number of unique IP addresses. Namely, we found that, while we had about 10.6 million `R1` packets, the number of unique IP addresses in our dataset is only 133,401. Compared to the 6,505,764 `R2` responses all having different IP addresses, this finding in itself is very interesting and highlights that the number of resolvers that directly sent `Q2` to our authoritative name server is very small.

We further analyzed those 133,401 unique addresses and found that 81,441 addresses appeared in only one `R1` query, which means that they performed recursive resolution only once. In other words, DNS queries for the rest of 10,557,069 `R1` packets were sent to the remaining 51,960 IP addresses. Again, considering that the 6.5 million `R2` all have different addresses, a resolution process of about 6.42 million was actually performed by DNS servers that sent more than one `Q2`.

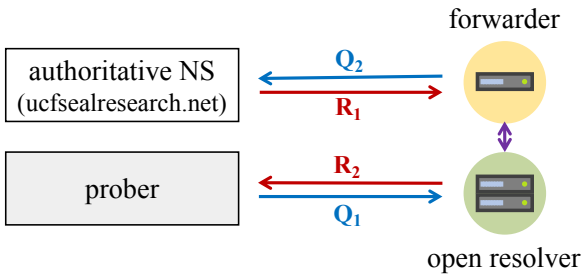


Figure 4.5: The flow of DNS request and response packets with a forwarder in the resolution. Unlike Fig. 4.2, a recursive resolution is processed by the forwarder. Our analysis indicates that those forwarders play an essential role in explaining a large number of incorrect or malicious answers.

From this analysis, we can infer the existence of forwarders that do not appear in Fig. 4.2. As shown in Fig. 4.5, a forwarder receives the DNS query from another resolver and performs recursive resolution instead. The existence of such forwarders can suggest a new interesting point when analyzing the accuracy and maliciousness of the `dns_answer` fields. This is because the wrong `dns_answer` in `R1` can be manipulated by the forwarder before sending it to the resolver, and further be delivered to the end-users. We will investigate deeper in the following section where the manipulation of the answer occurs.

Table 4.11 shows the top 5 forwarders which sent the largest number of `Q2` packets to the authoritative name server. Interestingly, three of the top five addresses are owned by an Algerie company, namely Algeria Telecom. The total number of `Q2` sent from these top five addresses totaled over half a million.

4.5.2 Comparing `R1` and `R2`

We also reexamined the accuracy of `R2` responses. Instead of simply verifying the resolved IP address in the `R2` responses as in section 4.4.1, we observed the behavior of open resolvers in conjunction with the collected `R1` packets.

Table 4.11: The top 5 forwarders' IP addresses sending the largest number of Q2 packets. Notice that # means the number of Q2 packets sent from an IP address.

IP address	#	Org Name (Country)
41.110.32.2	148,718	Algeria Telecom (Algerie)
41.110.30.2	121,698	Algeria Telecom (Algerie)
156.200.40.50	93,368	Te Data (Egypt)
41.110.31.2	69,729	Algeria Telecom (Algerie)
200.75.51.133	69,501	ETB (Colombia)
Total	503,014	-

4.5.2.1 Direct Resolution (Without Forwarder)

As we discussed above, in many recursive resolution processes, a forwarder was included to deliver the DNS query and response between the prober and the authoritative name server. As such, our first concern is the number of open resolvers that actually perform recursive resolution without forwarders. To figure them out, we examined the number of resolvers' IP addresses that appeared in the {R1, R2} pair. As a result, we found 70,694 open resolvers (about 1.09% of collected R2 packets) that performed direct recursive resolution without employing a forwarder. This number is less than the 81,441 addresses that only sent a query for one subdomain, discussed in the previous section. In other words, among the 81,441 IP addresses that sent only one query, the number of open resolvers that performed a direct resolution is 70,694. The remaining 10,747 IP addresses can be considered as forwarders that participate in only one DNS resolution. It can also be inferred that those forwarders are only used in a few limited cases, and are not available publicly.

4.5.2.2 Subdomain based Analysis and Correctness

Next, we examined the correctness of the resolution results by looking into each {R1, R2} pair. The reason for this analysis is to further explore the misbehavior and malfunction of the forwarders and resolvers, such as cases that result in empty dns_answer, packet drop, manipulation, etc..

Table 4.12: The number of packets in R1, R2, their intersection, and differences. Notice that R1-R2 means that the number of packets included in the R1 dataset but not in the R2. Conversely, R2-R1 means the number of packets only in R2.

	#	R1 ∩ R2	R1- R2	R2- R1
R1	10,638,510	3,215,947	5,546,983	3,289,817
R2	6,505,764			

In general, the process of domain name resolution by the open resolvers follows Fig. 4.1 (regardless of the existence of the forwarder in Fig. 4.2 and Fig. 4.5). As a result, if the translation of the subdomain is done successfully, the packet flow must appear in both R1 (Q2) and R2 at the same time. In particular, in our experiments, we used different subdomains to eliminate the effect of caching on our visibility into the resolution from the open resolvers. As such, we can assume that all resolutions will follow the flow. However, we observed two abnormal cases: 1) packets that exist in R1 but not in R2 and 2) packets that exist in R2 but not in R1. Along with these two cases, we also included a general resolution process that goes through both R1 and R2. Notice that we conducted our analysis based on the subdomain, not the IP address, which means the existence of the same IP address (in case of using a forwarder) does not affect our end results. Table 4.12 shows a summary of the statistics of R1 and R2.

