USING STAND-OFF OBSERVATION AND MEASUREMENT TO UNDERSTAND ASPECTS OF THE GLOBAL INTERNET

by

Rui Bian

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical and Computer Engineering

Fall 2022

USING STAND-OFF OBSERVATION AND MEASUREMENT TO UNDERSTAND ASPECTS OF THE GLOBAL INTERNET

by

Rui Bian

Approved: ____
Jamie D. Phillips, Ph.D.
Chair of the Department of Electrical and Computer Engineering

Approved: ____
Levi T. Thompson, Ph.D.
Dean of the College of Engineering

Approved: ____
Louis F. Rossi, Ph.D.
Vice Provost for Graduate and Professional Education and
Dean of the Graduate College

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed:

Chase Cotton, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed:

Haining Wang, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed:

Xiaoming Li, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed:

Shuai Hao, Ph.D.
Member of dissertation committee

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincerest gratitude to my advisors, Dr. Haining Wang and Dr. Chase Cotton, for their invaluable guidance and continuous support throughout my Ph.D. studies. Their dedication and enthusiasm for research and rigorous academic attitude deeply impact my career and life. Without their help, this dissertation would not have been possible.

Next, I would like to thank Dr. Xiaoming Li and Dr. Shuai Hao for serving on my dissertation committee and providing their insightful comments.

Furthermore, I would like to thank the members of our research group including but not limited to Lin Jin, Zeyu Chen, Rebekah Houser, Teddy Katayama, Guannan Liu, and Yubao Zhang, for the constructive suggestions and creative collaborations. Also, I would like to express my appreciation to all of my friends in Newark, DE for making my Ph.D. journey pleasurable.

This work is dedicated to my children, Benjamin and Sylvia. You have made me stronger, better and more fulfilled than I could have ever imagined. I love you to the moon and back. Last but not the least, I thank my parents for their unconditional support and encouragement. Thank you. In addition, my special thanks go to my wife, Mengyao Ma, for her warm love and support.

TABLE OF CONTENTS

Chapter

Appendix

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The Internet has become a central part of our daily lives. In the meantime, the Internet is a very complex system and it is challenging to understand the nature of the Internet ecosystem from different perspectives. To extend our knowledge of the Global Internet and better understand the nature of the Internet, we design unique active and passive measurements to study several crucial components of the Internet, including anycast in global routing, open proxy ecosystem, and transparent proxy systems.

Anycast has been widely adopted by today's Internet services, including DNS, CDN, and DDoS protection. Prior research has focused on various aspects of anycast, either its usage in particular services such as DNS or characterizing its adoption by Internet-wide active probing methods. We first explore an alternative approach to characterize anycast based on previously collected global BGP routing information. Leveraging state-of-the-art active measurement results as near-ground-truth, our passive method without requiring any Internet-wide probes can achieve high accuracy in detecting anycast prefixes. While investigating the root causes of inaccuracy, we reveal that anycast routing has been entangled with the increased adoption of remote peering. The invisibility of remote peering from layer-3 breaks the assumption of the shortest AS paths on BGP and causes an unintended impact on anycast performance.

Open proxies provide free relay services and are widely used to anonymously browse the Internet, avoid geographic restrictions, and circumvent censorship. To shed light on the ecosystem of open proxies and characterize the behaviors of open proxies, we conduct a large-scale, comprehensive study. We characterize open proxies based on active and passive measurements and examine their network and geographic distributions, performance, and deployment. In particular, to obtain a more in-depth and broader understanding of open proxies, we analyze two particular groups of open proxies—cloud-based proxies and long-term proxies. To process and analyze the enormous amount of responses, we design a lightweight method that classifies and labels the proxies based on DOM structure which defines the logical structure of Web documents. Furthermore, we parse the contents to extract information to identify the owners of proxies and track their activities for deploying malicious proxies. We reveal that some owners regularly change the proxy deployment to avoid being blocked and deploy more proxies to expand their malicious attacks.

Transparent proxies are one type of web proxies that host between clients and servers. Transparent proxies intercept requests and responses between clients and web servers. In this work, we study an overlooked issue around web browsing, the hidden interception of the HTTP path by on-path devices, which is not yet thoroughly studied and well understood by previous works. We propose a novel method that utilizes designed requests to detect the interception to discover the hidden transparent proxies. We characterize various aspects of transparent proxies – geographically and AS level distribution, server hosting, software, and services. We investigate the vulnerabilities of transparent proxies and examine the impact on end-users.

Chapter 1
INTRODUCTION

The Internet has become a vital part of modern society and has changed humans' daily life significantly. As the Internet grows larger, measuring and characterizing its dynamics becomes harder. It is challenging to understand the nature of the Internet ecosystem from different perspectives. To better understand the nature of the Internet and expand our knowledge of the global Internet, we design a set of novel measurement techniques to explore and analyze the vital components of the Internet. More specifically, we studied anycast in global routing, open proxy ecosystems, and transparent proxy systems.

In this dissertation, our first study focuses on passively characterizing anycast prefixes based on BGP information and understanding remote peering's effect. The second study analyzes the open proxy ecosystem and related security problems. In our third study, we explore HTTP interception and cache poisoning problems in transparent proxies. In Sections 1.1 to 1.3, we introduce the motivation of each study, and the organization of this dissertation is described in Section1.4.

1.1. Characterizing Anycast Prefixes and Understanding Remote Peering's effect

Anycast is a network addressing and routing methodology in which the single destination IP address is announced from multiple locations. The Border Gateway Protocol (BGP) is responsible for directing clients to the site that is the "closest" to them based on "best routing" (i.e., AS path), providing reduced latency and improved availability to end-users. In recent years, researchers have conducted studies to understand and characterize anycast from various aspects. Due to the inefficient distinctions between unicast and anycast from the perspective of a routing table, the common method to identify anycast addresses is through *active* Internet-wide measurements. One method is using latency measurements based on the detection of speed-of-light violations. However, the latency of ping may not always reliably reflect the geographic distance of two IP addresses. Also, the active probing method requires many vantage points to achieve the necessary coverage. To overcome these limits, we explore a passive approach to identify and characterize IP anycast by leveraging BGP routing information. Specifically, we propose and analyze BGP-related features to classify anycast and unicast prefixes and utilize simple classifiers to train and predict anycast prefixes on the Internet. Furthermore, we delve into the misclassified instances to find the root causes of inaccuracy. Through a deeper analysis, we identify that many of these cases involve *remote peering* [46, 93]. Remote peering allows a network to peer at an Internet exchange point (IXP) without a physical presence within the IXP's infrastructure. Remote peering can improve connectivity and reduce costs. However, it also brings an unintended impact on global routing due to its invisibility at layer-3, breaking the assumption that the peered autonomous systems are physically close and provide a short path for transporting traffic. As such, we investigate the impact of remote peering on anycast routing using passive methods and validate our analysis through traceroute results.

1.2. Open Proxy Ecosystem Analysis

Open proxies provide relay service to clients free of charge, allowing them to anonymously browse the Internet, bypass geographic restrictions, or circumvent censorship. Those enormous numbers of proxies have formed a giant and complicated ecosystem. Researchers have conducted studies to explore and characterize the open proxies in various aspects, such as performance, behaviors, security, and distributions [112, 107, 82, 97, 50]. However, the ownership of those malicious proxies and corresponding campaigns have not been well studied before. In particular, open proxy owners can control plentiful proxies in diverse areas at different times to strengthen the effectiveness of their activities or campaigns. Also, they could frequently change open proxies' deployment and behaviors to hide their activities and avoid being detected. Thus, a systematic investigation of how to open proxies are deployed and managed on the Internet is sorely needed but still missing. We perform a large-scale, comprehensive measurement-based analysis of open proxies. We design a measurement methodology to facilitate the analysis of massive returned responses from open proxies and accurately identify the proxies that manifest similar behaviors, possibly controlled by the same owner, to create a campaign. Moreover, to increase the understanding of the open proxy ecosystem, we study two special open proxy groups, cloud-based proxies and long-term proxies.

1.3. A Large-scale Analysis of Transparent Proxies in the Internet

Transparent proxies [56, 120, 123, 74, 49] are one type of web proxy that are deployed between clients and servers. Transparent proxies intercept requests and responses, but clients and web servers may not realize the existence of transparent proxies. The transparent proxies can be deployed by Internet service providers, enterprises, and clients, so that the ISPs, enterprises or client can monitor, filter and censor the traffic. Also, by caching the content, transparent proxies can reduce the traffic volume effectively to decrease the cost. There are only a few prior studies measuring and studying transparent web proxies [120, 123]. We investigate an overlooked issue of web browsing, the hidden interception of the HTTP path by on-path devices especially transparent proxies, which is not yet thoroughly studied and well understood. HTTP queries from clients are typically handled by the requested web servers. However, if transparent proxies handle such queries and transparent proxies understand and process the requests differently from the original web server, the responses could be different from desired results, which will bring problems. For example, some transparent proxies ignore the destination IP address in request but use forced DNS resolution results to send requests. We design a framework using this behavior to detect transparent proxies. More importantly, the HTTP interceptions are not authorized by users and are near impossible to detect on the user's side, which leads to security and ethical concerns. Users have higher risks of putting their trust in transparent proxy servers, which often lack proper maintenance (e.g. equipped with the outdated web server software), compared to a well-known companies' web servers. Users' private information may be exposed to rogue transparent proxy owners, which could cause large damages. Transparent proxies are also vulnerable to cache poisoning and other attacks such as CPDoS. We conduct a large-scale analysis of transparent proxies to study HTTP interception, cache poisoning and other security problems around transparent proxies.

1.4. Roadmap

The remainder of this dissertation is organized as follows. In Chapter 2, we propose a passive way to characterize anycast prefixes based on BGP information and examine the remote peering effect on anycast prefixes classification. In Chapter 3, we analyze a significant amount of collected open proxies. We thoroughly study the distribution, ownership, and malicious behaviors of open proxies. In Chapter 4, we study the HTTP interceptions of transparent proxies using a large number of globally distributed vantage points. On the other hand, we examine the cache poisoning and denial-of-service attack vulnerabilities of transparent proxies on the Internet. Finally, we summarize this dissertation in Chapter 5.

Chapter 2

CHARACTERIZING ANYCAST PREFIXES AND UNDERSTANDING REMOTE PEERING'S EFFECT

IP anycast is widely used in modern Content Delivery Networks (CDNs) [45], Domain Name System (DNS) [62, 89], and Distributed Denial of Service (DDoS) protections [89]. With anycast, the same IP address(es) is announced from multiple locations, and the Border Gateway Protocol (BGP) is responsible for directing clients to the site that is the "closest" to them on the basis of "best routing" (i.e., AS path), providing reduced latency and improved availability to end-users.

In recent years, researchers have conducted studies to understand and characterize anycast from various angles, such as its adoption [52] or the efficiency in particular services like DNS [79]. Due to the insufficient distinctions between unicast and anycast from the perspective of a routing table, the common method to identify anycast addresses is through *active* Internet-wide measurements. Cicalese *et al.* [52, 53] studied the enumeration and city-level geolocation of anycast prefixes by using latency measurements based on the detection of speed-of-light violations. However, the latency of ping may not always reliably reflect the geographic distance of two IP addresses [42, 122]. Also, active probing requires the use of many vantage points to achieve the necessary coverage.

To overcome these limitations, in this work, we explore a passive approach to identify and characterize IP anycast by leveraging BGP routing information. Specifically, we propose and analyze a set of BGP-related features to classify anycast and unicast prefixes, and utilize simple classifiers to train and predict anycast prefixes on the Internet. The results demonstrate that our passive approach, without requiring probing, can achieve 90% accuracy. Furthermore, we delve into the instances misclassified by our approach to find the root causes of inaccuracy.

The two major assumptions of our approach are that (1) anycast prefixes may have more upstream autonomous systems (ASes) than unicast prefixes, as anycast is announced from multiple physical locations and peering with transit providers at different places, and (2) the distance between such upstream ASes will be topologically larger than that in the scenarios of unicast prefixes (i.e., more hops in AS paths), as some of them are geographically distant from others. However, in our false positives, we also find some unicast prefixes falling into such a category. Through a deeper analysis, we identify that many of these cases involve *remote peering* [46, 93].

Remote peering allows a network to peer at an Internet exchange point (IXP) without a physical presence within the IXP's infrastructure, either over a long cable or over IXP's reseller partners that provide IXP layer-2 access. Remote peering enables the fast deployment of connectivity to an IXP and reduces cost. However, it also brings unintended impact on global routing due to its invisibility at layer-3, breaking the assumption that the peered autonomous systems are physically close and provide a short path for transporting traffic. As such, we investigate the impact of remote peering on anycast routing by using passive methods and validate our analysis through traceroute results.

The remainder of this chapter is organized as follows. We introduce the background of anycast and remote peering §2.1. We present our methodology to identify anycast prefixes in §2.2. We investigate inaccuracies in our method in §2.3 and the impact of remote peering on anycast routing in §2.4. We survey related work in §2.5 and summarize the chapter in §2.6.

2.1. Background

2.1.1 BGP and Anycast

Border Gateway Protocol (BGP) [99] is the de facto inter-domain routing protocol, designed to exchange reachability information among autonomous systems on the

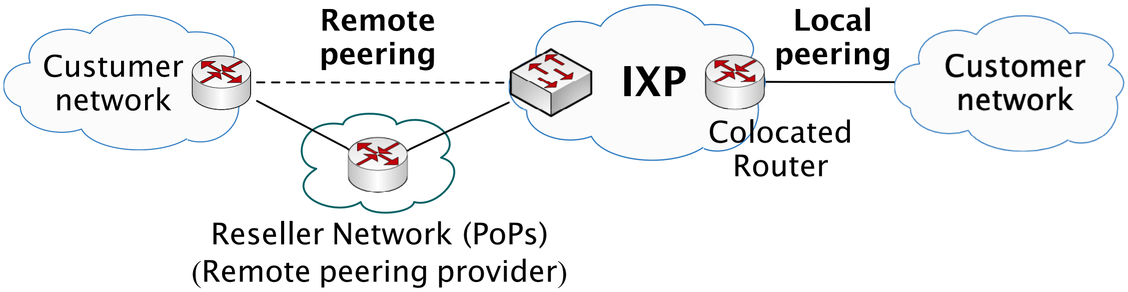


Figure 2.1: Local and Remote Peering Models

Internet. BGP selects a best AS path based on various attributes (e.g., the shortest path) to reach the specific destination.

Anycast [32] is a network addressing and routing methodology by which a collection of servers announce the same IP address from multiple geographically distributed sites. As routers usually choose the shortest AS path, the user requests sent to an anycast address are routed to the topologically nearest endpoint. As a result, anycast has many advantages over unicast such as reduced latency, load balancing, DDoS mitigation, and improved robustness.

2.1.2 Remote Peering

Peering is a relationship where two networks exchange traffic directly rather than through a transit provider. Remote peering [46, 93] is a new peering type where a network peers at an IXP through layer-2 remote peering providers such as resellers without a physical presence in the IXP's infrastructure. Fig. 2.1 shows an example of remote peering. Remote peering can be implemented with standard methods like MPLS (Multi-Protocol Label Switching) and VPNs (Virtual Private Networks) in layer-2, and provide benefits such as low cost, increased connectivity, and easy management. Nevertheless, it also has some drawbacks such as degradation of performance, loss of resilience, and difficulty for layer-3 management [93]. Furthermore, due to the invisibility at layer-3, BGP routers are not aware of remote peering and may select as the shortest path a route where the actual endpoints are far from one another.

2.2. Methodology

In this section, we describe the datasets and the features we propose to extract from passively-collected BGP data for the purpose of identifying anycast routing. Using a reference dataset as *near*-ground-truth, we characterize the behavior of such BGP-related features in the wild. We then employ standard classification methods, decision tree and random forest, to train and evaluate the effectiveness of our approach for anycast detection using our proposed classification features. The repository including scripts and data used in our study is available at [31].

2.2.1 Datasets

BGP Routing Information.

The datasets we used to detect and characterize anycast prefixes are from the RouteViews project [105] and RIPE's Routing Information Service (RIS) [103]. In RouteViews and RIPE RIS, servers receive BGP information by peering with other BGP routers, often at large IXPs. We use CAIDA's BGPStream [95] to collect and process the data from RouteViews and RIPE RIS.

Anycast Dataset.

We use the anycast prefix list obtained through active measurements by Cicalese et al. [52] as *near*-ground-truth, which provides a *conservative* estimation of Internet anycast usage. The detection method in [52] is based on speed-of-light violations: if the latency measurements from multiple vantage points towards the same target exhibit geo-inconsistency, the target is classified as anycast. They validated their method and scrutinized the dataset they make publicly available [34] using ground-truth collected through protocol-specific techniques (e.g., DNS CHAOS requests or DPI over HTTP).

However, we also notice that some prefixes strongly suggested as anycast by our method are not included in their dataset. We manually check and, through traceroute measurements, verify that most of them are indeed anycast prefixes.

2.2.2 BGP-related Features

Due to the different deployment patterns between anycast and unicast, we leverage BGP routing information to characterize anycast prefixes. We propose and explore the following BGP-related features that could be used to identify anycast prefixes: as an anycast prefix is announced from multiple locations, some of its peer ASes should not be close to one another, both geographically and topologically.

`N` **- Number of upstream ASes:** We count the number of unique *upstream* ASes of each prefix. Given a prefix announced by AS_n, we define *upstream* ASes as the set of AS_n's neighbor ASes that are connected to AS_n with either a customer-to-provider relationship (i.e., AS_n's transit providers) or a peer-to-peer relationship, according to CAIDA's AS Relationships Dataset [44].

`P1` **- Percentage of upstream AS pairs whose distance is more than 1:** We define the *distance between two ASes* as the least number of AS hops between them in the observed paths. For each prefix, we construct all the AS pairs between its upstream AS neighbors and label the number of AS pairs as P. We then identify the fraction of those AS pairs whose distance is more than one, i.e., $P1 = \ P_{dist>1}/\ P$.

`P2` **- Percentage of upstream-AS pairs whose distance is more than 2:** Similarly, P2 is defined as the fraction of those AS pairs with distance more than two, i.e., $P2 = \ P_{dist>2}/\ P$. Note that we propose P1 and P2 based on the assumption that the upstream ASes of an anycast prefix are more likely to be remote, both geographically and topologically.