Packets in R1 but not in R2. The first case we considered is packets that exist in R1 but not in R2. This case implies that an authoritative name server received the query and replied back and that the resolution result is discarded or dropped in the middle. We observed that 5,546,983 out of 10,638,510 subdomains in R1 are not seen in R2. In this case, since resolvers did not provide the resolved address to the prober, we cannot verify its accuracy. However, it can be seen that a large number of resolvers exhibit abnormal behavior. Notice that there is also the possibility of an unreliable network, which may result in dropping some packets. However, the large number of dropped packets (more than 50%) alludes to the former explanation, where the dropped packets due to the network condition are often very small.

Packets in R2 but not in R1. The second case is packets that exist in R2 but are not in R1. In this case, it seems weirder than the previous case because it provides the answer without going through the normal resolution process. This case may occur when a resolver returns a predetermined result without recursive resolution. We found that more than half (i.e., 3,289,817) of the total R2 packets (i.e., 6,505,764) fall into this case. However, among the 3,289,817 R2 packets, 95.27% of them (i.e., 3,134,036) do not have an dns_answer field, which happens when the DNS queries are responded to without contacting the authoritative name server. This number also corresponds to 86.05% of all R2 responses without dns_answer fields in Table 4.2.

On the other hand, there are 155,781 R2 packets which include dns_answer fields although they did not perform resolution. Among them, 95,840 R2 responses included incorrect answers, which happens when the resolver injects the predetermined (or random) result. Besides, we also found that 59,941 R2 packets that contain correct responses, while those packets are not in R1. The potential reasons behind this are: 1) missing packets on tcpdump and 2) different caching schemes at the resolvers (e.g., SLD-based caching). As future work, we will further explore this case of packets and resolvers.

Packets in both R1 and R2. The third case is the most general case, which is a common recursive resolution process. In other words, the conversion is done via the authoritative name server by the open resolver, and the result is returned back to the prober. In this case, it is expected that the correct resolved address would be included in an dns_answer in the R2 packets. However, we observed that 508,082 R2 packets (about 15.8% of the subdomains) included in both R1 and R2, were returned to the prober without dns_answer fields. For example, the subdomain (or000.2543237) introduced in Table 4.10 represents this case. In addition, 2,707,865 R2 packets which are about 84.2% of this case, were returned with a resolved address, of which 2,692,621 packets delivered the correct answer and 15,244 packets contained the incorrect answer address.

4.5.2.3 *Malicious Answers*

We also conducted an additional analysis of the malicious answers in Table 4.8. This analysis is for figuring out how many of the 26,926 packets that led us to a malicious destination were actually resolved with the authoritative name server and how many of them used forwarders. To do this, we listed the subdomains that have been translated into malicious addresses and looked them up in the `R1` dataset.

As a result, we found that 23,349 subdomains out of 26,926 (about 86.72%) in the `R2` packets do not appear in the `R1` dataset. This finding means that some resolvers did not perform recursive resolution for the domain name queried by the prober, which also means that their responses are highly likely to be predetermined. This includes the possibility that the targeted resolver sent the query to the forwarder, but the forwarder provided a predetermined answer without contacting the authoritative name server. However, since we could not look into the communication between the targeted resolver and forwarder, it is difficult to verify this theory, although very plausible.

The second is the case that open resolvers performed resolution through the authoritative name server, and 3,577 `R2` packets (about 13.28%) belonged to this case. Among them, 2,935 resolvers delivered queries to the authoritative name server via forwarders, while 642 sent queries directly to our server without forwarders. Given the high likelihood that a specific malicious destination is likely to be a predetermined value, it is uncertain why they sent a query to the authoritative name server to get a resolved address. However, we can at least see that the resolver itself manipulated the answers in 642 `R2` packets, while the answers in 2,935 `R2` packets could have been manipulated by either the resolver or the forwarder.

4.6 Discussion

The Need for Continuous Monitoring of Open Resolvers. As show in the above analysis, open

DNS resolvers still pose a threat to the Internet. The fact that the number of open resolvers has declined does not mean that their threat is going to go away anytime soon. For example, the number of open resolvers with a malicious behavior has increased, which is a clear example of the need for steady observation of those resolvers and the role they play in the DNS ecosystem.

However, and to the best of our knowledge, such a continuous and steady observation of the open resolvers on the Internet is not well performed. For example, one of the most popular open resolver-related projects is the `openresolverproject.org` [43], which shows the number of open resolvers distributed over the Internet and some flag values (`RA` bit or rcode). However, this project but does not provide any in-depth analysis of malicious IP addresses included in the responses. Moreover, and most importantly, *the project has been discontinued since January 2017*.

Another project, which is called the shadowserver dnsscan [55], provides daily information on the number and geographical distribution of open resolvers, but does not specifically analyze the behavior of each open resolver. Therefore, it is difficult to use the result of this project for understanding the threat of open resolvers in such relevant details. This is because the decrease in the number of open resolvers, as pointed by our work, does not directly mean that the associated threats are also reduced. Just as the number of open resolvers showing malicious behavior has increased, an in-depth analysis of their behavior is required for an accurate understanding of the role of open resolver on the Internet.