`MD` **- Maximum distance between upstream ASes:** MD is the largest distance of two upstream ASes of a prefix. This variable tries to capture that upstream ASes for anycast prefixes are more spread out compared to unicast.

`ML` **- Maximum length of AS paths:** ML represents the length of the longest AS path observed for a prefix. AS paths towards anycast prefixes tend to be shorter, since they are announced from multiple locations.

2.2.3 Feature Validation

Given the features we proposed, we explore their potential for identifying anycast prefixes by analyzing their behavior with respect to prefixes labeled in the *near*-ground-truth dataset.

`N` : Figure 2.2(a) shows the distributions of the number of upstream ASes, where we can see that the two classes of prefixes are clearly distinguishable from each other. Most anycast prefixes (90.2%) have more than 17 upstream ASes, while 69.5% of unicast prefixes only have one or two upstream ASes. This is consistent with the intuition that the routes towards an anycast prefix would be highly varied due to the geographically distributed deployment.

`P1`: Figure 2.2(b) shows the distributions of P1. Obviously, P1 of anycast prefixes is much larger than P1 of unicast prefixes. Specifically, P1 is greater than 0.33 for 91.9% of anycast prefixes, and smaller than 0.07 for 78.1% of unicast prefixes. A larger P1 for anycast prefixes implies that the upstream ASes are relatively far from one another because the upstream ASes of an anycast prefix are more geographically and topologically distributed.

`P2`: Similar to P1, from Figure 2.2(c), P2 is smaller than 1% for 95.4% of unicast prefixes but larger than 7% for 73.7% of anycast prefixes.

`MD`: Figure 2.2(d) shows the distributions of maximum distance between upstream ASes for anycast and unicast prefixes. About 83.1% of anycast's MD is greater than 8 but 76.8% of unicast prefixes' MD is smaller than 1.

`ML`: Figure 2.2(e) shows the distributions of the longest AS paths for anycast and unicast prefixes. The ML of most anycast prefixes (93.3%) is smaller than three hops, while only 18.3% of ML for unicast prefixes are less than three. Anycast usually has a shorter maximum AS path than unicast, because anycast traffic is typically routed to the closest replica.

Table 2.1: Number of Prefixes in Classification

	total	training	testing
Anycast	3,907	2,609	1,298
Unicast	728,010	487,775	240,235
total	731,917	490,384	241,533

Table 2.2: Evaluation of Classifiers

	precision	recall	f1-score
Decision Tree	90.98%	89.45%	90.21%
Random Forest	93.94%	89.52%	91.68%

2.2.4 The Classifier

To further validate the effectiveness of identifying anycast from BGP paths, we use a combination of our proposed features to build simple (*decision tree* and *random forest*) classifiers and train them with the *near*-ground-truth datasets by using the scikit-learn library [106] in Python.

The (*near*-)Ground-Truth. The anycast dataset is described in §2.2.1. We use the monthly-refined datasets from 1/2017 to 6/2017 and retrieve the labeled anycast prefixes from a complete snapshot of BGP data by RIPE NCC and RouteViews on 6/1/2017. In total, we extract 3,907 anycast prefixes and label the remaining 728,010 prefixes as unicast.

Evaluation of the Classifiers. We manually divide the labeled prefixes into exclusive training and testing sets, where 66% of the dataset is used for training and the rest is used for testing. We use class-weights to handle unbalanced class sizes in the dataset. Table 2.1 shows the detailed breakdown.

Table 2.2 lists the evaluation results of anycast classification using respectively a random forest and a decision tree classifier. Our results show that both classifiers can achieve high accuracy (more than 90%). Table 2.3 lists the percentage of incorrectly

Table 2.3: Percentage of Mis-Classified Instances

	Anycast	Unicast	Overall
Decision Tree	10.55%	0.05%	0.10%
Random Forest	10.48%	0.03%	0.09%

classified instances. The fractions of incorrectly-labeled anycast prefixes in the two classifiers are 10.55% and 10.48%. For unicast, the misclassification rates are as low as 0.05% and 0.03%, respectively.

2.3. Analyzing Misclassification

After using BGP-related features to classify anycast and unicast prefixes, we further inspect the instances of false negative (anycast prefixes wrongly labeled as unicast prefixes) and false positive (unicast prefixes wrongly labeled as anycast prefixes) to understand the causes of inaccuracy. For false negatives (0.05% and 0.03% in the decision tree and random forest classifiers respectively), we identify that they are mainly caused by geographically distributed autonomous systems. By manually examining false positives (10.55% and 10.48%), we find that the anycast dataset we used does miss some cases that are highly likely to be anycast. Also, we discover that the emerging remote peering introduces unintended impact on the anycast routing, which essentially reduces the distinction between anycast and unicast in our BGP-related features.

The distributions of the studied features of false positives (FP) and false negatives (FN) are also presented in Figure 2.2, which shows that the feature distributions of FN are similar to those of anycast prefixes and the feature distributions of FP are similar to those of unicast prefixes.

False Negative (FN). We misclassify 344 anycast prefixes as unicast. Table 2.4 shows that several FN features have values that are very different from those we find in near-ground-truth data (shown in Figure 2.2). For example, based on our heuristics,

Table 2.4: Anomaly in FN

Feature	Value	% in FN
N	1	46.80
$\texttt{P1}\|_{N\neq1}$	0	18.90
$\texttt{P2}\|_{N\neq1,P1\neq0}$	0	14.82
MD	≤ 4	82.27
ML	> 3	57.85

Table 2.5: Anomaly in FP

Feature	Value	% in FP
N	> 3	99.06
P1	≥ 0.5	82.22
P2	≥ 0.07	77.78
MD	≥ 4	78.09
ML	≤ 3	77.78

anycast prefixes should have relatively large N and P1, i.e., more upstream ASes and more pairs with long distance. However, in FN we observe 46.80% of prefixes with only one upstream AS (N=1). We use RIPEstat Geoloc tool [102] and MaxMind's GeoLite City Dataset [83] to examine the geo-locations of these upstream ASes, and find that all 24 ASes appear in at least three different locations, indicating that an upstream AS whose geographic presence is largely distributed would cause such misclassifications.

Furthermore, given N $\neq$ 1, there are still 18.90% of anycast prefixes in FN with P1=0 (i.e., no upstream AS pair with the distance greater than 1). This could be because some anycast prefixes are not globally distributed (i.e., *regional* anycast deployment [71]), resulting in upstream ASes that are close. Such concentration can contribute to abnormal values for P2, MD, and ML as well.

False Positive (FP). For false positives, the abnormal feature values and percentage of the prefixes with such values are shown in Table 2.5. Table 2.5 shows that the feature values of such "unicast" prefixes are similar to those of anycast prefixes. One possible reason is that these false positives are indeed anycast prefixes but have been wrongly labeled as unicast in the near-ground-truth dataset, which has been actually obtained using a conservative classification approach, avoiding labeling prefixes as anycast when active measurements provide insufficient evidence [52].

We investigate which organizations originate these prefixes. Figure 2.3 shows the owners that possess at least 2 FP prefixes. We observe that 87.3% of false positive prefixes belong to IT companies or infrastructure providers. It is very likely that such organizations have deployed anycast-based services. To validate this intuition, we traceroute to these prefixes from distributed vantage points from RIPE Atlas (in US, Brazil, Japan, Australia, South Africa, and Netherlands). We successfully reach 117 out of 318 FP prefixes. We then leverage the IP geolocation and latency measurements to manually infer the types of these prefixes based on speed-of-light violations. Among these 117 prefixes, 31 of them show strong evidence of anycast routing. Therefore, some of the false positives we obtained are actually true positives, due to the incompleteness of the anycast near-ground-truth dataset (which indeed has been generated using a conservative approach [52]).

However, we do find that several unicast prefixes show a very similar deployment pattern to anycast. By mining the corresponding AS paths and the IP geolocation of intermediate network nodes from traceroutes, we speculate that the main cause is the emerging remote peering deployment. We find that 28.61% (91 out of 318) of the false positives might be caused by remote peering (§2.4.1). For these 91 unicast prefixes, the average values of N, P1, P2, MD, and ML are 7, 0.38, 0, 2, and 6, respectively. These numbers indicate that remote peering will blur the distinction of our features between unicast and anycast. We present a detailed study of the potential impact of remote peering on anycast routing in §2.4.

2.4. Remote Peering in Anycast Routing

The inspection of false positives suggests that remote peering might introduce unintended impact on path selection due to its invisibility at layer-3, where the direct (remote) peering at IXPs leads the local traffic to a distant location. Such a case is especially a disservice to anycast when some clients are directed to a sub-optimal replica. In this section, we attempt to identify the anycast prefixes that could be impacted by remote peering. We retrieve paths (i) towards anycast prefixes and (ii) potentially containing remote peering instances, and we validate those paths through RIPE Atlas measurements. We then perform latency measurements and present specific case studies to illustrate the practical impact of remote peering on anycast routing.

2.4.1 Identifying Remote Peering in Anycast

We leverage the remote peering data from a publicly available dataset, the Remote IXP Peering Observatory [73], in which remote peering instances have been identified in 26 large IXPs worldwide. To identify BGP paths potentially involving remote peering, first we construct AS pairs that are connected through remote peering. We do so by pairing ASNs that according to [73] are connected through remote peering at an IXP (AS_{rp}), with the member ASNs (AS_{mem}) obtained from the same IXP's website: $RP\text{-}AS \rightarrow (AS_{rp}, AS_{mem})$. We then search for such pairs in all AS paths towards anycast prefixes.[1] If there is any such pair appearing in the AS path of an anycast prefix, we label this prefix as potentially affected by remote peering.

The datasets and results are shown in Table 2.6. In all large IXPs of Europe (AMS-IX, CATNIX, DEC-IX Frankfurt, FranceIX and LINX), remote peering has the potential to affect more than 10% of anycast prefixes. In total, there are 19.2% (751/3,907) of anycast prefixes potentially impacted by remote peering.

2.4.2 Path Collection

To collect more information on anycast paths potentially affected by remote peering and further understand its practical impact, we conduct active measurements using the RIPE Atlas platform [101]. We select RIPE Atlas probes from the ASes that (i) host a BGP monitor and (ii) observe anycast routing paths, and perform traceroutes from the probes to the first address of anycast prefixes that are potentially affected by remote peering (§2.4.1). On average, we use 10.3 probes to traceroute a prefix.

[1] Here we use the near-ground-truth dataset (which is more conservative in labeling prefixes as anycast).

We parse the traceroute results to map each IP address to its ASN in order to obtain AS paths. Next, we look for remote peering AS pairs in these AS paths. If found, we collect and label them as paths towards prefixes potentially affected by remote peering.

Table 2.6 lists details for ASes and anycast prefixes involved in remote peering at each IXP for which we have remote peering data [73]. In total, we collect 1,013 AS pairs that are involved in remote peering from 26 IXPs. We find that 751 anycast prefixes (19.2% of total anycast prefixes) are reached through BGP paths that include an RP-AS pair, and we successfully traceroute 688 of them. Since two ASes labeled as a RP-AS pair could also peer locally at other IXPs, we then use the traIXroute [111, 92] open-source tool to identify the IXP crossings in the traceroutes towards these 688 prefixes, looking for IXPs where the remote peering actually occurs. This way, we are able to confirm that 293 of these anycast prefixes are actually affected by remote peering, since both the RP-AS pairs and the corresponding IXPs are detected in traceroutes.

We are not able to draw conclusions for the remaining 458 prefixes (out of 751), because (1) some destination IP addresses are not reachable, (2) some intermediate IP addresses have no matching ASNs, and (3) traIXroute [111] does not include data from all IXPs where the remote peering instances have been detected. Even though these limitations lower the validation rates, we still find a significant portion of anycast prefixes that are reached through paths involving remote peering, which provides a lower bound for this phenomenon.

2.4.3 Impact of Remote Peering: Performance Analysis and Case Study

Leveraging the traceroute experiments we used in §2.4, we study the impact of remote peering by analyzing the performance and route selection in real-world case studies.

Performance Analysis and Case Study. To quantify the performance impact of remote peering on anycast path selection, we measure the round-trip time (RTT) to each anycast prefix from the same measurements collected in §2.4.2. Among the successful traceroutes, we find that 38% (126/332) of RTTs in traceroutes towards anycast prefixes potentially affected by remote peering are larger than the average RTT of prefixes without remote peering. In these 126 traceroute probes, the average RTT towards prefixes potentially affected by remote peering is 119.7 ms while the average RTT of the other prefixes is 84.7 ms. An average latency increase of 35.1 ms.

In a concrete example, we traceroute to the IP address of the DNS D-root from a probe located in Singapore. Ideally, we expect that our traceroute can reach the D-root instance in Singapore [104]. However, we found that the traceroute goes to Europe via AMS-IX and through remote peering, and reach another D-root server in Amsterdam, Netherlands, with a 158 ms RTT. Consequently, remote peering not only can affect performance, but it may also impact traffic engineering or load balancing, potentially routing traffic through to unintended locations.

DNS Root Sever Anycast Data. We conduct an extensive study using a dataset of traceroutes towards anycast addresses provided by University of Maryland (UMD) [60], which includes traceroute data from selected probes to C-, D- and K-DNS root server sites. By searching for IPs/ASes involving remote peering in paths towards such anycast addresses, we identify remote peering in D and K root server traces. Specifically, we find remote peering instances located in AMS-IX and DECIX from D-root experiments, and SIX.SK, FranceIX Paris, AMS-IX and Linx from K-root experiments. These results are consistent with our previous results in §5.2.

Also in the UMD dataset, we find specific cases where remote peering affects anycast routing by taking traffic on geographically-long routes. For example, we observed that traceroutes from probes in Eastern Russia were routed to Netherlands and Germany, respectively, through routes with remote peering, while there are root DNS server instances in Hong Kong and Tokyo. These cases confirm the observations from Li et al. in [79], in which the same dataset has been used to study the inefficiency of anycast path selection, and explain the reason why some users cannot reach the optimal DNS root sites (although the work from Li et al. does not mention remote peering among potential causes).

2.5. Related Work

Anycast deployment and performance have been characterized and evaluated by different active probing methods. Madory *et al.* [81] use geolocation of transit IP and geo-inconsistency to detect anycast prefixes. Cicalese *et al.* [52, 53, 54] propose a method for enumeration and geolocation of anycast instances based on latency measurements. Vries *et al.* [59] propose a method that maps anycast catchments via active probes to provide better coverage.

Anycast-based Internet Services. Fan *et al.* [62] combine the CHAOS queries with traceroutes and use new IN records to support open recursive DNS servers as vantage points to detect and study anycast-based DNS infrastructures. Calder *et al.* [45] study the performance of an anycast CDN and find that some clients are directed to a sub-optimal front-end. Moura *et al.* [89] study the Nov. 2015 event of Root DNS attacked by DDoS from the anycast's perspective. Giordano *et al.* [69] perform a passive characterization study on anycast traffic in CDNs and present temporal properties, service diversity, and deployments of anycast traffic.

Schmidt *et al.* [58] investigate the relationship between IP anycast and latency from four Root DNS nameservers. Their key results show that geographic location and connectivity have a stronger impact on latency than the number of sites. Li *et al.* [79] perform a study on anycast's route selection and performance using D-root Server traces, and they validate that equal-length AS paths are the main reason for anycast latency inflation. Wei *et al.* [116] study the service (in)stability of anycast services. They confirm that a small number of users are affected by the instability of anycast, potentially caused by the load balancers on the path.

Remote Peering. Castro *et al.* [46] present a systematic study of remote peering at IXPs using ping-based methods. They discuss the impact of remote peering on Internet reliability, security, and economies. Nomikos *et al.* [93] perform a comprehensive measurement study of remote peering, and they achieve very high accuracy and coverage levels by combining RTT measurements with other domain-specific information like facility locations, IXP port capacity, and private connectivity. They study the features and trends of remote peering, showing that remote peering may route traffic to more distant destinations. Their work does not focus on anycast prefixes though.

2.6. Summary

We presented a passive method to study IP anycast by utilizing BGP data. We proposed a set of BGP-related features (thus not based on active measurements) to classify anycast and unicast prefixes. Extracting data from RouteViews and RIPE RIS, we evaluated the effectiveness of our proposed approach against a near-ground-truth dataset based on active-probing measurements [52]. The evaluation results show that our approach achieves high classification accuracy—about 90% for anycast and 99% for unicast—and is also able to detect anycast prefixes incorrectly labeled as unicast in the near-ground-truth dataset.

In addition, while delving into the causes of inaccuracy, we found indication that remote peering might have an unintended impact on anycast routing. We investigated this phenomenon by combining regular traceroutes, measurements executed with the traIXroute [111, 92] open-source tool, BGP data from RouteViews and RIPE RIS, and data from the Remote IXP Peering Observatory [73]. Our study showed that remote peering has the potential to affect 19.2% of the anycast prefixes and we confirmed via traceroute measurements that around 40% of such prefixes were indeed impacted by remote peering. We also revealed that remote peering could increase transmission latency by routing traffic to distant suboptimal anycast sites.

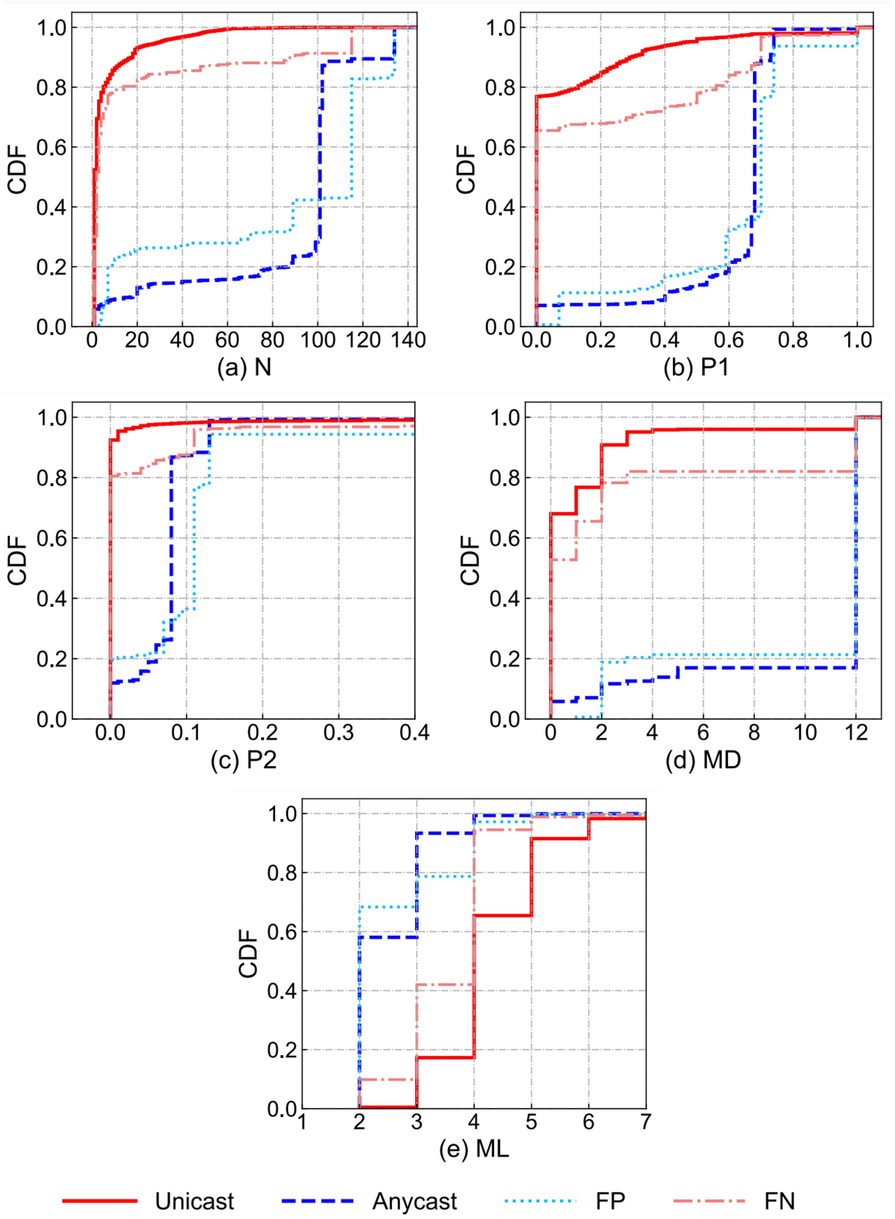


Figure 2.2: Distributions of the 5 classification features we propose for (1) anycast/unicast from the near-ground-truth dataset (§2.2.3) and (2) False Positives/False Negatives from our passive classification (§2.3)

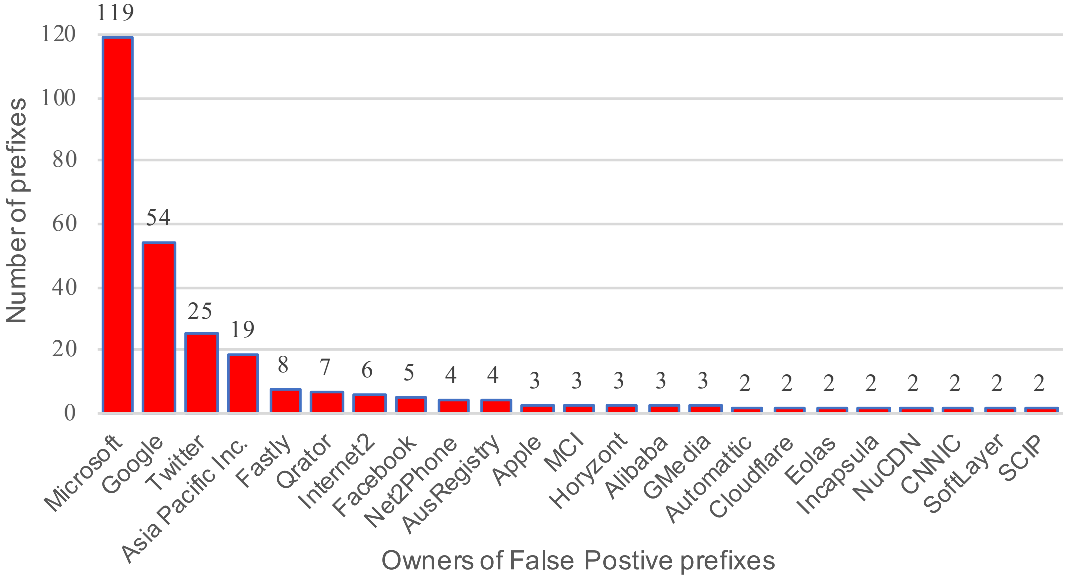


Figure 2.3: Breakdown of the Owners of False Positive Prefixes

Table 2.6: Datasets of Remote Peering. (#RP: the number of ASes involving remote peering collected from [73]; #mem-AS: the number of IXP member ASes; #RP-AS: the number of remote peering AS pairs collected from BGP information; #RP-Any: the number of anycast prefixes with remote peering AS pairs (RP-AS); %RP-Any: percentage of anycast prefixes with RP-AS in total anycast prefixes; #m-pfx: the number of anycast prefixes that include RP-AS pairs in BGP paths and that can be reached by traceroute; #v-pfx: the number of prefixes where we validated RP-AS through traceroute.)

IXP	#RP	#mem-AS	#RP-AS	#RP-Any	%RP-Any	#m-pfx	#v-pfx
AMS-IX	355	821	758	608	15.83	545	165
BIX	9	65	1	1	0.026	1	0
BIX.BG	17	79	0	0	0	-	-
CABASE[†]	15	71	0	0	0	-	-
CATNIX	9	42	7	568	14.78	568	5
DE-CIX Fr[‡]	367	826	383	520	13.53	520	182
FICIX	4	34	3	35	0.91	35	0
France-IX[♮]	118	369	147	388	10.10	326	71
HKIX	46	288	15	85	2.21	85	38
IIX	92	222	0	0	0	-	-
INEX	11	101	0	0	0	-	-
QLD-IX	4	81	2	31	0.81	31	31
IX Man[♯]	12	95	5	65	1.69	65	0
LINX LON1	151	787	224	511	13.30	511	140
LINX NoVA	9	45	5	36	0.94	36	0
LONAP	13	200	13	83	2.16	83	60
MIX-IT	49	241	26	237	6.17	237	43
NIX.CZ	32	152	17	66	1.71	66	0
SGIX	8	96	0	0	0	-	-
SIX.SK	4	57	0	0	0	-	-
SwissIX	48	185	78	135	3.51	147	91
Thinx	29	183	2	9	0.23	9	0
TPIX	33	220	0	0	0	-	-
TPIX-TW	4	41	1	6	0.16	6	0
UA-IX	38	189	0	0	0	-	-
VIX	32	140	17	97	2.52	97	30
Total[¶]	1,075	3,377	1,013	751	19.2	688	293

[†] CABASE-BUE-IX Argentina; [‡] DE-CIX Frankfurt; [♮] France-IX Paris; [♯] IX Manchester

[¶] We remove the duplicated prefixes.

Chapter 3

OPEN PROXY ECOSYSTEM ANALYSIS

Open proxies provide free relay services to users, allowing them to browse the Internet anonymously [68, 78], avoid geographic restrictions [84, 117], or circumvent censorship [40, 41, 118]. Many open proxy aggregators [17, 20, 24, 5, 23, 18, 14, 21, 12, 19, 4, 29, 15, 9, 1] collect and publish thousands of "active" open proxies each day. Those enormous numbers of proxies have formed a large and complex ecosystem. In recent years, researchers have conducted studies to explore and characterize the open proxies in various aspects, such as performance, behaviors, security, and distributions [112, 107, 82, 97, 50]. They analyzed how the proxies can modify or manipulate the requested resources, such as HTML contents, image files, and executable files. The behaviors of such modifications have been used for advertisement injection [109, 33, 36], tracking user information [70, 75], and malicious code execution [64, 108]. However, the owners of those malicious proxies and corresponding campaigns have not been well studied before. In particular, open proxy owners can deploy and manage many proxies in diverse locations at different times to enhance the effectiveness of their activities or campaigns. Also, they could change their deployment and behaviors to hide their activities and avoid being detected and blocked. Thus, a systematic investigation on how open proxies are deployed and managed on the Internet is sorely needed but still missing.

In this chapter, we perform a large-scale, comprehensive measurement-based analysis to investigate the ecosystem of open proxies. We design a measurement methodology to facilitate the analysis of massive returned responses from open proxies and accurately identify the proxies that manifest similar behaviors, possibly controlled by the same owner, to create a campaign. Moreover, to advance the understanding of the open proxy ecosystem, we study two specific groups of open proxies, the cloud-based proxies and long-term proxies. We identify and characterize the cloud-based open proxies by compiling a comprehensive list of cloud providers' IP ranges. We compare cloud-based open proxies with non-cloud-based open proxies in various ways. Open proxies are vulnerable to being abused due to their openness. As a result, typically the malicious open proxies could be quickly blacklisted [124] as their malicious behaviors are not hard to detect, and hence the lifetime of malicious open proxies is usually short. Therefore, to understand the usage and deployment of those long-term open proxies, we investigate the long-term proxies and compare them with short-term open proxies.

The three major contributions of this work are summarized as follows.

- We collect more than 436 thousand open proxies in nine months, among which we identify and measure more than 104 thousand proxies that returned responses. To the best of our knowledge, the measurement scale of our work is the largest in the studies of open proxy in terms of data collection and analysis.

- We design a lightweight method to classify these open proxies based on the Document Object Model (DOM) structure. More importantly, we attempt to parse and extract the owner information of proxies that could be inferred from the HTTP responses. Through the analysis of malicious proxy owners, we discover different malicious cases and campaigns using open proxies. We further show that some owners are changing their deployments to avoid being blocked and deploy more proxies to enhance the power of their malicious attacks.

- We present an in-depth analysis of two specific deployments of open proxies, *i.e.*, the cloud-based open proxies and long-term open proxies. We study the characteristics of cloud-based proxies, showing that the cloud-based proxies have better performance and longer lifetime than non-cloud proxies. The cloud-based proxies also have a higher percentage of unchanged proxies for providing more reliable relay services. We also examine the long-term open proxies and uncover why they can survive in the wild Internet for a long time.

The remainder of this chapter is organized as follows. We introduce the background of open proxy and survey the related work in §3.1. We present our methodology of measuring and analyzing open proxies in §3.2. We characterize the open proxy ecosystem in §3.3. In §3.4, we analyze the content modifications of open proxies and examine the owners and campaigns of malicious open proxies. Then, we study two special groups of open proxies, cloud-based proxies and long-term proxies, in §3.5 and §3.6, respectively. In §3.7, we discuss the ethical considerations and limitations of this work. Finally, we chapter the chapter in §3.8.

3.1. Background and Related Work

3.1.1 Background

A web proxy is a relay server that forwards HTTP(S) requests and returns responses between a client and a server. Generally, a web proxy allows a certain group of users to access web pages to reduce bandwidth or bypass geographic restrictions. In particular, open proxies are publicly available proxy servers that any user can use without authentication, simply configuring the corresponding IP address and port.

In many cases, open proxies can help users hide their original IP addresses to circumvent the geolocation-based restraint since the webserver can only see the open proxy's IP address. In contrast, some open proxies may reveal original IP addresses or the presence of the proxy by adding specific headers, such as `X-FORWARD-FOR` or `HTTP_VIA`.

3.1.2 Related Work

Open proxy studies. Scott *et al.* [107] studied the open proxies that expose usage statistics from open management interfaces of manager programs such as Squid and analyzed the usage, distribution, and traffic pattern of identified open proxies. Tsirantonakis *et al.* [112] presented a study focusing on content modifications in open proxies by examining and comparing the DOM structure. They analyzed multiple types of malicious behavior, such as replacing advertisements, collecting user information, and fingerprinting browsers. Furthermore, Perino *et al.* [97] built an open proxy measurement platform to examine the characteristics, behavior, performance, and usage of open proxies. Mani *et al.* [82] also explored the availability, performance, HTML manipulation, and file manipulation of open proxies and compared open proxies with Tor. Choi *et al.* [50] conducted a comparative analysis of open proxies and residential proxies. They used passive methods to study open proxies' distributions, blacklist-check results and relations with GDP, Internet freedom, *etc.*In this study, we present a more comprehensive and larger-scale study of the open proxy ecosystem. More importantly, by identifying content modifications and malicious behavior, we attempt to extract the information that can be used to infer and track the open proxy owners who possibly control a bunch of proxies. Also, we first investigate two particular types of open proxies, cloud-based and long-term proxies.

Relay system studies. CoDeen [96, 114] implemented a proxy network consisting of web cache servers deployed in PlanetLab and provided insights of the proxy system management and the analysis of unusual web traffic observed from the proxy view. Weaver *et al.* [115] proposed Netalyzr, a diagnostic tool to analyze the user's connections, and found that 14% of clients use a web proxy. Huang *et al.* [72] studied the presence of multiple types of middleboxes by leveraging the vantage points of residential IP proxy service. Mi *et al.* [85] explored the residential IP proxy ecosystem and its security and management issues.

Manipulations by middlebox. Chung *et al.* [51] detected end-to-end violations of DNS, HTTP, and HTTPS through a paid residential proxy service. They found that up to 4.8% of nodes are subject to some type of end-to-end violations. O'Neill *et al.* [94] measured the prevalence of TLS proxies using a probing tool deployed through Google AdWords campaigns. They found that 1 in 250 TLS connections are TLS-proxied and identified over 1,000 malware interceptions. Carnavalet *et al.* [57] studied TLS proxies used by antivirus and parental control applications that would be vulnerable to Man-in-the-Middle attacks. Durumeric *et al.* [61] built a heuristic to detect HTTPS interception by characterizing the TLS Handshakes of popular browsers and interception products. Their study shows that TLS interceptions drastically reduce connection security. Tyson *et al.* [113] investigated HTTP header manipulation of proxies and middleboxes and analyzed the factors affecting head manipulation. In this study, we also examine and classify content modifications by open proxies.

3.2. Methodology

To have a broad view and deep understanding of the open proxy ecosystem, we systemically collect open proxies from multiple sources and test them using a website with static content under our control. We then detect content modification by DOM tree comparison of the original content and the proxied content. By combining information extraction with manual inspection, we classify modifications into different categories and identify malicious proxy owners who control a set of proxies that share the same behavior.

3.2.1 Collecting Open Proxies

In this study, we collect more than 436,000 open proxies in total from multiple sources, including:

- Websites that collect and publish open proxies,
- Open-source tools that collect, validate, and publish available open proxies,
- Crowd-sourcing open proxy lists published by users.

The details of collection sources are listed in Table 3.1. We collect open proxy information from the above sources daily in nine months (from September 2019 to June 2020). In particular, for several sources that update their lists hourly, we crawled them every hour. We compile proxies from all sources daily and remove duplicate proxies.

Table 3.1: Sources of open proxies

Type of Sources	Source
Proxy websites	proxy-daily [17] proxylistdaily [20] smallseotools [24] dailyfreeproxy [5] sinium [23] proxy-list.download [18] openproxy.space [14] proxyserverlist24 [21] live-socks [12]
Proxy collection tools	ProxyBroker [19] Gretronger Tool [10]
Other proxy lists	clarketm [4] TheSpeedX [29] opsxcq [15] fate0 [9] a2u [1]

3.2.2 Measurement of Open Proxies

We conduct both active and passive measurements on collected open proxies to examine the open proxy ecosystem and behaviors.

Active measurement. To study the performance and behavior of open proxies, we set up two controlled websites and send HTTP/HTTPS requests to our controlled websites via each collected proxy. We simultaneously test 100 proxies and set 15 seconds timeout to filter out unresponsive or unreachable proxies. We use a server deployed in our university to issue requests to the static websites via proxies.[1] In each test, we record status code, response time (time from sending requests to receiving

[1] In this work, we deployed one vantage point in our laboratory. Based on the previous study [82], the behavior of proxies does not significantly vary with the different locations of the vantage points. We also did not observe different behaviors when utilizing additional vantage points.

responses), download time (time from sending requests to finishing download all the requested resources), HTTP response headers, and HTTP page contents. In addition, to measure the performance, we send three ping probes from the deployed server to obtain the round-trip time (RTT) and use the `curl` to download a 5 MB test file via open proxies and to measure download speed.

Passive measurement. To understand the deployment and ecosystem of open proxies, we collect different types of data through diverse sources. Open proxies may have domain names associated with their IP addresses, so we perform the reverse DNS resolution (rDNS) to acquire domain names. To explore the distribution of open proxy networks, we query the WHOIS Database for AS information. Country-level geolocation of proxy is achieved by the Maxmind database [83]. To study the cloud proxies, we manually collect IP address ranges from 31 public cloud service providers to identify the proxies deployed in cloud platforms. Finally, we identify the blacklisted open proxies by leveraging the open-source blacklist scan tool `Pydnsbl` [22] that integrates data from 53 blacklist sources.