Censys [9] and Rapid7 [50] also provide weekly (censys) or monthly (rapid7) scan using ZMap. These raw scan datasets are more useful in that the DNS response packets can be inspected, but still have limitations. First, this raw dataset is from the measurement result only using prober, not the authoritative name server. As shown in Fig. 4.2, if the measurement is conducted only at the prober, we cannot catch the packet flow of `R1` and `Q2`, which makes it difficult to investigate the behavior of open resolvers in-depth. Moreover, because both repositories use ZMap as a scanning tool, these datasets may have a blind spot that ZMap has. For example, the current ZMap can miss packets in that it only stores results for the responses from the target port of the scan (e.g., DNS

responses only from port 53). This incomplete measurement can lead to the underestimation of the threat of misbehaving resolvers.

To this end, we believe a systematic and constant follow-up of the behavioral analysis in the open resolver ecosystem is a gap in the literature, and is needed for improving DNS security. For understanding the behavioral changes in open resolvers and finding countermeasures against the malicious activities such a steady observation is required.

Private Network in Incorrect Information. In Table 4.7, we show that four of the top 10 IP addresses with the incorrect `R2` responses in 2018 are addresses in private networks (196.168.1.1, 192.168.2.1, 172.30.1.254, and 10.0.0.1). Besides the Top 10, several private networks appeared in the incorrect responses as well.

We speculate multiple scenarios that could lead to such a behavior. For example, landing on such a private network may be a redirection to a webpage for the user's consent or form submission in a public network (e.g., Wi-Fi in airport). It is also likely to be a similar redirection in the responses with the particular company's IP address. However, in the case of a private network, and given the fact that our DNS query was sent from outside the network, this behavior is difficult to accurately understand. If it is a DNS server for users inside the network, it means that the connection is also allowed from the outside.

Open Resolver as an Existent Threat. In section 4.2.3, we described two threats that open resolvers can bring about: DNS amplification (DDoS attacks) and DNS manipulation. In our analysis, we found that there are millions of open resolvers still exist in the wild, which allows us to deduce that these resolvers can be exploited by adversaries for launching amplification attacks. The number of open resolvers around the world can be equated to the magnitude of the potential threat as it is a threat from the functional loophole of the open resolver (no verification method for spoofed source IP address is in place). The mere existence of open resolvers and the adversary's malice are a guarantee for an attack.

However, in terms of DNS manipulation, the existence of malicious open DNS resolvers may not directly correspond to an actual threat. This is due to the passive role of open resolvers in DNS resolution. A malicious open resolver can perform its (malicious) actions only when it receives a domain name resolution request. If no user queries the malicious open resolver, the manipulated DNS record is essentially meaningless. At this point, we need to see how malicious open resolvers are actually queried by legitimate users. Moreover, it would be further important research topic to investigate how malicious open resolvers attract legitimate users.

4.7 Summary

In this study, we conducted an up-to-date measurement of the distribution and behavior of open resolvers. Through an Internet-wide probing, we can see that about 3 million open resolvers still exist on the Internet and many of them operate in a way that deviates from the standard. From the result, we detected two threats posed by open resolvers.

First, the presence of millions of open resolver increases the threat of a DNS amplification DDoS attack. The adversary can exploit open resolvers as an amplifier by simply sending DNS 'ANY' queries to them, which results in the concentration of large DNS answers to the victim. Moreover, we also found evidence suggesting open resolvers' abnormal behaviors. The flag bits in the DNS response from open resolvers are often inappropriately marked. More than 69k open resolvers in 2018, for example, state that they are not recursion available (RA bit of 0) although they include the result of recursive resolution. More seriously, it is also shown that over 110k open resolvers provide the incorrect IP address as a DNS response, while more than 26k open resolvers return the IP addresses reported as malware, phishing, etc.. Furthermore, through the deeper analysis of the packets collected at the authoritative server, we demonstrate the use of forwarders in the open resolver ecosystem and the possibility that incorrect or malicious responses are a by-product of the involvement of these forwarders.

CHAPTER 5: CONCLUSION

As the importance of computers and the Internet grows, the security of critical infrastructure continues to be emphasized. On the other hand, there have been a series of attacks that attempt to undermine or further destroy the confidentiality, integrity, and availability of these critical infrastructures, which leads to new approaches to defend against emerging attacks. As such, attack techniques and security have been developing with mutual influence.

However, not all efforts to improve security always lead to meaningful results. In other words, when a systematic approach to security is made, including **proper metrics setup**, **accurate measurements**, and **multifaceted analysis**, the critical infrastructure on the Internet, such as the true information sharing system, web application services, endpoints, and domain name systems, can be protected. In this dissertation, we highlighted the importance of these key components through the examination of existing research and systems.

First, we questioned the quantitative evaluation metrics used in the existing threat information sharing system. Through the evaluation, we could see that the volume of contribution the participants made did not directly match to their actual level of contribution. When we applied qualitative metrics such as correctness, relevance, integrity, and utility, we could see results different from simple quantitative assessments, which means that quantity should not be the only metric in the evaluation. Based on our findings, we recommend, in the future, threat information sharing systems to explore alternatives that supplement the limitations of such existing quantitative assessments.

Secondly, we investigated the role of assumptions in measurement. Assumptions are necessary for any scientific measurement and experiment, but they always require careful consideration. By measuring the actual effectiveness of pulsing DDoS attacks in the realistic environment, we found that the attack may exist but its effect could have been slightly overestimated. As a result, this study emphasized that the establishment of accurate assumptions is a prerequisite to accurate mea-

surement results.

Finally, we conducted large-scale measurement and analysis of open resolvers. In this study, we incorporated in-depth behavioral analysis into the general statistical analysis. Through that comprehensive analysis, we were able to find empirical evidence of DDoS manipulation attack as well as to infer the threat magnitude of DDoS amplification DDoS attack by figuring out the number of open resolvers distributed around the world. This study allowed us to know that a new viewpoint in analysis could give new insights into the potential threats on the Internet. This is why researchers should make an effort to analyze a given dataset in a multi-pronged manner.

In sum, we could conclude that it is difficult to say that the above three key components are completely demonstrated even in the ever-evolving security technology. Of course, since the above concepts are without absolutely correct answers, it would be impossible that researchers and system administrators achieve perfection. On the other hand, however, that is a reason why they need to put more effort. To improve the security of critical infrastructure, and even the Internet itself, it will always be necessary to think deeper about their own system and research and to look at it from a new perspective.

APPENDIX A: UCF IRB LETTER

UNIVERSITY OF CENTRAL FLORIDA

Institutional Review Board
FWA00000351
IRB00001138, IRB00012110
Office of Research
12201 Research Parkway
Orlando, FL 32826-3246

Memorandum

To: Jeman Park

From: UCF Institutional Review Board (IRB)

CC: Barbara Fritzsche

Date: May 12, 2020

Re: Request for IRB Determination

The IRB reviewed the information related to your dissertation *Improving the Security of Critical Infrastructure: Metrics, Measurements, and Analysis.*

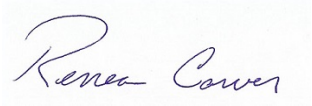
As you know, the IRB cannot provide an official determination letter for your research because it was not submitted into our electronic submission system.

However, if you had completed a Huron submission, the IRB could make one of the following research determinations: “Not Human Subjects Research,” “Exempt,” “Expedited” or “Full Board.

Based on the data points you provided, this study would have been issued a Not Human Subjects Research determination outcome letter had a request for a formal determination been submitted to the UCF IRB through Huron IRB system.

If you have any questions, please contact the UCF IRB irb@ucf.edu.

Sincerely,



Renea Carver
IRB Manager

APPENDIX B: COPYRIGHT INFORMATION

IEEE COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

QOI: ASSESSING PARTICIPATION IN THREAT INFORMATION SHARING
Jeman Park, Hisham Alasmary, Omar Al-Ibrahim, Charles Kamhoua, Kevin Kwiat, Laurent Njilla, Aziz Mohaisen
2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)

COPYRIGHT TRANSFER

The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the "IEEE") all rights under copyright that may exist in and to: (a) the Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

GENERAL TERMS

1. The undersigned represents that he/she has the power and authority to make and execute this form.
2. The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.
3. The undersigned agrees that publication with IEEE is subject to the policies and procedures of the IEEE PSPB Operations Manual.
4. In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall be null and void. In this case, IEEE will retain a copy of the manuscript for internal administrative/record-keeping purposes.
5. For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.
6. The author hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the author has obtained any necessary permissions. Where necessary, the author has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE

You have indicated that you DO wish to have video/audio recordings made of your conference presentation under terms and conditions set forth in "Consent and Release."

CONSENT AND RELEASE

1. In the event the author makes a presentation based upon the Work at a conference hosted or sponsored in whole or in part by the IEEE, the author, in consideration for his/her participation in the conference, hereby grants the IEEE the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive, in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the "Presentation"). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IEEE and live or recorded broadcast of the Presentation during or after the conference.
2. In connection with the permission granted in Section 1, the author hereby grants IEEE the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice and biographical information as part of the advertisement, distribution and sale of products incorporating the Work or Presentation, and releases IEEE from any claim based on right of privacy or publicity.

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Jeman Park	10-11-2017
Signature	**Date (dd-mm-yyyy)**

Information for Authors

AUTHOR RESPONSIBILITIES

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE PSPB Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE's publishing policies may be found at http://www.ieee.org/publications_standards/publications/rights/authorrightsresponsibilities.html Authors are advised especially of IEEE PSPB Operations Manual section 8.2.1.B12: "It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it." Authors are also advised of IEEE PSPB Operations Manual section 8.1.1B: "Statements and opinions given in work published by the IEEE are the expression of the authors."

RETAINED RIGHTS/TERMS AND CONDITIONS

- Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
- Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author's personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale.
- Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use.The IEEE Intellectual Property Rights office must handle all such third-party requests.
- Authors whose work was performed under a grant from a government funding agency are free to fulfill any deposit mandates from that funding agency.