3.2.3 Detecting Content Modification and Identifying Open Proxy Owners

We employ a similar approach to detecting and clustering the content modification as the study done by Tsirantonakis *et al.* [112]. Specifically, we extract the DOM structure of returned content from proxies and compare it with the original web page's DOM structure. Although it is straightforward and convenient to detect modification or unexpected response by DOM structure, it is challenging to process massive data from thousands of proxies with modified contents. In total, we receive 83,815 unique response contents. We observe that proxy owners can change the modified contents by injecting or replacing them with random text in contents, but their DOM structures remain the same. To facilitate data processing, we first cluster content modification proxies to groups based on their DOM structures. Overall, we identify 1,745 unique DOM structures from all collected responses. Through examination of several cases in each group, we classify the open proxies as benign or malicious. Furthermore, to

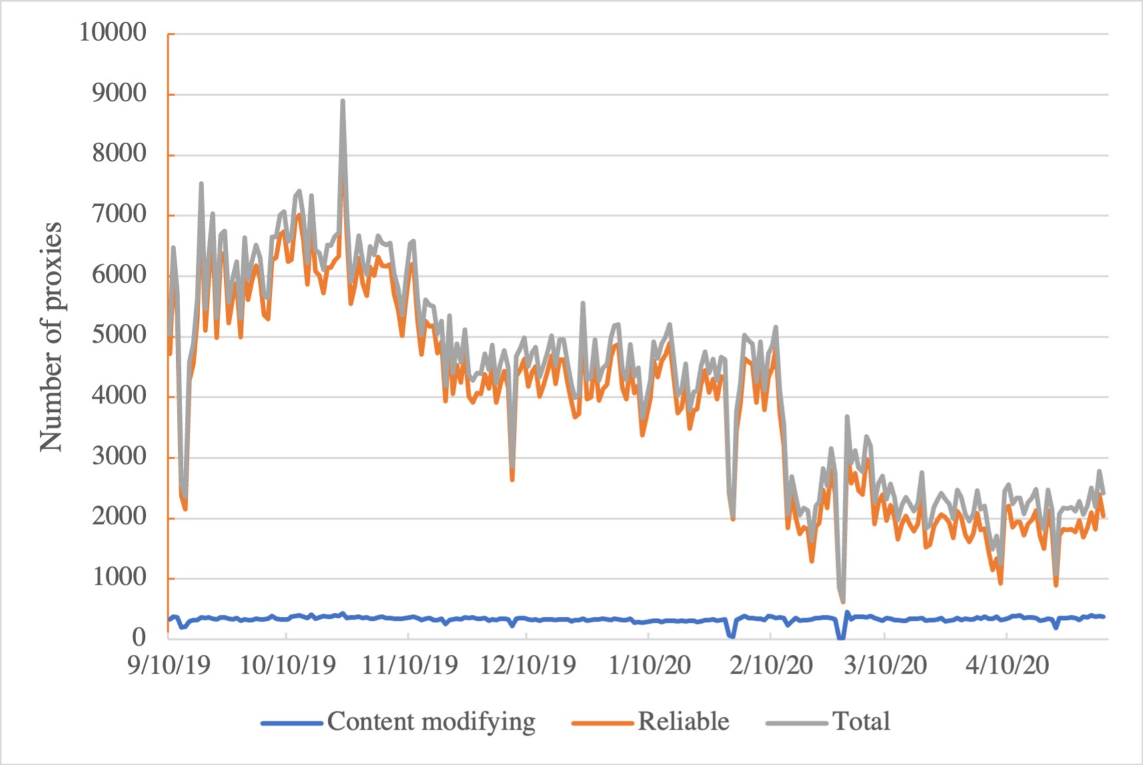


Figure 3.1: Number of daily unique open proxies

identify possible owners of open proxy groups, we parse received HTML contents and extract elements, including metadata (title, keywords, and other fields), inject library, and URLs to search for identifiers of owners.

For the obfuscated codes, we manually inspect them by using multiple methods, including Unicode decoding, Base64 decoding, function evaluation, variable evaluation, and code formatting. By combining extracted elements and manual inspection, we can classify malicious behavior and identify open proxy group owners (detailed cases examined in Section 3.4.2).

3.3. Overview of Open Proxy Characterization

In this section, we characterize the open proxy ecosystem. First, we present the network distribution and geographic distribution of open proxies. Next, we study the reliability and performance of *responsive* proxies. For content modifications and malicious owners, we present details in Section 3.4.

Table 3.2: Port distributions of collected and responsive proxies

All Proxies			Responsive Proxies		
Port	#	%	Port	#	%
9999	96,802	22.18%	9999	34,599	33.23%
8080	74,072	16.97%	8080	27,348	26.27%
4145	47,543	10.89%	3128	7,534	7.24%
3128	18,988	4.35%	80	5,767	5.53%
1080	17,746	4.06%	8118	2,050	1.97%
80	13,580	3.11%	53281	1,256	1.21%
38801	10,514	2.41%	8888	1,077	1.03%
9000	9,303	2.13%	8213	1,025	0.98%
8118	8,817	2.02%	3129	907	0.87%
8888	4,158	0.95%	999	809	0.78%
All others	134,928	30.91%	All others	21,742	20.88%

Daily statistics of proxies. The number of unique proxies (content modifying, reliable and total responsive proxies), over time, is shown in Figure 3.1. The median number of daily reliable proxies is 4,141.5, with a range of [622, 8,473]. The median number of daily content modifying proxies is 337, with a range of [18, 452]. The responsive proxies include reliable and content modifying proxies. The median number of daily total responsive proxies is 4,461.5, with a range of [640, 8,899]. With our nine-month collections and testing, we collect 436,451 unique proxies and 104,114 responsive proxies (23.97% of collected proxies).

Port Distribution. The port distributions of collected and responsive proxies are shown in Table 3.2. Port 9999, 8080, 3128, 80, and 8118 are the most popular ports in open proxies. In collected proxies, there are 14,239 proxies (3.26%) found to use multiple different ports. In responsive proxies, there are 4,677 proxies (4.49%) found to use multiple different ports. One proxy is found to use 403 different ports in total during the nine months. Those observations demonstrate that open proxy owners may often change the web proxy port. The reason might be that switching ports can protect the proxy server as malicious users cannot easily leverage the proxy servers for malicious purpose.

Domain names. There are 130,435 unique domain names of collected proxies and 32,409 unique domain names of responsive proxies. The most common domain names resolved from the IP addresses of collected and responsive proxies are shown in Table 3.3.

More than 60% of reverse DNS lookup results is `NXDOMAIN`, which means those proxies do not have domain names. Because users only need the IP address and port to use open proxies, it is reasonable that open proxies do not possess domain names necessarily. In addition, we manually inspect other popular names associated with open proxies and find that many of them have been noticed by their abnormal behaviors.

`hn.kd.ny.adsl` often changed its matching IP address and those IP address belong to China Unicom. This domain name is reported to perform repetitive port scans and blind SQL injections [13, 3, 25, 28, 27, 7]. In addition, because we use reverse DNS lookup to find the domain name of open proxies, the returned results might not be the real domain names of open proxies. `hn.kd.ny.adsl` is not a valid fully qualified domain name (FQDN), and we speculate that it is an internal domain name leaked to the public.

`azteca-comunicaciones.com` is the domain name of a Columbia communication company – Azteca Comunicaciones. It also has been found to be mapped to many IP addresses and those IP addresses are identified as open proxies and spammers [16, 11, 6].

`static.vnpt.vn` matches multiple IP addresses and all of them belong to Viet-Nam Data Communication Company. This domain name is reported to send spams through different IP addresses [8, 26, 2, 30].

Geolocation. The geolocation information of collected open proxies is shown in Table 3.4 and Figure 3.2. The collected proxies are located in 172 countries, and the geographic distributions are skewed that over 80% of open proxies are located in 10 countries. China, Thailand, United States, Brazil, India, and Indonesia have the most

Table 3.3: Domain name distributions of collected/responsive proxies

Domain name	Count	Percentage
All proxies		
NXDOMAIN	229,481	63.07%
hn.kd.ny.adsl.	1,078	0.3%
azteca-comunicaciones.com.	325	0.09%
static.vnpt.vn.	220	0.06%
int0.client.access.fanaptelecom.net.	164	0.05%
All others	132,255	36.43 %
Responsive proxies		
NXDOMAIN	60,906	64.57%
azteca-comunicaciones.com.	177	0.19%
hn.kd.ny.adsl.	111	0.12%
static.vnpt.vn.	82	0.09%
customer.worldstream.nl.	52	0.06%
All others	32,859	34.97%

collected open proxies and responsive proxies.

Cloud. In collected proxies, 18,005 proxies (4.13%) are hosted on the public cloud platform. In responsive proxies, 5,637 proxies (5.41%) are hosted on public cloud platforms. The details of the cloud-based open proxy study are presented in Section 3.5.

Prefix. There are 100,410 unique /24 prefixes in collected proxies and 33,783 /24 prefixes in responsive proxies. The Top 10 prefixes with the most collected and responsive proxies are shown in Table 3.5. The results show that in some prefixes, most servers are deployed as open proxies. Those open proxies may be deployed by the same owner. The details of ownership are discussed in section 3.4.

Autonomous System (AS). The collected proxies reside in 9,060 ASes, and responsive proxies reside in 5,282 ASes. The most popular ASes for collected and responsive proxies are shown in Table 3.8. The distributions of AS are also significantly unbalanced, where more than half of open proxies reside in only ten ASes. Most of these ASes belong to telecommunication and Internet companies that provide server hosting

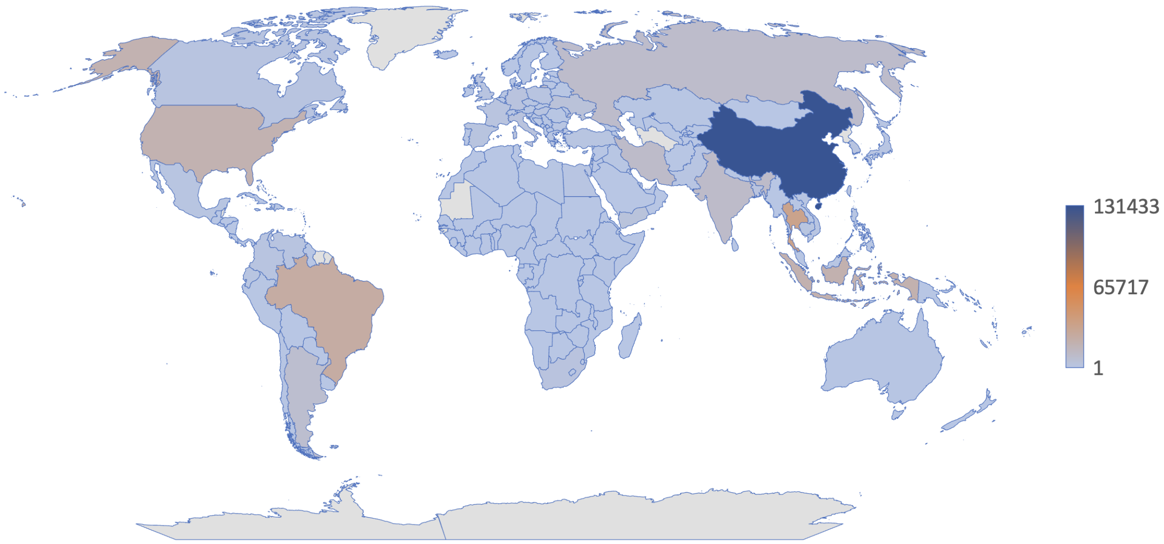


Figure 3.2: Geo-distribution of open proxies.

services.

Blacklist. The open-source blacklist scan tool `Pydnsbl` [22] that integrates data from 53 sources is used to extract open proxies being blacklisted. In collected proxies, 272,719 proxies (62.48%) appear in at least one blacklist, 163,732 proxies are not on any blacklist. In responsive proxies, 70,122 proxies (67.35%) appear in at least one blacklist, 33,992 proxies are not found on blacklists. The high percentage shows that most open proxies may have performed suspicious or malicious activities.

Behavior. The content modification results are shown in Table 3.6. We identify that 92.73% of proxies always returned the expected response all the time. This result shows that most of the working open proxies are reliable. In the meantime, 6.04% of proxies consistently perform the content modification. Interestingly, 1.23% of proxies change their behaviors from time to time. The owners of these proxies may change their behavior by purpose to hide their malicious activities and avoid being detected. We describe a detailed analysis of content modifications in section 3.4.

Lifetime. Here, we further define the open proxies which consistently return unchanged content as reliable proxies. The CDFs and boxplots of proxy lifetimes (responsive, reliable, and content modification proxies) are shown in Fig 3.3. The average

Table 3.4: Geolocation of collected and responsive proxies

All Proxies		Responsive Proxies	
Country	%	Country	%
China	41.92%	China	38.15%
Thailand	8.70%	Thailand	8.56%
United States	7.32%	Indonesia	7.89%
Brazil	6.14%	United States	6.90%
Indonesia	5.76%	India	5.04%
India	3.21%	Brazil	4.88%
Iran	3.03%	Russia	3.20%
Russia	2.77%	Iran	1.36%
Argentina	2.02%	Singapore	1.15%
Ukraine	1.20%	Bangladesh	1.14%
All others	17.92%	All others	21.69%

lifetime of responsive, reliable, and content modification proxies are shown in Table 3.7. We observed that nearly 80% of proxies' lifetime is one week or less. Content modification proxies' lifetime is slightly longer than reliable proxies, which means content modification proxies are more resistant than reliable proxies. The detailed discussions about content modification proxies are presented in Section 3.4, and we discuss long-term proxy in Section 3.6.

Performance. The CDFs and boxplots and of responsive, reliable, and content modification proxies' RTTs and download speed are shown in Fig 3.4 and Fig 3.5. The performance of responsive, reliable, and content modification proxies is shown in Table 3.7. The figures and table show that reliable proxies have better performance than content modification proxies, with shorter RTTs and faster download speed.

Summary. In this section, we characterize open proxies from multiple aspects. We present network distributions (port, domain name, and AS), geographic distribution, lifetime, performance (RTT and download speed), and reliability (blacklist check and content modification). Moreover, we observed that the majority of open proxies are concentrated in a small set of AS and countries. The lifetime of open proxies is very

Table 3.5: Top 10 prefixes with most collected and responsive proxies

All Proxies		Responsive Proxies	
Prefix	IP#	Prefix	IP#
123.163.27.0/24	252	123.163.27.0/24	234
111.72.25.0/24	250	123.160.1.0/24	218
123.160.1.0/24	250	123.163.122.0/24	212
123.149.141.0/24	249	123.163.97.0/24	209
123.149.136.0/24	245	163.204.246.0/24	208
111.79.44.0/24	244	60.13.42.0/24	205
223.199.18.0/24	242	163.204.243.0/24	205
1.196.177.0/24	242	163.204.241.0/24	201
123.149.137.0/24	241	163.204.242.0/24	200
111.79.45.0/24	239	163.204.244.0/24	200

Table 3.6: Content Modifications of Proxies

Behavior	# Proxy	Percentage
Always modify	6,326	6.04%
Never modify	97,074	92.73%
Sometimes modify	1,287	1.23%

short that most proxies can not live up to one week. Two-thirds of open proxies are listed in blacklists, and 7.31% of open proxies returned modified contents.

3.4. Content Modification and Malicious Open Proxy Owners

In this section, we identify the behaviors of malicious and benign proxies and present detailed case studies to explore the owners who deploy the malicious proxies and how the proxy owners can benefit from the campaigns using open proxies.

3.4.1 Content Modification

Since we received thousands of responses via proxies every day, it is challenging to process and analyze such massive data. To reduce manual effort and simplify the analysis, we utilize the DOM structure to analyze the contents. To do so, we

Table 3.7: Lifetime and performance of proxies

Average	Responsive	Reliable	Modifying
Lifetime (days)	9.45	9.37	10.89
response time (s)	4.99	5.24	1.95
Download time (s)	5.12	5.37	2.04
RTT (ms)	233.24	231.7	250.78
Download speed (KBps)	254.43	271.07	57.47

simply record the tag names and locations of each HTML contents. If there are different tag names or locations between two HTML pages, we consider those two DOM structures are different. In total, we identified 1,745 unique DOM structures of all collected responses. Next, we select representative cases to classify proxies. We parse the HTML contents to extract proxy activity information to understand each proxy group's behavior and nature.

By combining extracted information with the manual examination, we classify the content modification proxies as benign or malicious. We consider the following scenarios with content modification proxies as benign:

- Lack of permission: access is denied due to no proper permission;
- Errors: that category includes network errors like DNS errors and configuration errors;
- Misclassification: incorrectly labeled as open proxies by open proxy collecting source;
- Blocked by network management software or AntiBot software, probably due to a restricted access policy.

Then, we identify the following cases of content modifications as the misbehavior of malicious proxies:

Table 3.8: Most popular ASes for collected and responsive proxies

	All Proxies	
ASN	Organizations	Percentage
4134	No.31,Jin-rong Street	24.89%
37963	Alibaba Advertising Co.,Ltd.	8.87%
4837	China Unicom China169 Backbone	5.61%
23969	TOT Public Company Limited	3.37%
7713	PT Telekomunikasi Indonesia	3.01%
14061	DigitalOcean, LLC	2.43%
45758	Triple T Internet/Triple T Broadband	2.24%
131090	CAT TELECOM Public Company Ltd	1.38%
16276	OVH	1.17%
17552	True Internet Co.,Ltd.	1.06%
	All others	45.99%
	Responsive Proxies	
ASN	Organizations	Percentage
4134	No.31,Jin-rong Street	27.9%
4837	China Unicom China169 Backbone	5.87%
14061	DigitalOcean, LLC	3.43%
45758	Triple T Internet/Triple T Broadband	3.36%
7713	PT Telekomunikasi Indonesia	3.03%
23969	TOT Public Company Limited	2.22%
17816	China Unicom IP network China169	2.09%
17552	True Internet Co.,Ltd.	1.24%
20473	Choopa, LLC	1.12%
17451	Biznet Networks	1.12%
	All others	48.61%

- Replacing original content: such proxies replace the static content in our original server and lead the user to other websites (shopping, adult, and news website) or applications;

- Ad injection: this type of proxies inject advertisement JavaScript to the original contents;

- CSS injection: these proxies inject the suspicious CSS file;

- Redirection: these proxies redirect users to other websites;

Table 3.9: Categories of content modification proxies

	Category	**# Proxy**	**Percentage**
Benign (23.58%)	Lack of permission	1,234	16.52%
	Error	112	1.50%
	Misclassification	366	4.90%
	Blocked	49	0.66%
Malicious (76.42%)	Replacement	466	6.24%
	Ad injection	2,393	32.04%
	CSS injection	9	0.12%
	Redirection	2,748	36.80%
	Collect user information	96	1.23%
	Cryptojacking	19	0.25%

- Collecting user information: these proxies inject scripts to obtain user information like Operating System, browser, and cookie;

- Cryptojacking: these proxies inject cryptocurrency mining scripts that take advantage of the user's resource to mine digital currency by stealth.

The categories of benign and malicious proxies are shown in Table 3.9. We identified 23.58% of content modifications are benign, and the majority of them are due to lack of permission or misclassification. The possible reason is that open proxy collectors did not validate the nature and availability of collected proxies and public them incorrectly in open proxy lists to the Internet. Malicious proxies occupy 76.42% of content modification proxies. Most of the malicious proxies belong to two categories — Ad injection and redirection. In addition, we find 19 proxies performing cryptojacking attacks.