AUTHOR ONLINE USE

- **Personal Servers**. Authors and/or their employers shall have the right to post the accepted version of IEEE-copyrighted articles on their own personal servers or the servers of their institutions or employers without permission from IEEE, provided that the posted version includes a prominently displayed IEEE copyright notice and, when published, a full citation to the original IEEE publication, including a link to the article abstract in IEEE Xplore. Authors shall not post the final, published versions of their papers.
- **Classroom or Internal Training Use.** An author is expressly permitted to post any portion of the accepted version of his/her own IEEE-copyrighted articles on the author's personal web site or the servers of the author's institution or company in connection with the author's teaching, training, or work responsibilities, provided that the appropriate copyright, credit, and reuse notices appear prominently with the posted material. Examples of permitted uses are lecture materials, course packs, e-reserves, conference presentations, or in-house training courses.
- **Electronic Preprints.** Before submitting an article to an IEEE publication, authors frequently post their manuscripts to their own web site, their employer's site, or to another server that invites constructive comment from colleagues. Upon submission of an article to IEEE, an author is required to transfer copyright in the article to IEEE, and the author must update any previously posted version of the article with a prominently displayed IEEE copyright notice. Upon publication of an article by the IEEE, the author must replace any previously posted electronic versions of the article with either (1) the full citation to the

IEEE work with a Digital Object Identifier (DOI) or link to the article abstract in IEEE Xplore, or (2) the accepted version only (not the IEEE-published version), including the IEEE copyright notice and full citation, with a link to the final, published article in IEEE Xplore.

Questions about the submission of the form or manuscript must be sent to the publication's editor.
Please direct all questions about IEEE copyright policy to:
IEEE Intellectual Property Rights Office, copyrights@ieee.org, +1-732-562-3966

IEEE COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

\u00A0Timing is Almost Everything: Realistic Evaluation of the Very Short Intermittent DDoS Attacks
Jeman Park, DaeHun Nyang, and Aziz Mohaisen
2018 16th Annual Conference on Privacy, Security and Trust (PST)

COPYRIGHT TRANSFER

The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the "IEEE") all rights under copyright that may exist in and to: (a) the Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

GENERAL TERMS

1. The undersigned represents that he/she has the power and authority to make and execute this form.
2. The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.
3. The undersigned agrees that publication with IEEE is subject to the policies and procedures of the IEEE PSPB Operations Manual.
4. In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall be null and void. In this case, IEEE will retain a copy of the manuscript for internal administrative/record-keeping purposes.
5. For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.
6. The author hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the author has obtained any necessary permissions. Where necessary, the author has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE

You have indicated that you DO wish to have video/audio recordings made of your conference presentation under terms and conditions set forth in "Consent and Release."

CONSENT AND RELEASE

1. In the event the author makes a presentation based upon the Work at a conference hosted or sponsored in whole or in part by the IEEE, the author, in consideration for his/her participation in the conference, hereby grants the IEEE the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive, in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the "Presentation"). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IEEE and live or recorded broadcast of the Presentation during or after the conference.
2. In connection with the permission granted in Section 1, the author hereby grants IEEE the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice and biographical information as part of the advertisement, distribution and sale of products incorporating the Work or Presentation, and releases IEEE from any claim based on right of privacy or publicity.

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Jeman Park	18-07-2018
Signature	**Date (dd-mm-yyyy)**

Information for Authors

AUTHOR RESPONSIBILITIES

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE PSPB Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE's publishing policies may be found at http://www.ieee.org/publications_standards/publications/rights/authorrightsresponsibilities.html Authors are advised especially of IEEE PSPB Operations Manual section 8.2.1.B12: "It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it." Authors are also advised of IEEE PSPB Operations Manual section 8.1.1B: "Statements and opinions given in work published by the IEEE are the expression of the authors."

RETAINED RIGHTS/TERMS AND CONDITIONS

- Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
- Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author's personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale.
- Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use.The IEEE Intellectual Property Rights office must handle all such third-party requests.
- Authors whose work was performed under a grant from a government funding agency are free to fulfill any deposit mandates from that funding agency.

AUTHOR ONLINE USE

- **Personal Servers**. Authors and/or their employers shall have the right to post the accepted version of IEEE-copyrighted articles on their own personal servers or the servers of their institutions or employers without permission from IEEE, provided that the posted version includes a prominently displayed IEEE copyright notice and, when published, a full citation to the original IEEE publication, including a link to the article abstract in IEEE Xplore. Authors shall not post the final, published versions of their papers.
- **Classroom or Internal Training Use.** An author is expressly permitted to post any portion of the accepted version of his/her own IEEE-copyrighted articles on the author's personal web site or the servers of the author's institution or company in connection with the author's teaching, training, or work responsibilities, provided that the appropriate copyright, credit, and reuse notices appear prominently with the posted material. Examples of permitted uses are lecture materials, course packs, e-reserves, conference presentations, or in-house training courses.
- **Electronic Preprints.** Before submitting an article to an IEEE publication, authors frequently post their manuscripts to their own web site, their employer's site, or to another server that invites constructive comment from colleagues. Upon submission of an article to IEEE, an author is required to transfer copyright in the article to IEEE, and the author must update any previously posted version of the article with a prominently displayed IEEE copyright notice. Upon publication of an article by the IEEE, the author must replace any previously posted electronic versions of the article with either (1) the full citation to the

IEEE work with a Digital Object Identifier (DOI) or link to the article abstract in IEEE Xplore, or (2) the accepted version only (not the IEEE-published version), including the IEEE copyright notice and full citation, with a link to the final, published article in IEEE Xplore.