3.4.2 Malicious Open Proxy Owners: Case Studies

Open proxies offer service for users free of charge, but the deployment is not free for owners. To understand the purpose and benefit of deploying open proxies, we attempt to identify and track the open proxy owners by information parsed from modified contents as we find that some proxy owners typically deploy and control a set of proxies that perform the same modifications. In this part, we discuss several case studies to demonstrate the purposes and deployment of open proxies by means of their owners.

ISP injection. Many open proxies inject similar JavaScript code snippet to display advertisements or collect user's information for censorship. They obtain user's information including domain name, screen width and height, and other parameters like id, enc, params, and idc_r. These proxies label users by allocating different parameters like id and enc. These pieces of information are concatenated to two common URLs (‘`notifa.info`’ and ‘`cfs.uzone.id`’) and then sent back. The example of injected code by ISP is shown in Fig. 3.6.

In total, we identified 6,572 such responses from 2,107 proxies observed in 237 days. The most proxies observed in one day are 86 proxies, and the average proxies observed in one day is 27.73. The lifetime range of those proxies is from 1 day to 102 days. This group of proxies belong to 43 ASes, all located in Indonesia. Indonesia ISP hosts these proxies to sell the ads and censor the traffics. Even though it is not clear if they are illegal, it is better to avoid using these proxies to protect users' privacy.

Cloud provider advertisement. We identified a group of proxies that inject JavaScript codes to provide Chinese cloud provider advertisements, called `Ruijieyun`, a cloud platform for marketing and third-party payment. The scripts will detect the user's IP address and determine if the user's IP is in their IP address ranges. If so, they will not provide ads, while if not, they will pop up ads to promote their cloud service. That strategy can enhance the ads' effectiveness to make ads only propagate among new users. In total, we received 4,067 responses from 420 proxies observed in 221 days. The maximum number of such proxies observed in one day is eighty. The average of such proxies observed in one day is 18.40, with the lifetime ranging from 1 day to 73 days. They reside in 14 Chinese ASes that all belong to Chinese telecommunication Companies. This case shows that `Ruijieyun` cloud service company deploys open proxies in multiple China telecommunication company networks to broadcast its advertisements for attracting new users to utilize its cloud service.

User network information collection. We observed that a group of proxies inject similar JavaScript code in the headers. These injections do not change the original contents but prompt users to send requests to the Google Analytics website with specific parameters:

```
https://www.google-analytics.com/collect?v=1&t=pageview&
tid=UAXXXXXXXXXXXX&dh=test777.com&cid=XXXXX&dp=/mp/ping/
```

The actual user ID is marked here to protect privacy. We parse the URL and parameters based on the references of Google Analytics. We focus on four key parameters: `tid, dh, cid`, and `dp`. The `tid` is tracking ID or web property ID that is associated with collected data. The `dh` is document hostname that specifies the hostname from which content was hosted. The `cid` is client ID that is used to identify a particular user, device, or browser instance. The `dp` is document path which is the path portion of the page URL. In the collected data, `tid`, `dh` and `dp` are identical in this open proxy group, while the `cid` is changed in each request. All the cases share the same tracking ID, which indicates that all collected data associates with the same owner. In addition, this owner collected user information that is hosted in one particular hostname (`test777.com`) and document path. We visited this website to explore the owners' purpose and found a Japanese research website for network and hardware experimentation. It has stopped updating since 2006. We notice the document path is named as 'ping', which could imply PING probing measurements. We speculate that this owner collected network measurement data from users by injecting JavaScript code. In total, we observed 338 responses from 54 proxies. Those proxies are located in 11 countries in Europe, Asia, and North America, indicating that the owner has deployed a large number of widely distributed proxies to obtain a large amount of measurement data. However, we argue that the owners should well inform users of the measurement content and obtain users' consent to conduct measurements in such a large-scale experiment. Also, the experiment code should be cleared up if proxy owners discontinue the measurement.

Cryptojacking. We identified a group of open proxies performing cryptojacking. The contents they returned look like a regular login page of an online forum that requests a username and password. Meanwhile, they inject JavaScript code that pops out a YouTube video while using the user's processor to mine cryptocurrency without permission or notification. The screenshots of the cryptojacking page are presented in Fig 3.7. By carefully inspecting the injected codes, we find that all the mining scripts contain the same identifier (*i.e.*, a wallet ID), which means all the mined cryptocurrencies will benefit the same owner. Hence, we infer that this owner deploys or rents multiple malicious proxies to enhance his/her mining capacity and obtain profit. In total, we received 1,416 responses from 19 proxies observed in 106 days. The maximum number of such proxies observed in one day is nineteen. The average proxies observed in one day is 12.91. The most extended lifetime of them is 94 days, and the shortest lifetime is 38 days. The owners choose to use different proxies and change the number of proxies to avoid being detected and blocked. Also, 18 out of 19 observed proxies are hosted in AS 14061 Digital Ocean, a popular global cloud infrastructure provider, while one is in Hetzner – a German Internet hosting company. These proxies are distributed in seven countries in North America, Asia, and Europe. This type of malicious proxies could cause considerable damage to users because if users do not notice this video and leave this video open, these malicious proxies can take advantage of users' processors to mine cryptocurrency for a long time.

Ad injection campaigns. One group of proxies returned a mobile news application called Orange News – a Hong Kong news application. Some proxies will provide business websites such as `Early Bird Cashflow`, which provides cash flow service, and `DragonEX` which offers digital currency trade and exchange service. In another case, proxies return a web game called `Tank Rumble` that users can use mouse and keyboards to control the tank to attack enemies. Another returned questionable content is an education website that provides an English training program called `Cambridge English`. Another application the proxies returned is a game communication app called `Nadeko`. No matter whether these websites own those proxies, it is reasonable to infer that those proxies' owners can obtain profit by redirecting users to their desired websites.

In this section, we first categorize open proxies based on the content modification behaviors. About 23.58% of open proxies that modify contents are benign, and most of them fall into the categories of the lack of permission and mis-classification. Then, we focus on the malicious open proxies that occupy 76.42% of content modification proxies. We conduct detailed case studies to thoroughly analyze the malicious open proxies' behaviors and deployments to explore proxy owners' purposes. The owners may achieve monetization from proxy users by injecting advertisements, collecting user information, replacing original content with applications and websites, and mining cryptocurrency. These proxy owners use open proxies to expand their influences and gain profits from numerous users.

3.5. Cloud-based Open Proxy

Cloud service has quickly grown in recent years, with 84% of organizations now using cloud services, up from a mere 48% five years earlier. In this section, we study open proxies hosted in the cloud, and we refer to them as cloud-based proxies. To identify the proxies hosted in cloud platforms, we first collect popular cloud providers' public IP address ranges from their official websites. In total, 31 cloud providers' public IP address ranges are collected. For large cloud providers like Amazon, Microsoft, and Google, we also collected the IP ranges of their regions. In total, we found 1,733 cloud regions and 29,632 cloud IP address blocks. Then we verify whether responsive proxy IP addresses are in the cloud IP blocks and collect cloud-based proxies. There are 5,637 responsive proxies in 57 cloud regions. The top 25 Cloud regions that contain proxies are shown in Fig 3.8. Most cloud-based proxies belong to Digital Ocean, Google Cloud,

Table 3.10: Top 10 cloud regions with most responsive proxies

cloud region	proxy#	Percentage
DigitalOcean, LLC	3378	59.93%
Google cloud	432	7.66%
Azure	295	5.23%
AWS-AMAZON-us-east-1	202	3.58%
AWS-AMAZON-us-west-2	144	2.55%
ALICLOUD-HK	136	2.41%
Digital Ocean, Inc.	117	2.08%
AWS-AMAZON-ap-southeast-1	105	1.86%
AWS-AMAZON-us-east-2	95	1.69%
AWS-AMAZON-eu-west-1	91	1.61%
All others	642	11.39%

Azure, and Amazon. Nearly 90% of cloud-based proxies belong to the top 10 cloud regions as shown in Table 3.10.

Port. The Top 10 ports of cloud-based proxy and non-cloud-based proxy are listed in Table 3.11. Port 3128 is the most popular in cloud-based proxies–almost one-third of cloud-based proxies use port 3128. However, in non-cloud-based proxies, only 5.79% of proxies use port 3128. Squid proxy is a widely used free Linux proxy software, and its default port is 3128. Hence we can speculate that squid proxy software is a highly prevalent proxy software in the cloud.

Autonomous System. The top 10 AS of cloud-based proxy and non-cloud-based proxy are shown in Table 3.12. Most cloud-based proxies are hosted in American cloud providers like Digital Ocean, Amazon, and Google. In contrast, most non-cloud-based proxies are hosted in Asian Telecommunication Companies like China Telecom, China Unicom, Thailand Triple T Internet/Triple T Broadband, and Indonesia PT Telkom.

Geolocation. We present the top 10 countries of cloud-based proxy and non-cloud-based proxy in Table 3.13. Most cloud-based proxies are located in developed countries such as the US, Singapore, and the UK. Most of them are in North America and Western Europe. Besides, most non-cloud-based proxies are located in developing

Table 3.11: Top 10 ports of cloud-based proxy and non-cloud-based proxy

cloud-based proxy		Non-cloud-based proxy	
Port	Percentage	Port	Percentage
3128	32.46%	9999	35.12%
8080	24.52%	8080	26.37%
80	19.78%	3128	5.79%
8118	5.94%	80	4.72%
8888	5.11%	8118	1.74%
8000	2.31%	53281	1.26%
1080	1.72%	8213	1.04%
44344	1.54%	3129	0.91%
44321	0.98%	999	0.82%
6666	0.64%	8888	0.80%

areas such as Asia and South America. That is perhaps because the Cloud services are more prevalent and available in developed countries than developing countries, and open proxy owners can quickly and inexpensively deploy their open proxy servers on the cloud. For developing countries, the cloud service is not widely available, and the price is relatively high, so open proxy owners unlikely to use the cloud services to deploy proxies. Also, more than 40% of non-cloud-based proxies are in China, and the main reason is likely that Chinese users may utilize open proxies to circumvent censorship.

Blacklist. The results of blacklist check for cloud-based proxy and non-cloud-based proxy are shown in Table 3.14. The percentage of proxies found in the blacklist is quite different: 69.39% of non-cloud-based proxies are found in the blacklists, while only 31.81% of cloud-based proxies blacklisted. The possible reasons for fewer proxies found in blacklist in the cloud are 1) cloud-based proxies are managed and monitored by cloud service providers, so they will be detected and blocked if they violate cloud service's policies; 2) the cloud-based proxies could be more dynamic than non-cloud-based proxies due to the elastic resource provision of cloud services, and blacklists are limited to detect such dynamic cloud IP addresses.

Behavior. Content modifications of cloud-based proxy and non-cloud-based proxy

Table 3.12: Top 5 ASes of cloud-based proxy and non-cloud-based proxy

ASN	AS	Percentage
	cloud-based proxy	
14061	DigitalOcean, LLC	63.31%
16509	Amazon.com, Inc.	13.84%
15169	Google LLC	7.45%
8075	Microsoft Corporation	5.23%
45102	Hangzhou Alibaba Advertising Co.,Ltd.	4.38%
	Non-cloud-based proxy	
4134	No.31,Jin-rong Street	29.50%
4837	China Unicom China169 Backbone	6.21%
45758	Triple T Internet/Triple T Broadband	3.55%
7713	PT Telekomunikasi Indonesia	3.21%
23969	TOT Public Company Limited	2.35%

are shown in Table 3.15. The percentage of proxies that constantly modify the contents of the cloud-based proxy (2.89%) is lower than that of non-cloud-based proxy (6.12%). Interestingly, the percentage of proxies that intermittently modify the contents of the cloud-based proxy (1.44%) is slightly higher than that of non-cloud-based proxy (1.22%). Due to the cloud's dynamic and elasticity, open proxy owners can easily manage and change the proxy settings and configurations, so they may adjust their policies to modify or just forward the contents. By combining the blacklist and behavior results, we can see that cloud-based proxies have better reliability than non-cloud-based proxies.

Lifetime. The CDF of cloud-based proxies and non-cloud-based proxies' lifetime is shown in Fig 3.9. The average lifetime of cloud-based proxies and non-cloud-based proxies are shown in Table 3.16. We can see that most cloud-based proxies have a longer lifetime (14.19 days) than non-cloud-based proxies (9.17 days). Cloud infrastructures can provide more protection so that cloud-based proxies are more resistant than non-cloud-based proxies.

Performance. The CDF of cloud-based proxies and non-cloud-based proxies' RTT

Table 3.13: Top 10 countries hosting cloud-based and non-cloud-based proxies

cloud-based proxy		Non-cloud-based proxy	
Country	Percentage	Country	Percentage
United States	59.89%	China	40.34%
Singapore	13.22%	Thailand	9.05%
Canada	5.57%	Indonesia	8.35%
United Kingdom	3.73%	India	5.20%
Netherlands	3.39%	Brazil	5.07%
Germany	2.77%	United States	3.87%
India	2.25%	Russia	3.39%
Ireland	1.84%	Iran	1.44%
Brazil	1.67%	Bangladesh	1.21%
Hong Kong	1.51%	Argentina	1.16%

Table 3.14: Blacklist check results of cloud-based and non-cloud-based proxies

	cloud-based proxy		Non-cloud-based proxy	
	#proxy	Percentage	#proxy	Percentage
in BL	1,793	31.81%	68,329	69.39%
not in BL	3,844	68.19%	30,148	30.61%

and download speed are presented in Fig 3.10 and Fig 3.11. The performance of cloud-based proxies and non-cloud-based proxies are shown in Table 3.16. These comparisons show that cloud-based proxies have better performance than non-cloud-based proxies. Cloud-based proxies have shorter RTT – near half of the non-cloud-based proxies' RTT, and cloud-based proxies have faster download speed – more than four times of non-cloud-based proxies' download speed. Typically, cloud service can provide better performance than traditional servers, so cloud-based proxies have better performance due to this reason.

In this section, we study a specific type of open proxies – the cloud-based proxies. We present cloud-based proxies' network and geographic distribution, behavior, and performance, and compare them with non-cloud-based proxies. We analyze the reasons

Table 3.15: Content modifications by cloud-based and non-cloud-based proxies

	cloud-based proxy		Non-cloud proxy	
	#proxy	Perc.	#proxy	Perc.
always modify	163	2.89%	6,023	6.12%
never modify	5,393	95.67%	91,248	92.66%
sometimes modify	81	1.44%	1,206	1.22%

Table 3.16: Lifetime and performance of cloud-based and non-cloud-based proxies

Average	Cloud-based	Non-cloud
lifetime (days)	14.19	9.17
response time (s)	4.28	5.04
download time (s)	4.31	5.18
RTT (ms)	129.3	238.83
download speed (KBps)	811.93	195.65

causing the differences between cloud and non-cloud-based proxies. Even though the scale of cloud-based proxies is smaller than that of non-cloud-based proxies, cloud-based proxies have multiple advantages such as higher reliability and better performance over non-cloud-based proxies. Also, proxy owners can take advantage of the cloud to change the proxy's behavior and make cloud-based proxies more dynamic.

3.6. Long-Term Open Proxy

It is easy and convenient to use open proxy since the proxy setting is simple (only enter the IP address and port) without authentication and free of charge. On the other hand, open proxies make it easier for miscreants to launch a variety of attacks. Hence, open proxies are vulnerable to be attacked and abused. To this end, open proxies' lifetime is relatively short. In our study, the average lifetime is 9.45 days. 53.93% of responsive proxies' lifetime is two days or less, and 80.92% of responsive proxies' lifetime is short than ten days. Only 0.20% responsive proxies' lifetime is more than two hundred days. Here, we examine the *long-term* proxies whose lifetime are equal

Table 3.17: Top 10 port of long-term proxy and short-term proxy

Long-term proxy		Short-term proxy	
Port	Perc.	Port	Perc.
8080	36.02%	9999	40.95%
3128	32.23%	8080	24.52%
80	25.59%	3128	6.82%
1080	1.90%	80	4.53%
8888	1.42%	8118	2.21%
81	0.47%	8213	1.20%
443	0.47%	8888	1.00%
808	0.47%	3129	1.00%
2222	0.47%	53281	0.82%
8811	0.47%	999	0.68%

and longer than two hundred days and compare them to relatively *short-term* proxies whose lifetime is less than ten days. In this section, we examine the characteristics of long-term open proxies and explore how and why they exist for quite a long time.

Port. The top 10 port of long-term proxy and short-term proxy are listed in Table 3.17. Port 8080 and Port 3128 are the most popular ports in long-term proxy – more than 68% of long-term proxies use port 8080 and 3128. However, in short-term proxies, port 9999 is dominant that contains 40.95% of them. The differences show the long-term proxies may use different proxy applications.

Autonomous System. The top 5 ASes of the long-term and short-term proxies are shown in Table 3.18. Most long-term proxies are hosted in Digital Ocean. In contrast, most short-term proxies are hosted in telecommunication networks like China Telecom, China Unicom, Thailand Triple T Internet, and Indonesia PT Telkom. As the discussion in section 3.5, cloud services provide elastic resources and more protections so that it is reasonable that most long-term proxies are deployed in ASes belonging to cloud platforms. We identify that 64.45% of long-term proxies are host in the cloud, while only 4.91% of short-term proxies are hosted in the cloud. The possible reason is that cloud service provides more reliable and resistant service to host proxy servers so

Table 3.18: Top 5 ASes of long-term and short-term proxies

ASN	AS	Percentage
	Long-term proxies	
14061	Digital Ocean	61.14%
39832	Opera Software	5.69%
24940	Hetzner Online GmbH	5.21%
16509	Amazon.com, Inc	2.37%
37963	Hangzhou Alibaba Advertising Co.,Ltd.	1.90%
	Short-term proxies	
4134	No.31,Jin-rong Street	34.14%
4837	China Unicom China169 Backbone	7.10%
45758	Triple T Internet/Triple T Broadband	4.04%
7713	PT Telekomunikasi Indonesia	3.65%
14061	DigitalOcean, LLC	3.00%

that long-term proxies contain a higher percentage of cloud proxies.