Questions about the submission of the form or manuscript must be sent to the publication's editor.
Please direct all questions about IEEE copyright policy to:
IEEE Intellectual Property Rights Office, copyrights@ieee.org, +1-732-562-3966

IEEE COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

Where Are You Taking Me? Behavioral Analysis of Open DNS Resolvers
Jeman Park, Aminollah Khormali, Manar Mohaisen, Aziz Mohaisen
2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)

COPYRIGHT TRANSFER

The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the "IEEE") all rights under copyright that may exist in and to: (a) the Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

GENERAL TERMS

1. The undersigned represents that he/she has the power and authority to make and execute this form.
2. The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.
3. The undersigned agrees that publication with IEEE is subject to the policies and procedures of the IEEE PSPB Operations Manual.
4. In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall be null and void. In this case, IEEE will retain a copy of the manuscript for internal administrative/record-keeping purposes.
5. For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.
6. The author hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the author has obtained any necessary permissions. Where necessary, the author has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE

You have indicated that you DO wish to have video/audio recordings made of your conference presentation under terms and conditions set forth in "Consent and Release."

CONSENT AND RELEASE

1. In the event the author makes a presentation based upon the Work at a conference hosted or sponsored in whole or in part by the IEEE, the author, in consideration for his/her participation in the conference, hereby grants the IEEE the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive, in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the "Presentation"). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IEEE and live or recorded broadcast of the Presentation during or after the conference.
2. In connection with the permission granted in Section 1, the author hereby grants IEEE the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice and biographical information as part of the advertisement, distribution and sale of products incorporating the Work or Presentation, and releases IEEE from any claim based on right of privacy or publicity.

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Jeman Park	09-04-2019
Signature	**Date (dd-mm-yyyy)**

Information for Authors

AUTHOR RESPONSIBILITIES

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE PSPB Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE's publishing policies may be found at http://www.ieee.org/publications_standards/publications/rights/authorrightsresponsibilities.html Authors are advised especially of IEEE PSPB Operations Manual section 8.2.1.B12: "It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it." Authors are also advised of IEEE PSPB Operations Manual section 8.1.1B: "Statements and opinions given in work published by the IEEE are the expression of the authors."

RETAINED RIGHTS/TERMS AND CONDITIONS

- Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
- Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author's personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale.
- Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use.The IEEE Intellectual Property Rights office must handle all such third-party requests.
- Authors whose work was performed under a grant from a government funding agency are free to fulfill any deposit mandates from that funding agency.

AUTHOR ONLINE USE

- **Personal Servers**. Authors and/or their employers shall have the right to post the accepted version of IEEE-copyrighted articles on their own personal servers or the servers of their institutions or employers without permission from IEEE, provided that the posted version includes a prominently displayed IEEE copyright notice and, when published, a full citation to the original IEEE publication, including a link to the article abstract in IEEE Xplore. Authors shall not post the final, published versions of their papers.
- **Classroom or Internal Training Use.** An author is expressly permitted to post any portion of the accepted version of his/her own IEEE-copyrighted articles on the author's personal web site or the servers of the author's institution or company in connection with the author's teaching, training, or work responsibilities, provided that the appropriate copyright, credit, and reuse notices appear prominently with the posted material. Examples of permitted uses are lecture materials, course packs, e-reserves, conference presentations, or in-house training courses.
- **Electronic Preprints.** Before submitting an article to an IEEE publication, authors frequently post their manuscripts to their own web site, their employer's site, or to another server that invites constructive comment from colleagues. Upon submission of an article to IEEE, an author is required to transfer copyright in the article to IEEE, and the author must update any previously posted version of the article with a prominently displayed IEEE copyright notice. Upon publication of an article by the IEEE, the author must replace any previously posted electronic versions of the article with either (1) the full citation to the

IEEE work with a Digital Object Identifier (DOI) or link to the article abstract in IEEE Xplore, or (2) the accepted version only (not the IEEE-published version), including the IEEE copyright notice and full citation, with a link to the final, published article in IEEE Xplore.

Questions about the submission of the form or manuscript must be sent to the publication's editor.
Please direct all questions about IEEE copyright policy to:
IEEE Intellectual Property Rights Office, copyrights@ieee.org, +1-732-562-3966

LIST OF REFERENCES

[1] Cloud delivered enterprise security by opendns. https://www.opendns.com.

[2] Public DNS - google developers. https://developers.google.com/speed/public-dns/.

[3] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Comparing DNS resolvers in the wild. In *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2010.

[4] M. Antonakakis, D. Dagon, X. Luo, R. Perdisci, W. Lee, and J. Bellmor. A centralized monitoring infrastructure for improving DNS security. In *proceedings of the International Symposium on Recent Advances in Intrusion Detection (RAID)*, 2010.

[5] Apache HTTP server benchmarking tool. https://httpd.apache.org/docs/2.4/en/programs/ab.html.

[6] C. API. http://docs.cymon.io/.

[7] S. Barnum. Standardizing cyber threat intelligence information with the structured threat information expression (stix^{TM}). *MITRE Corporation*, 11, 2012.

[8] S. Brown, J. Gommers, and O. Serrano. From cyber security information sharing to threat management. In *Proceedings of ACM WISCS*, 2015.

[9] Censys. https://censys.io/data/.

[10] CloudFlare. The ddos that knocked spamhaus offline (and how we mitigated it). http://blog.cloudflare.com/the-ddos-that-knocked-spamhaus-offline-and-ho, 2013.

[11] Corero. Short, stealthy, sub-saturating ddos attacks pose greatest security threat to businesses. https://bit.ly/2N9ciEe, 2017.

[12] Cymon. https://cymon.io/208.91.197.91.

[13] D. Dagon, N. Provos, C. P. Lee, and W. Lee. Corrupted DNS resolution paths: The rise of a malicious resolution authority. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2008.

[14] J. Damas, M. Graff, and P. Vixie. Extension mechanisms for DNS (EDNS(0)). IETF RFC 6891, 2013.

[15] L. Dandurand and O. S. Serrano. Towards improved cyber security information sharing. In *Proceedings of IEEE CyCon*, 2013.

[16] Z. Durumeric, E. Wustrow, and J. A. Halderman. Zmap: Fast internet-wide scanning and its security applications. In *Proceedings of the USENIX Security Symposium*, 2013.

[17] D. Eastlake. Domain name system (dns) iana considerations. IETF RFC 6895, 2013.

[18] M. Feldman and J. Chuang. Overcoming free-riding behavior in peer-to-peer systems. *ACM SIGecom Exchanges*, 5(4):41–50, 2005.

[19] GoDaddy. https://www.godaddy.com/.

[20] J. C. Haass, G.-J. Ahn, and F. Grimmelmann. Actra: A case study for threat information sharing. In *Proceedings of WISCS*, 2015.

[21] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita. Botnet in ddos attacks: trends and challenges. *IEEE Communications Surveys & Tutorials*, 17(4):2242–2270, 2015.

[22] J. Hur. Improving security and efficiency in attribute-based data sharing. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 25(10):2271–2282, 2013.

[23] P. Kampanakis. Security automation and threat information-sharing options. *IEEE Security & Privacy*, 12(5):42–51, 2014.

[24] M. S. Kang, V. D. Gligor, and V. Sekar. SPIFFY: inducing cost-detectability tradeoffs for persistent link-flooding attacks. In *Proc. of NDSS*, 2016.

[25] M. S. Kang, S. B. Lee, and V. D. Gligor. The crossfire attack. In *Proc. of IEEE S&P*, 2013.

[26] Y.-M. Ke, C.-W. Chen, H.-C. Hsiao, A. Perrig, and V. Sekar. Cicadas: Congesting the internet with coordinated and decentralized pulsating attacks. In *Proc. of ACM ASIACCS*, 2016.

[27] R. Kohavi and R. Longbotham. Online experiments: Lessons learned. *IEEE Computer*, 40(9), 2007.

[28] M. Kührer, T. Hupperich, J. Bushart, C. Rossow, and T. Holz. Going wild: Large-scale classification of open DNS resolvers. In *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2015.

[29] M. Kührer, T. Hupperich, C. Rossow, and T. Holz. Exit from hell? reducing the impact of amplification ddos attacks. In *Proceedings of the USENIX Security Symposium*, 2014.

[30] E. Luiijf and M. Klaver. On the sharing of cyber security information. In *Proceedings of Springer ICCIP*, 2015.

[31] J. Luo and X. Yang. The newshrew attack: A new type of low-rate tcp-targeted dos attack. In *Proc. of IEEE ICC*, 2014.

[32] R. A. Martin. Making security measurable and manageable. In *Proceedings of IEEE MILCOM*, 2008.

[33] D. A. Menascé. Qos issues in web services. *IEEE Internet Computing*, 6(6):72–75, 2002.

[34] D. L. Mills. Internet time synchronization: the network time protocol. *IEEE Transactions on Communications*, 39(10):1482–1493, 1991.

[35] P. V. Mockapetris. Domain names: Implementation specification. IETF RFC 883, 1983.

[36] P. V. Mockapetris. Domain names - concepts and facilities. IETF RFC 1034, 1987.

[37] P. V. Mockapetris. Domain names - implementation and specification. IETF RFC 1035, 1987.

[38] A. Mohaisen, O. Al-Ibrahim, C. A. Kamhoua, K. A. Kwiat, and L. Njilla. Rethinking information sharing for threat intelligence. In *Proceedings of the ACM/IEEE HotWeb*, 2017.

[39] A. Mohaisen and O. Alrawi. Unveiling zeus: automated classification of malware samples. In *Proc. of ACM WWW*, 2013.

[40] A. Mohaisen and O. Alrawi. AMAL: high-fidelity, behavior-based automated malware analysis and classification. In *Proceedings of WISA*, 2014.

[41] A. Mohaisen and O. Alrawi. *Detection of Intrusions and Malware, and Vulnerability Assessment: 11th International Conference, DIMVA 2014*, chapter AV-Meter: An Evaluation of Antivirus Scans and Labels, pages 112–131. 2014.

[42] A. Mohaisen, O. Alrawi, and M. Mohaisen. Amal: High-fidelity, behavior-based automated malware analysis and classification. *ACM Computers & Security*, 52:251–266, 2015.

[43] Open Resolver Project. `http://openresolverproject.org/`.

[44] J. Park, H. Alasmary, O. Al-Ibrahim, C. Kamhoua, K. Kwiat, L. Njilla, and A. Mohaisen. Qoi: Assessing participation in threat information sharing. In *Proc. of IEEE ICASSP*, 2018.