Geolocation. We present the top 10 countries of long-term proxy and short-term proxy in Table 3.19. Most long-term proxies are distributed in developed countries such as the US, Germany, and the Netherlands. Most of them are in North America and Western Europe. Besides, most short-term proxies locate in less developing countries like China, Thailand, and Indonesia. The reason might be that open proxies in developing countries are more vulnerable due to strict control, including filtering and censorship.

Blacklist. The blacklist check results of long-term proxy and short-term proxy are shown in Table 3.20. The percentage of proxies found in blacklists is quite different: 69.05% of short-term proxies are blacklisted, while only 18.01% of long-term proxies are included by those blacklists.

Behavior. Content modifications of long-term proxy and short-term proxy are presented in Table 3.21. Even though the modification rate of long-term proxy is higher than short-term proxies, after analyzing the categorizes of behaviors, we find that 95%

Table 3.19: Top 10 countries of long-term proxy and short-term proxy

Long-term Proxy		Short-term Proxy	
Country	Percentage	Country	Percentage
United States	36.49%	China	45.74%
Germany	9.95%	Thailand	9.53%
Netherlands	9.95%	Indonesia	7.21%
India	8.06%	United States	6.63%
Singapore	7.11%	India	4.24%
Canada	6.16%	Brazil	4.21%
United Kingdom	5.21%	Russia	2.32%
China	4.74%	Iran	1.35%
Russia	2.84%	Singapore	1.18%
Iran	1.42%	France	0.85%

Table 3.20: Blacklist results of long-term proxy and short-term proxy

	long-term proxy	short-term proxy
in blacklist	18.01%	69.05%
not in blacklist	81.99%	30.95%

of modifications are benign. Most of them are due to misclassification and misconfiguration. Interestingly, we observe that the behaviors of long-term proxies are quite consistent: they either always perform the content modification or never do it. No long-term proxies are found to intermittently modify the content.

Performance. The performance of long-term proxies and short-term proxies is shown in Table 3.22. The long-term proxies demonstrate clearly better performance than short-term proxies. Long-term proxies' RTT is about half of the short-term's proxies, and Long-term proxies' download speed is nearly four times of short-term proxies.

In this section, we compare long-term proxies with short-term proxies from different aspects. Our analysis shows that long-term proxies have better performance than short-term proxies. The reasons why long-term proxies can exist for a long time are (1) they are well managed by excellent hosting providers; (2) they are misclassified

Table 3.21: Content modifications of long-term and short-term proxy

	Long-Term	Short-Term
Always modify	32.70%	6.20%
Never modify	67.30%	92.63%
Sometimes modify	0.00%	1.17%

Table 3.22: Performance of long-term and short-term proxy

Average	Long-Term	Short-Term
Response time (s)	0.85	4.68
Download time (s)	0.86	4.82
RTT (ms)	119.56	238.06
Download speed (KBps)	901.56	238.04

by proxy collectors for a long time, but proxy collectors falsely publish them. 3) owners accidentally misconfigured such proxies to be open to any user and owners doesn't notice that and remedy them.

3.7. Discussion

3.7.1 Ethical Considerations

In this study, we collect open proxies from published open proxy lists. We do not utilize large scale port scanning to detect open proxies. Thus, the normal usage of open proxies is not affected, and private proxies are not exposed. In addition, open proxies are used to access our designed static websites that do not cause any harm to open proxies. The collected data does not include any open proxy owners' and other users' personal and private information. In summary, this study does not bring any risk and damage to proxy owners and users.

3.7.2 Limitations

We share similar approaches with earlier research to detect content modification, which cannot determine if the behavior-changing proxies have a hidden malicious

Table 3.23: Study content of open proxy studies (Cloud: Cloud-based proxies, Long:Long-term proxies, Blacklist: Blacklist check)

Studies	Cloud	Long	Content manipulations	Owner study	Blacklist
Scott [107]	×	×	×	✓	×
Tsirantonakis [112]	×	×	✓	×	✓
Perino [97]	×	×	✓	×	×
Mani [82]	×	×	✓	×	×
Choi [50]	×	×	×	×	✓
This study	✓	✓	✓	✓	✓

Table 3.24: The number of collected open proxies, the percentage of content modifications and modification types in existing studies(#Collected: number of collected open proxies, #Responsive: number of responsive open proxies, %CM: percentage of Content Modifying proxies)

Studies	#Collected	#Responsive	%CM	Modification Types
Scott [107]	4,250	1,880	N/A	N/A
Tsirantonakis [112]	65,871	19,473	5.15%	Tracking/Fingerprinting/ Privacy leakage/Malware
Perino [97]	180,000	39,143	≈10%	Ad injection/Fingerprinting/ Tracking
Mani [82]	107,034	31,000	≈8%	Ad injection/Cryptojacking/ Eavesdropping/Malware
Choi [50]	1,045,468	N/A	N/A	N/A
This study	436,451	104,114	7.27%	Replacement/Ad injection/ CSS injection/Redirection/ Collect user info/Cryptojacking

purpose. The previous studies also have the same limitations. Our open proxy sources may not be complete, and some open proxies may not be included in our dataset. However, we have attempted to find as many open proxy lists as possible, which can be automatically crawled and downloaded to shorten the experiment time and enrich our proxy dataset. Moreover, in this study, we have also collected and tested most open proxies in the previous works.

We use the DOM tree structures to identify content modifications similar to

Tsirantonakis's work [112]. However, in this work, we only use DOM structures to determine whether contents are modified and which parts are modified. To identify proxy groups that share similar behaviors, we employ a new approach that extracts owner information from elements, including metadata (title, keywords, and other fields), inject library, and URLs by parsing the HTML content. By combining the DOM structures and parsed owner information, we can quickly and accurately identify proxy owners and then group them.

We do not investigate whether the open proxies modify dynamic elements since it is challenging to decide whether the changes of dynamic elements are caused by themselves or open proxies. In the future, we will design websites that include dynamic elements to test open proxies and introduce new methods to distinguish the modifications caused by open proxies from those made by websites.

3.7.3 Comparisons with Other Studies

Here we present a comparison of our work with existing open proxy studies and highlight the improvement of our study from prior work.

In this study, we conducted a larger-scale analysis of the open proxy ecosystem. Table 3.24 lists the number of collected and responsive proxies in related studies and our work. Among those studies, the size of our collected open proxy dataset is the second-largest. Note that, although the study [50] examined a larger open proxy dataset, it lacks the active measurements and verification process of open proxies as it only analyzed open proxies based on the passive measurements. By contrast, our study combines active and passive measurements to investigate the open proxy ecosystem. Furthermore, as shown in Table 3.24, our study collects and examines significant more responsive open proxies than other studies, and those responsive open proxies are more critical and representative in the open proxy ecosystem.

Table 3.24 compares the percentage of identified content modifications and studied modification types. Our work presents the most comprehensive analysis on the misbehavior of open proxies. Table 3.23 shows the research content of the open proxy studies. We first analyzed cloud-based proxies and long-term proxies. In particular, although the work done by Scott *et al.* [107] examined several specific open proxy server owners, our study is the first to identify and analyze the groups of open proxy owners and their behaviors in a systematic manner.

3.8. Summary

This chapter presents a comprehensive measurement study and in-depth analysis of the open proxy ecosystem. We conducted a large-scale measurement that collected more than 436 thousand proxies (including more than 104 thousand responsive proxies) over ten months. We characterized the open proxies' deployment, performance, and behaviors. We collected and analyzed large amounts of responses and classified open proxies based on their DOM tree structures. Furthermore, we identified and tracked the owners of open proxy groups by parsing HTML content and extracting identifier information. We analyzed the categories of content modification and deployment as well as the management strategy of malicious open proxies. We found that 76.42% of content modification proxies demonstrate malicious behaviors, among which Ad injection and redirection are the most prevalent activities. Our case studies show that malicious open proxy owners manipulate proxy deployment to increase their impacts by changing the deployment of their proxies (*e.g.*, the ASes and locations). Finally, we studied two specific groups of proxies, cloud-based proxies and long-term proxies. Our analysis shows that cloud-based proxies are a small portion of the open proxy ecosystem, but these proxies are more reliable and have better performance than non-cloud proxies. Meanwhile, long-term proxies demonstrate better performance than short-term proxies.

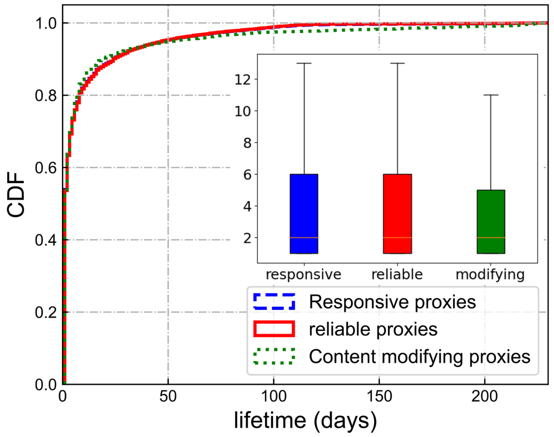


Figure 3.3: CDF and boxplot of Lifetime

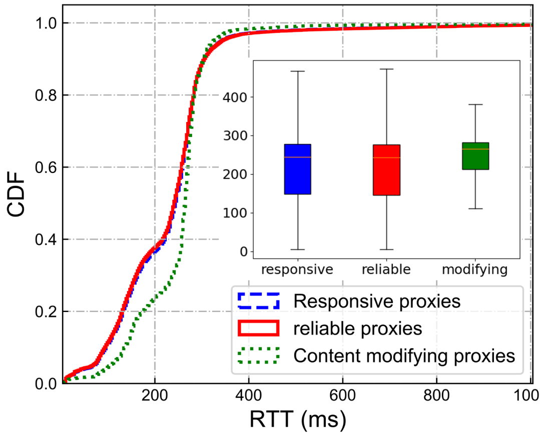


Figure 3.4: CDF and boxplot of RTT.

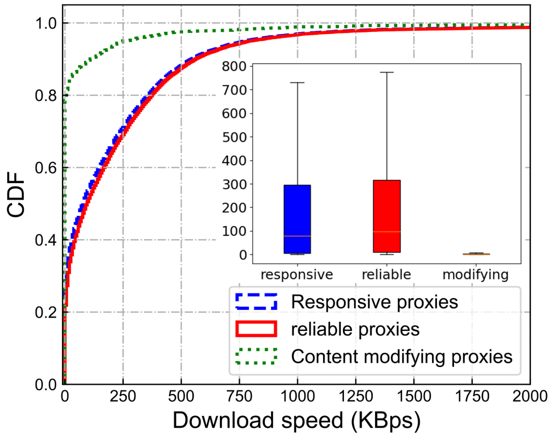
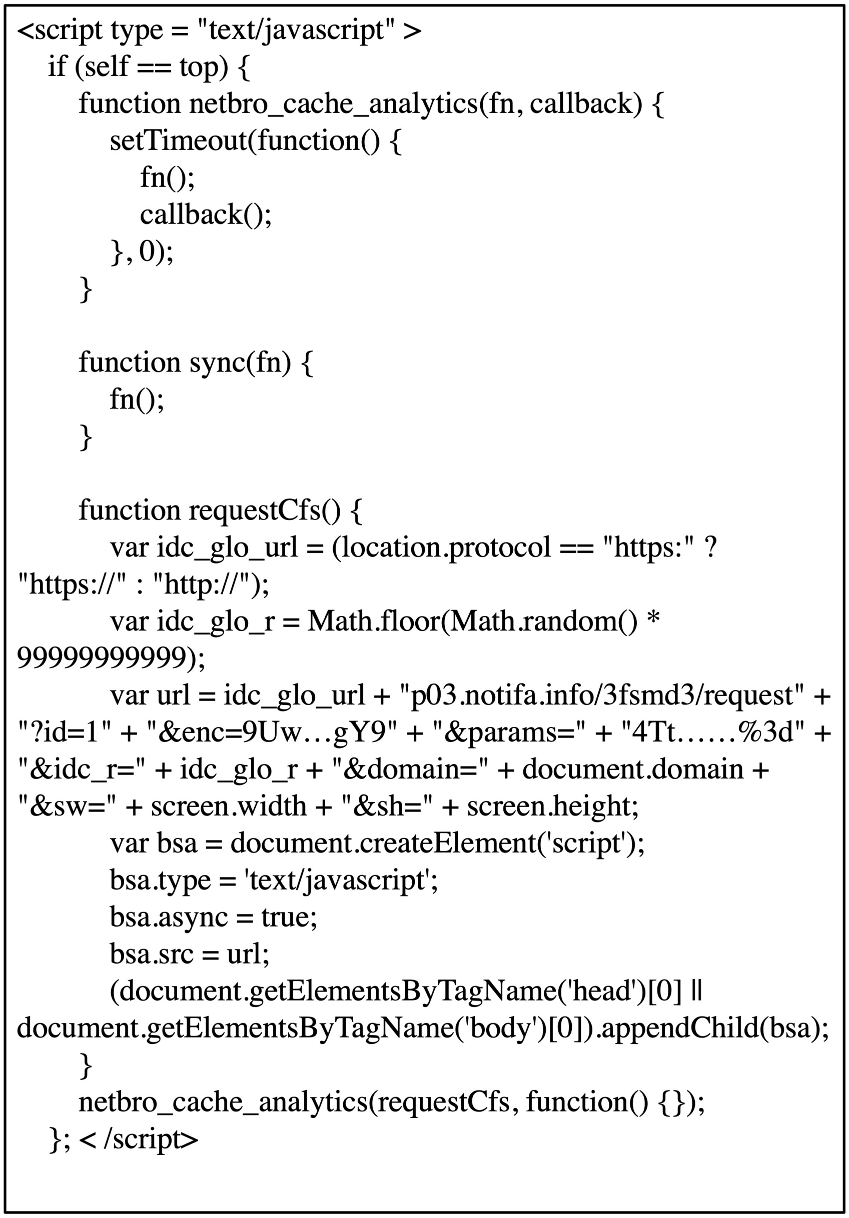


Figure 3.5: CDF and boxplot of Download Speed

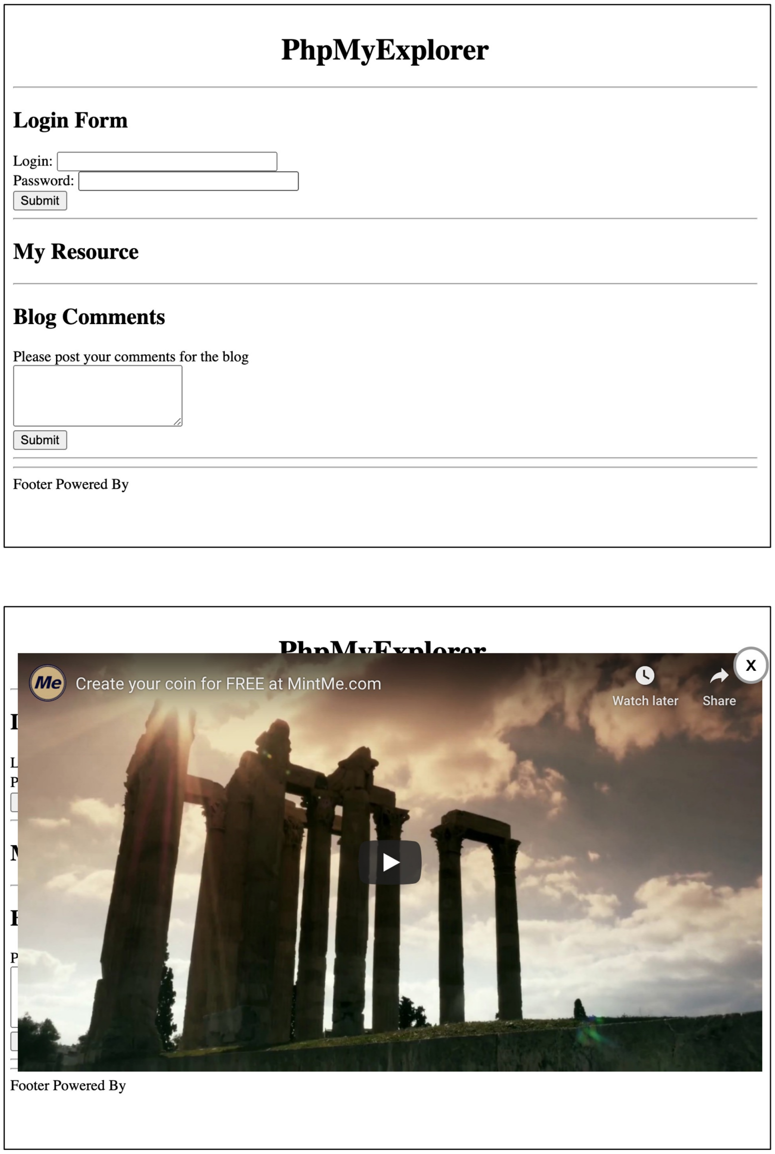


```
<script type = "text/javascript" >
   if (self == top) {
      function netbro_cache_analytics(fn, callback) {
         setTimeout(function() {
            fn();
            callback();
         }, 0);
      }

      function sync(fn) {
         fn();
      }

      function requestCfs() {
         var idc_glo_url = (location.protocol == "https:" ?
"https://" : "http://");
         var idc_glo_r = Math.floor(Math.random() *
99999999999);
         var url = idc_glo_url + "p03.notifa.info/3fsmd3/request" +
"?id=1" + "&enc=9Uw…gY9" + "&params=" + "4Tt……%3d" +
"&idc_r=" + idc_glo_r + "&domain=" + document.domain +
"&sw=" + screen.width + "&sh=" + screen.height;
         var bsa = document.createElement('script');
         bsa.type = 'text/javascript';
         bsa.async = true;
         bsa.src = url;
         (document.getElementsByTagName('head')[0] ||
document.getElementsByTagName('body')[0]).appendChild(bsa);
      }
      netbro_cache_analytics(requestCfs, function() {});
   }; < /script>
```

Figure 3.6: Injected code by ISP



PhpMyExplorer

Login Form

Login:
Password:
Submit

My Resource

Blog Comments

Please post your comments for the blog

Submit

Footer Powered By

PhpMyExplorer

Create your coin for FREE at MintMe.com

Footer Powered By

Figure 3.7: Screenshots of the cryptojacking page. The top is the login page, and the bottom is the login page covered by the pop-up video of Ad.