[45] J. Park, A. Khormali, M. Mohaisen, and A. Mohaisen. Where are you taking me? behavioral analysis of open dns resolvers. In *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 493–504. IEEE, 2019.

[46] J. Park, M. Mohaisen, D. Nyang, and A. Mohaisen. Assessing the effectiveness of pulsing denial of service attacks under realistic network synchronization assumptions. *Computer Networks*, 2020.

[47] J. Park, D. Nyang, and A. Mohaisen. Timing is almost everything: Realistic evaluation of the very short intermittent ddos attacks. In *Proc. of PST*. IEEE, 2018.

[48] P. Pearce, B. Jones, F. Li, R. Ensafi, N. Feamster, N. Weaver, and V. Paxson. Global measurement of DNS manipulation. In *Proceedings of the USENIX Security Symposium*, 2017.

[49] S. Qamar, Z. Anwar, M. A. Rahman, E. Al-Shaer, and B.-T. Chu. Data-driven analytics for cyber-threat intelligence and information sharing. *Elsevier Computers & Security*, 67:35–58, 2017.

[50] Rapid7. `https://opendata.rapid7.com/`.

[51] R. Rasti, M. Murthy, N. Weaver, and V. Paxson. Temporal lensing and its application in pulsing denial-of-service attacks. In *Proc. of IEEE S&P*, 2015.

[52] RUBBoS. `http://jmob.ow2.org/rubbos.html`.

[53] K. Schomp, T. Callahan, M. Rabinovich, and M. Allman. Assessing DNS vulnerability to record injection. In *Proceedings of the International Conference on Passive and Active Measurement (PAM)*, 2014.

[54] M. Schuchard, A. Mohaisen, D. Foo Kune, N. Hopper, Y. Kim, and E. Y. Vasserman. Losing control of the internet: using the data plane to attack the control plane. In *Proc. of NDSS*, 2011.

[55] Shadowserver. `https://dnsscan.shadowserver.org/`.

[56] H. Shan, Q. Wang, and Q. Yan. Very short intermittent ddos attacks in an unsaturated system. In *Proc. of SecureComm*, 2017.

[57] A. Shevtekar and N. Ansari. Is it congestion or a ddos attack? *IEEE Communications Letters*, 13(7), 2009.

[58] C. Sillaber, C. Sauerwein, A. Mussmann, and R. Breu. Data quality challenges and future research directions in threat intelligence sharing practice. In *Proceedings of WISCS*, 2016.

[59] G. Sisson. Dns survey: October 2010. http://dns.measurement-factory.com/surveys/201010/dns_survey_2010.pdf, 2010.

[60] F. Skopik, G. Settanni, and R. Fiedler. A problem shared is a problem halved: A survey on the dimensions of collective cyber defense through security information sharing. *Elsevier Computers & Security*, 60:154–176, 2016.

[61] J. M. Smith and M. Schuchard. Routing around congestion: Defeating ddos attacks and adverse network conditions via reactive bgp routing. In *Proc. of IEEE S&P*, 2018.

[62] Y. Takano, R. Ando, T. Takahashi, S. Uda, and T. Inoue. A measurement study of open resolvers and dns server version. In *Proceedings of the Internet Conference (IC)*, 2013.

[63] H. Tanaka, K. Matsuura, and O. Sudoh. Vulnerability and information security investment: An empirical analysis of e-local government in japan. *Elsevier Journal of Accounting and Public Policy*, 24(1):37–59, 2005.

[64] M. Thomas and A. Mohaisen. Kindred domains: detecting and clustering botnet domains using dns traffic. In *Proceedings of the ACM International Conference on World Wide Web (WWW)*, 2014.

[65] D. K. Tosh, S. Sengupta, C. A. Kamhoua, K. A. Kwiat, and A. P. Martin. An evolutionary game-theoretic framework for cyber-threat information sharing. In *Proceedings of IEEE ICC*, 2015.

[66] R. Tracker. https://ransomwaretracker.abuse.ch/ip/208.91.197.91/.

[67] J. C. M. S. V. Paxson, M. Allman. Computing tcp's retransmission timer. IETF RFC 6298, 2011.

[68] Vultr. https://www.vultr.com/.

[69] C. Wagner, A. Dulaunoy, G. Wagener, and A. Iklody. Misp: The design and implementation of a collaborative threat intelligence sharing platform. In *Proceedings of WISCS*, 2016.

[70] A. Wang, A. Mohaisen, W. Chang, and S. Chen. Capturing ddos attack dynamics behind the scenes. In *Proc. of DIMVA*, 2015.

[71] A. Wang, A. Mohaisen, W. Chang, and S. Chen. Delving into internet ddos attacks by botnets: Characterization and analysis. In *Proc. of IEEE DSN*, 2015.

[72] N. Weaver, C. Kreibich, and V. Paxson. Redirecting DNS for ads and profit. In *Proceedings of the USENIX Workshop on Free and Open Communications on the Internet (FOCI)*, 2011.

[73] S. Yu, W. Zhou, W. Jia, S. Guo, Y. Xiang, and F. Tang. Discriminating ddos attacks from flash crowds using flow correlation coefficient. *IEEE Transactions on Parallel and Distributed Systems*, 23(6):1073–1080, 2011.