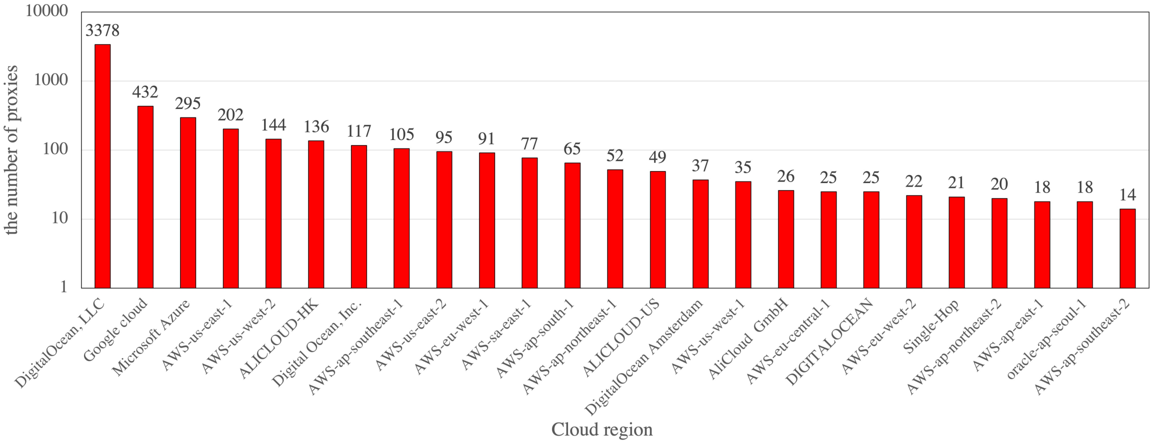


Figure 3.8: The number of collected proxies in different regions of cloud platforms

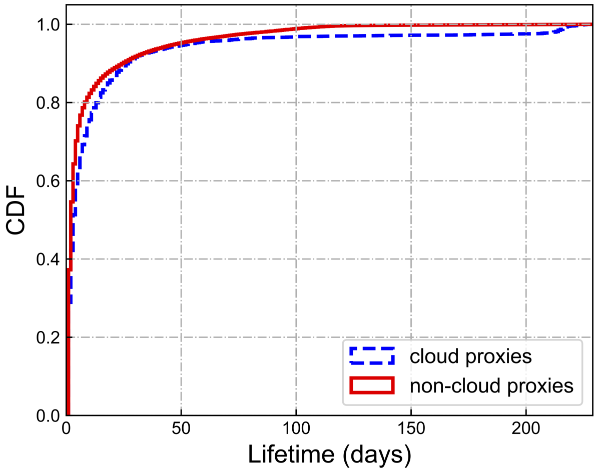


Figure 3.9: CDF of Lifetime of cloud-based proxy and non-cloud-based proxy

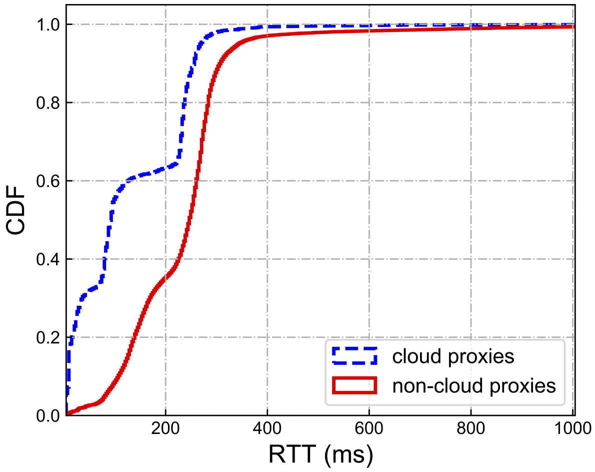


Figure 3.10: CDF of RTT of cloud-based proxy and non-cloud-based proxy

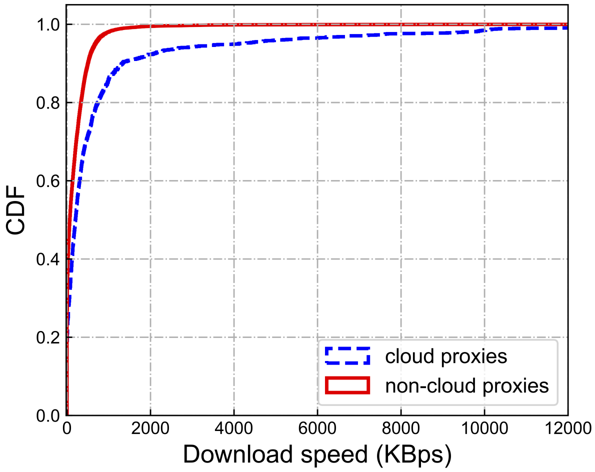


Figure 3.11: CDF of Download Speed of cloud-based proxy and non-cloud-based proxy

Chapter 4

A LARGE-SCALE ANALYSIS OF TRANSPARENT MIDDLEBOX ON THE INTERNET

4.1. Introduction

Transparent proxies [56, 120, 123, 74, 49] are one type of web proxy servers [80, 38, 77, 115, 86, 50, 110] that relay the traffic between clients and servers. Transparent proxies intercept requests and responses between clients and web servers, but clients and web servers may not be aware of the existence of transparent proxies. The transparent proxies are typically deployed by ISPs (Internet Service Providers) and enterprises, or are enabled as a function on the user-side devices such as home routers, so that the proxy servers can monitor, filter, and censor the traffic [65, 76, 47, 121]. By caching the contents [48, 119, 43, 55, 35], transparent proxies can reduce the traffic volume effectively. However, transparent proxies may be legacy proxies that are not well managed and updated. Transparent proxies may be vulnerable to known attacks such as cache poisoning [91] and Denial of Service attacks. There are only a few prior types of research measuring and studying transparent web proxies [120, 123].

In this work, we investigate an overlooked issue of web browsing, the stealthy interception of the HTTP path by on-path devices especially transparent proxies, which is not yet thoroughly studied and well understood. HTTP queries from clients are typically handled by the requested web servers. However, if transparent proxies handle such queries and transparent proxies understand/process the requests differently from the original web server, the responses could be different from desired results, which may cause potential risks. For example, some transparent proxies ignore the destination IP address in a request but use IP addresses from separated DNS resolutions to forward the request. In this work, we develop a novel technique to comprehensively detect transparent proxies on the Internet. In addition, we particularly examine the transparent proxies that could be vulnerable to cache poisoning attacks.

More importantly, such HTTP interceptions performed by transparent proxies are not authorized by users and are difficult to detect on the user's side, which leads to security and ethical concerns. Users have higher risks of putting their trust in transparent proxy servers, which often lack proper maintenance (e.g., equipped with outdated web server software) compared to a well-known domain's web servers. More seriously, users' personal information may be exposed to rogue transparent proxy owners, thereby causing private leakage damages. In this chapter, we conduct a large-scale analysis of transparent proxies. Our study investigates the magnitude of this problem, characterizes various aspects of transparent proxies, and examines the impact on end-users. In the end, we provide insights for mitigation.

Challenges. There are three main challenges we face for systematically analyzing transparent proxies. (1) It is difficult to detect the presence of the transparent proxy because its IP address may only be visible to the backend servers, not the clients. In other words, we can only get transparent proxy IP addresses from the server side. (2) Another challenge is to acquire clients belonging to different locations and Autonomous Systems (ASes) to perform large-scale measurements, which also should allow fine-tuning of the measurement parameters. A suitable vantage point platform is needed for providing comprehensive coverage. (3) To detect the caching effects of transparent proxies, we need to carefully choose the domain names because not all domain names will be cached by transparent proxies. The requested URLs should also be carefully crafted to avoid affecting normal users and web servers.

Our approach. To address these challenges, we design and develop a new measurement methodology and apply it to a large-scale experiment. We utilize a residential proxy network based on `TCP SOCKS` which provides over 230,000 unique residential IP addresses across more than 200 countries. This comprehensive coverage allows us to understand transparent proxies from a worldwide point of view.

To verify the interception of transparent proxies, we deploy several web servers and domain names. Each vantage point is instructed to send HTTP requests to a list of domains and query non-exist files under and without our controls, but the destination IP address is our controlled server, e.g., URL: `http://123.123.123.123/UUID.css(jpg) host: example.com` (where `123.123.123.123` is our controlled server IP address). Since each requested file UUID.css (jpg) is non-existent, it cannot be cached by transparent proxies when the first request is received. In addition, because of the non-existed requested file, it does not affect the other clients. To increase the success rate of caching detection, Alexa top 200 domain names are selected as the domain test list because of their popularity. We also added one of our controlled domain names to the domain test list to get the IP addresses of transparent proxies. Cache poisoning transparent proxies are distributed globally, but most are located in several countries and ISPs.

Contributions. The major contributions of our study are listed below.

- Understanding: We systematically measure HTTP interceptions by transparent proxies which change the IP addresses in requests to intercept HTTP traffic surreptitiously.

- Methodology: We design novel approaches to conduct a large-scale analysis to characterize HTTP interception, through 951,877 residential IP addresses worldwide.

- Findings: We found that Thousands of transparent proxies are performing forced DNS resolution to intercept HTTP traffic and are vulnerable cache poisoning attacks. Damage might be huge if attackers target popular websites and the vulnerable transparent proxies serve many clients. Transparent proxies are also vulnerable to other attacks such as CPDoS (Cache Poisoned Denial of Service).

- Dataset: We will release our dataset on GitHub to help researchers and Internet users detect HTTP interceptions by transparent proxies.

The remainder of this chapter is organized as follows. We introduce the background of transparent proxy and threat model in §4.2. We present our methodology for measuring and analyzing transparent proxies in §4.3. We characterize the HTTP interception in the transparent proxy ecosystem in §4.4 and present an analysis of vulnerable transparent proxies in §4.5. In §4.6, we discuss the threat of vulnerable transparent proxies. We provide the mitigation in §4.7. We survey the related work in §4.8, and finally, we conclude the chapter in §4.9.

4.2. Background and Threat Model

In this section, we first give an overview of how transparent proxies intercept HTTP requests. Then we introduce our threat model of vulnerable transparent proxy.

4.2.1 HTTP and Transparent Proxy

HTTP. Hypertext Transfer Protocol (HTTP) [66, 37] is an application-layer protocol for transmitting hypermedia documents, such as HTML. It was designed for communication between web browsers and servers, but it can also be used for other purposes. HTTP follows a classical client-server model, with a client opening a connection to make a request, then waiting until it receives a response. HTTP is a stateless protocol, meaning the server does not keep any data (state) between two requests.

When a client wants to obtain a resource, the client requests it via a URL. The server uses this URL to choose one of the variants available-–each variant is called a representation—and returns a specific representation to the client.

A Uniform Resource Locator (URL), colloquially termed a web address, is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it. As an example, `http://www.example.com/sample.css`. We also can use the IP addresses to query web servers, e.g., `http://127.0.0.1/sample.css`.

An HTTP header is a field of an HTTP request or response that passes additional context and metadata about the request or response. For example, a request message can use headers to indicate its preferred media formats, while a response can use the header to indicate the media format of the returned body.

The Host request header specifies the host and port number of the server to which the request is being sent. If no port is included, the default port for the service requested is implied (e.g., 443 for HTTPS and 80 for HTTP).

Transparent Proxy. A transparent proxy is Also known as an intercepting proxy, inline proxy, or forced proxy. A transparent proxy intercepts normal application-layer communication without requiring any special client configuration. Clients need not be aware of the existence of the proxy. A transparent proxy is normally located between the client and the Internet, with the proxy performing some of the functions of a gateway or router.

Intercepting proxies are commonly used in businesses to enforce acceptable use policies, and to ease administrative overheads since no client browser configuration is required. This second reason however is mitigated by features such as Active Directory group policy, or DHCP, and automatic proxy detection.

Intercepting proxies are also commonly used by ISPs in some countries to save upstream bandwidth and improve customer response times by caching. This is more common in countries where bandwidth is more limited (e.g., island nations) or must be paid for.

4.2.2 CPDoS: Cache Poisoned Denial of Service

In this work, we examined transparent proxies with existing web attacks – CPDoS: Cache Poisoned Denial of Service. CPDoS is a new class of web cache poisoning attacks aimed at disabling web resources and websites. The basic idea of CPDoS is: 1) An attacker sends a simple HTTP request containing a malicious header targeting a victim resource provided by some web server. The request is processed by the intermediate cache, while the malicious header remains unobtrusive. 2) The cache forwards the request to the origin server attackers it does not store a fresh copy of the targeted resource. At the origin server, the request processing provokes an error due to the malicious header it contains. 3) As a consequence, the origin server returns an error page stored by the cache instead of the requested resource. 4) The attacker knows that the attack was successful when she retrieved an error page in response. 5) Legitimate users try to obtain the target resource with subsequent requests and they will get the cached error page instead of the original content.

There are three CPDoS attack types: HTTP Header Oversize (HHO), HTTP Meta Character (HMC), and HTTP Method Override Attack (HMO). HHO is sending oversized headers that are not allowed by the cache to generate error messages. HMC is sending HTTP headers that contain a harmful meta-character such as line break or carriage return (\n), line feed (\r), or bell (\a). HMO is sending headers to override methods such as DELETE and PUT, and that method is prohibited by web servers. In this work, we utilized these three CPDoS attack vectors to examine transparent proxies.

4.2.3 Threat Model

Our threat model is shown in Figure 4.1 and Figure 4.2. We assume that users' HTTP requests are monitored by transparent proxies. These transparent proxies are able to intercept the HTTP requests which are originally sent directly to the web server. The transparent proxies send the requests based on their own HTTP understanding and configuration (e.g., forward requests based on IP addresses in original requests, or forward requests based on IP address from their forced DNS resolutions using the domain name in the host header). After the responses are received by transparent proxies, the transparent proxies send the responses to the clients. In other words, the responses are sent from the transparent proxies but not the original server. Therefore, from a client's perspective, HTTP responses appear to come from the original web servers, making the actual interception behaviors difficult to be discerned.

Due to the differences in handling HTTP requests by transparent proxies, clients might receive responses from different web servers. In addition, transparent proxies

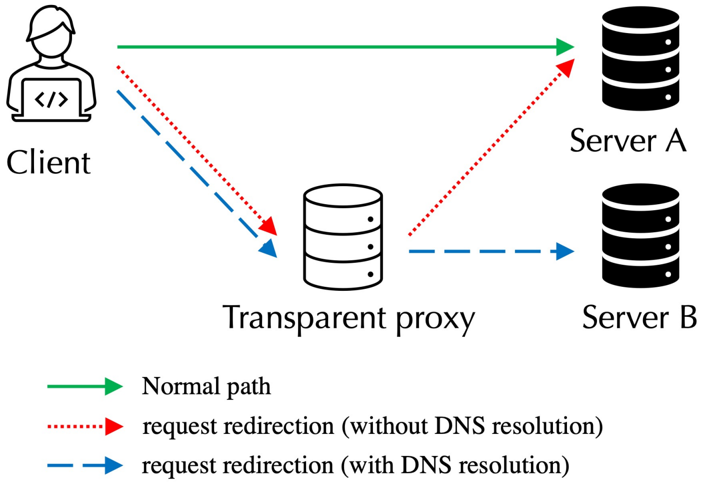

Client
Server A
Transparent proxy
Server B
Normal path
request redirection (without DNS resolution)
request redirection (with DNS resolution)

Figure 4.1: HTTP interception caused by transparent proxy

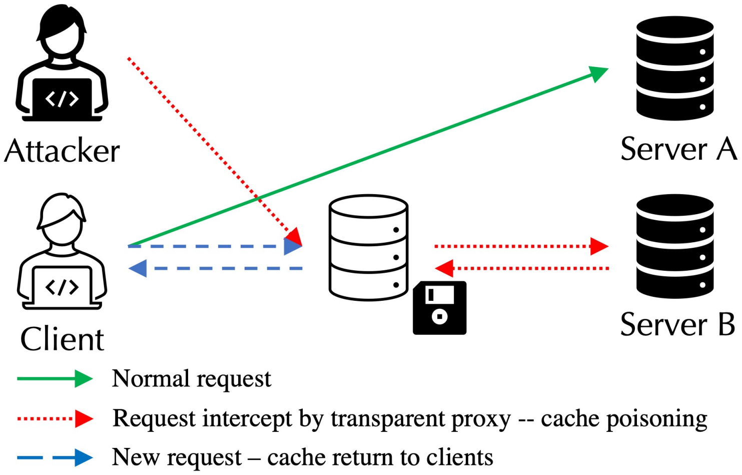

Attacker
Server A
Client
Server B
Normal request
Request intercept by transparent proxy -- cache poisoning
New request – cache return to clients

Figure 4.2: Cache poisoning caused by transparent proxy

typically cache selected possibly highly reused contents such as HTML, pictures, or CSS files, which will cause serious problems.

We discovered that we could inject our selected contents into transparent proxies and trick transparent proxies into caching such content. If other users share the same transparent proxy and query the same contents, the transparent proxy may return our injected contents from its cache. In this scenario, transparent proxies suffer cache poisoning and may cause significant damage.

Scope of study. We aim to measure and characterize transparent proxies through large-scale data analysis. We focus on how transparent proxies intercept HTTP requests and whether transparent proxies suffer cache injection attacks and other web attacks. Other network traffic manipulation mechanisms, such as DNS interception and BGP prefix hijacking, which have been systematically studied before, are not considered in our study.

4.3. Methodology and Data Collection

In this section, we introduce the methodology and data collection in this study, addressing the challenges described in Section §4.1. First, we describe the high-level ideal of our approach and the design requirements it needs to meet. Then, we elaborate on the details of each component of our measurement framework and the workflow of discovering vulnerable transparent proxies. Finally, we discuss potential ethical concerns regarding our data collection.

4.3.1 Overview

We first illustrate our methodology of detecting the presence of transparent proxies on the Internet and identifying potential interceptions by those (vulnerable) transparent proxies.

Approach.

Transparent proxies can monitor and intercept HTTP requests and forward the requests to the web server. During the interceptions, HTTP request messages can be parsed and reconstructed by transparent proxies when being sent to the web server. HTTP requests contain the destination IP address in the Host header, which specifies the address and the domain name of the web server, respectively. Typically, the host header is mapped with the destination IP address, so the request is going to the desired server eventually. The situation becomes complicated when the host header is not mapping with the destination IP address. If there are no transparent proxies, the request will be directly sent to the right server based on the destination IP address. If there is an on-path transparent proxy, this transparent proxy may

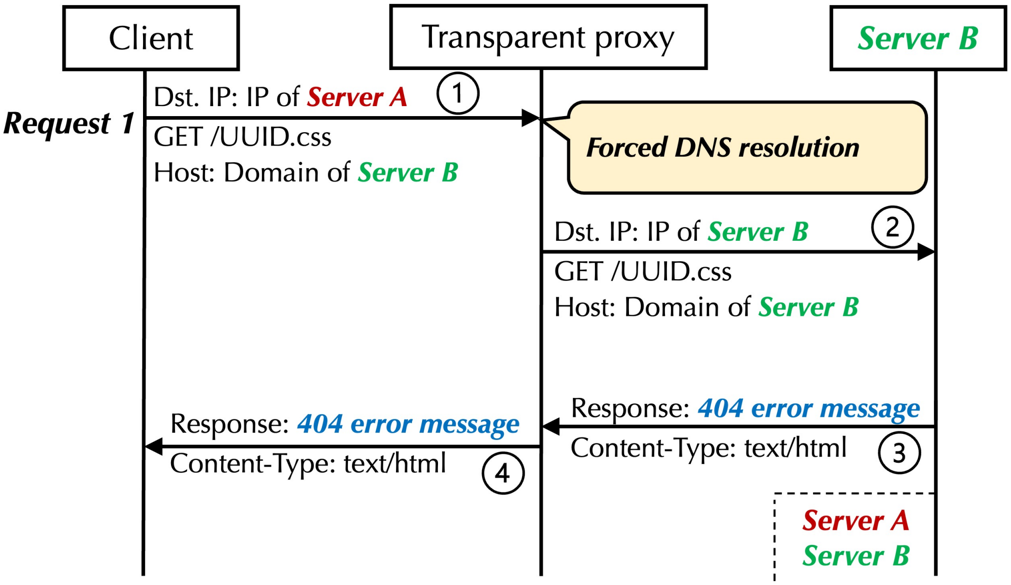


Figure 4.3: request/response flow produced by FDR transparent proxies

forward these requests to a web server that matches the domain name. In doing so, the transparent proxy independently performs DNS requests and redirects the requests to the IP address of the requested domain based on its DNS resolution result. We define this type of transparent proxy as `FDR` – transparent proxy with Forced DNS Resolution. The request/response flow produced by FDR transparent proxies is shown in Figure 4.3. In packet ①, a client sends a request using server A's IP as the destination IP and server B' domain as the host header. A transparent proxy performs DNS resolution, so the destination IP is changed to server B's IP address in packet ②. Server B's content is returned to the client through packets ③ and ④.

Moreover, we discovered another type of transparent proxy with higher vulnerability. This type of proxies forward requests based on the destination IP address. These transparent proxies cache the responses, and they use the host header as the cache key. The cache key is the unique identifier for an object in the cache. Each object in the cache has a unique cache key. A cache hit occurs when a viewer request generates the same cache key as a prior request, and the object for that cache key is in the cache and valid. When there's a cache hit, the requested object is served to the clients from the transparent proxy's cache. Hence, if later requests using the same host

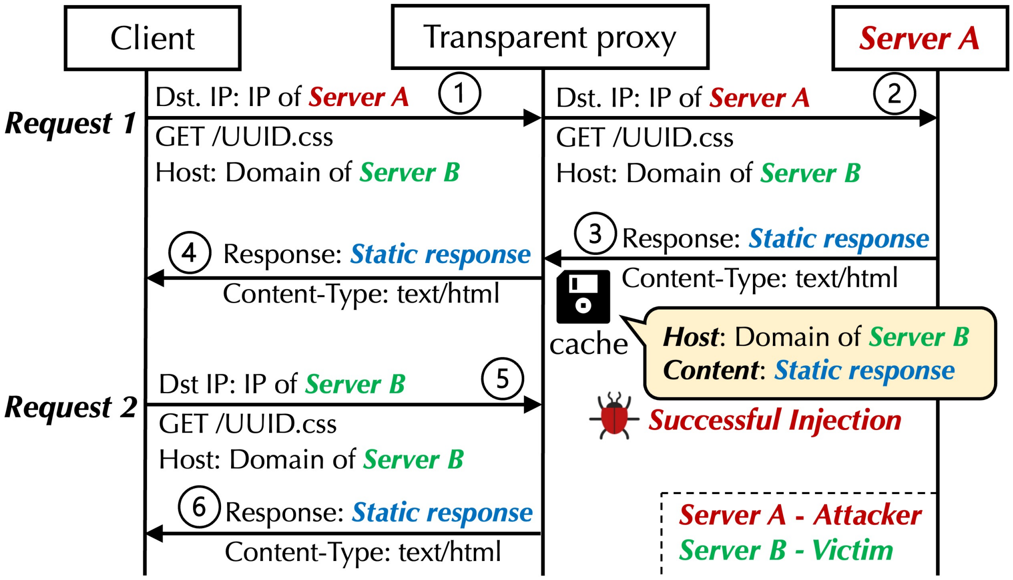


Figure 4.4: request/response flow produced by CPV transparent proxies

header are received, the transparent proxies can return the cached responses. We can use this cache behavior to inject content into the proxy's cache. Specifically, we first send an HTTP request using our server as the Destination IP address and another domain name as the Host Header. This step is to inject our content into the transparent proxies. Then, we send the second request with earlier host headers but a matching destination IP address. This step validates if we can receive the cached content when we request normal HTTP queries. If we still receive the same responses as previously, we can infer that there is a transparent proxy with cache poison vulnerability. We define this type of transparent proxy as `CPV` –transparent proxy with Cache Poison Vulnerability. Note that we can change the host headers to any domain name which means we can inject our contents into any web server using transparent proxies. The request/response flows produced by CPV transparent proxies are shown in Figure 4.4. In packet ①, a client sends a request using server A's IP as the destination IP and server B' domain as the host header. A transparent proxy forwards this request to server A through packet ②. Server A returns a response to the transparent proxy through packet ③. The transparent proxy puts the content in the cache. In packet ④, the transparent proxy returned the response to the client. Another client sends a normal request with server B's IP address and domain name as shown in packet ⑤. The transparent proxy sends the cached content to the client by packet ⑥.

In summary, we can identify those two types of transparent proxies by sending requests with different destination IP addresses in the following steps. (1) We instruct a client to send an HTTP request using our controlled server attacker's IP address as the destination IP address with a host header of the victim server and record its corresponding responses at the client side and both of attacker and victim server side. (2) If the request is forwarded to reach the victim server, we compare the IP address in the log of the victim server with the client's IP address. If the two IP addresses are not the same, we regard it as `FDR` – transparent proxy with forced DNS resolution. (3) instruct a client to send an HTTP request to the server victim with the host header victim; (4) if the client receives the server attacker's content, then compare the IP address in the weblog of the server attacker with the client's IP address. If the two IP addresses differ, we regard it as `CPV` –transparent proxy with cache poisoning vulnerability.

Design requirements. Our methodology should meet several requirements to obtain valid results.

Firstly, the queried resource of each request from the client should be different to avoid caching. Secondly, as we capture packets separately from clients and web servers, we should be able to correlate a request from a client with one captured by our web server. The two issues are addressed by uniquely prefixing each requested file name. Thirdly, the clients in our study should be diverse, being able to send HTTP packets directly to the specified web servers with the specified domain name. Fourthly, aiming to study interception characteristics in-depth, the vantage points are expected to issue diversified HTTP requests (e.g., requests of different methods and headers). The measurement infrastructure used by previous works, including advertising networks, HTTP proxy networks, and Internet scanners, does not meet the requirements.

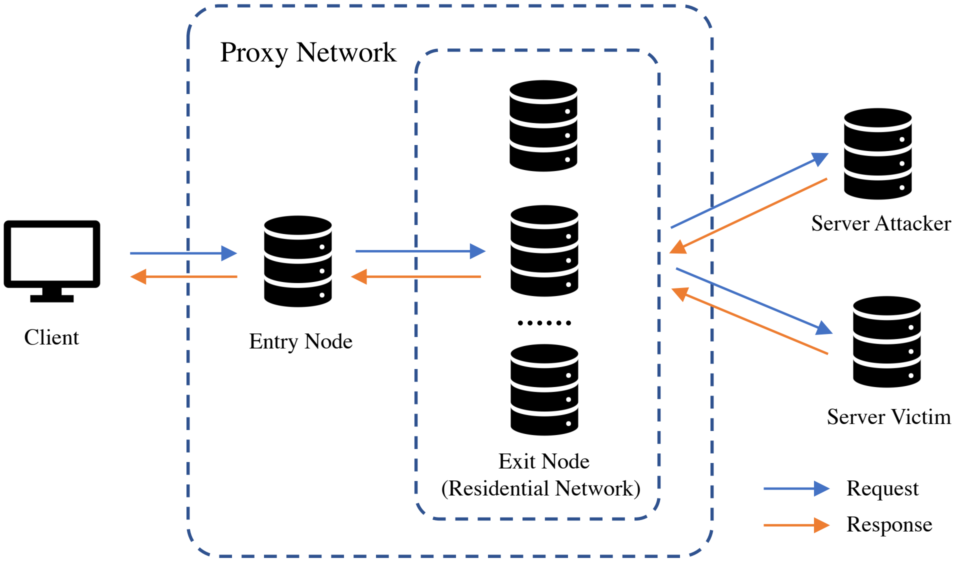


Figure 4.5: Residential SOCKS Proxy Network

4.3.2 Methodology

The experiment setup is shown in Figure 4.5. We will discuss each part of the experiments.

4.3.2.1 Experiment Setup

For the experiment, we set up two servers – server A and server B. Server A is a web server under our control. Server B is the victim server which can not be controlled by us. We configure server A to return static text contents to any requests. Even though the requested host and file cannot be found, the attacker server returns the unchanged response. This is to trick the transparent proxy into caching the content. In our experiments, we send HTTP requests to our controlled servers with the various host headers and file paths. If this content is cached in proxies and the cache key is the host header but not the IP address, the clients will get the content of the server attacker but the server victim. To increase the possibility of caching, we configure the server attacker to send Cache-Control headers to define that this content can be cached and the cache period is long enough. For now, we set Cache-Control:max-age=63072000, public. This ensures the cache's max-age is 2 years.

Server B is a typical web server that returns normal responses. When the requested host and files are not found, server B returns corresponding error messages. Server B can be either our controlled server or web servers without our control. Transparent proxies have policies and configurations about domain selections for caching. Popular websites may have larger possibilities to be cached. In this work, we utilized Alexa top 200 domain names and one controlled domain name as server B's domain names.

4.3.2.2 Generating HTTP Requests

In this study, we need to address the issue of the inconsistent source IPs between a request from the client and its corresponding requested web server. To this end, we devise a method to link those requests by setting a unique file name. The file name includes a distinct UUID (universally unique identifier) generated for each client and a file extension (such as CSS or jpg).

We construct two types of HTTP requests in each experiment. For the first request, the destination IP is our controlled server – server attacker that replies static response to all HTTP requests, the Host is a different web server – server victim's domain name, and Path is UUID + file extension. UUID is a universally unique identifier that labels requests and vantage points (clients). file extensions are file types that can be cached by proxies such as .jpg or .css files. One example request is

HTTP Query:

DST IP:	IP address of the *Server* **A**
Host:	Domain name of the *Server* **B**
Path:	/UUID.css

For the second request: The host is the server victim's domain name. The destination IP address is changed to the server victim's IP address The path is UUID + file extension; UUID is the same as the first request. file extensions are also the same as the first request. One example request is

HTTP Query:
DST IP: IP address of the *Server* **B**
Host: Domain name of the *Server* **B**
Path: /UUID.css

4.3.3 Vantage Points

Our study requires a large number of clients distributed globally. Besides, our clients should be able to send customized HTTP requests. To the end, we leverage a residential proxy network – ProxyRack [98] based on TCP SOCKS which allows us to directly send HTTP packets from globally-distributed clients, to depict a global landscape of vulnerable transparent proxies.

ProxyRack interacts with our measurement client with a Super-proxy. When HTTP packets are sent by our machine, they go to affiliated nodes and leave the network from diverse exit nodes. The packets are forwarded to the web servers (Server A or Server B). ProxyRack has recruited more than 600K exit nodes, so we are able to send HTTP requests from nodes distributed globally to web servers. Because we only can interact with super-proxy, we cannot directly know the IP address of exit nodes (vantage points). In this study, we use an indirect method to obtain the vantage point IP addresses. First, we send a request to IP-API.com through ProxyRack, and IP-API.com return a JSON format of response which include the query IP address (vantage point IP), geolocation, AS, and other information. In this way, we can obtain the vantage point IP address.

4.3.4 Data collection and Dataset

Table 4.1 summarizes our collected dataset in this study. In total, we obtain HTTP traffic from 951,877 residential IP addresses globally.

Format of dataset. Through launching HTTP requests from clients, monitoring web server logs, and capturing HTTP requests, we are able to identify if there are transparent proxies existent on the path. To perform this correlation analysis, our

Table 4.1: Statistics of collected dataset

# IP	# AS	# ISP	# /24 pref	# /16 pref	# country
951,877	10,145	14,657	320,444	22,672	205

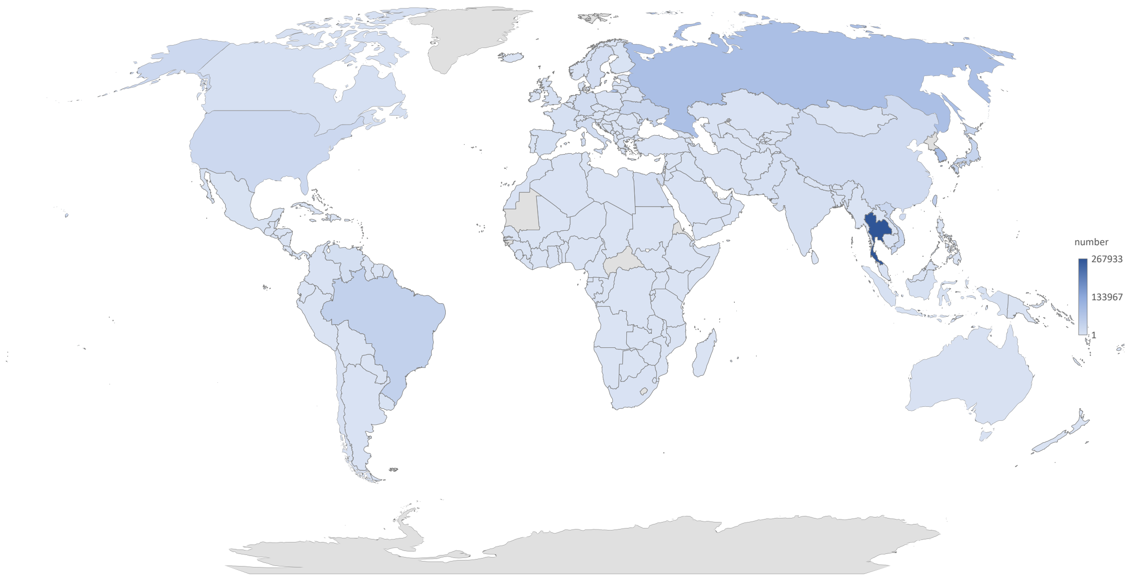


Figure 4.6: Geo-distribution of vantage points

collected data for each HTTP request is stored in a JSON format [39]. For each client, we capture each request and the corresponding response. At our controlled web servers, we collected the source IP address, timestamp, method, URL, headers, user agent, and requested domain name.

Geo-distribution of clients. Leveraging ProxyRack proxy network, we address the challenge of obtaining clients globally. Here we use the geo-distribution of distinct IPs to give an evaluation of our clients. Our collected clients span more than 951,877 unique addresses in 205 countries and 10,145 ASes. Figure 4.6 shows the geo-distribution and our clients cover most countries in the world, with Thailand, South Korea, Russia, Japan, and the U.S. topping the list.

4.3.5 Ethical Considerations

Here, we discuss the ethical considerations when we design and perform our study. Throughout this study, we take utmost care to protect users from the side–effects that may be caused by our experiment.

The residential proxy network we used in this study, ProxyRack, is a commercial service that attracts participants to join the business for profit. Specifically, owners of exit nodes (i.e., our vantage points) have an agreement with ProxyRack that permits ProxyRack traffic to exit from their hosts. Therefore, launching HTTP requests from ProxyRack adheres to the granted permission from the owners of exit nodes.

Regarding our methodology, we carefully craft our HTTP requests and limit their quantities to avoid excessive network traffic. In addition, in our experiments, traffic will go to the victim only when there exists a transparent proxy, otherwise, the traffic will go to our controlled servers. Also, our controlled attacker server only returns static contents which are harmless. We use UUID as the requested file name to avoid affecting other users who share the same transparent proxy.

Through said approaches, we believe we have minimized the threat to users' privacy and security in the experiments.

4.4. Transparent Proxy Interception Analysis

To conduct a global measurement of the transparent proxies, we leverage a residential proxy network based on TCP SOCKS. Here we report our measurement results and analysis by showing its landscape and characteristics.

4.4.1 Scope and Magnitude

We performed our global sale measurement from September 2021 to June 2022, conducting 1,401,567 scans in total. In those scans, we identified `FDR` — transparent proxy with Forced DNS Resolution — in 114,310 scans and `CPV` – transparent proxy

Table 4.2: Distribution of identified FDR and CPV transparent proxies

	FDR	CPV
IP	32,246	11,286
AS	1,458	226
ISP	2,018	257
Countries	98	51
/24 prefix	21,437	2,542
/16 prefix	3,690	474

with Cache Poison Vulnerability — in 29,971 scans. In total, 32,246 IPs are vulnerable to FDR transparent proxies. We found FDR cases in 1458 ASes and CPV cases in 226 ASes. For ISP, we found FDR in 2018 ISPs and CPV in 257 ISPs. For Geo-distribution, 98 and 51 countries are identified as vulnerable to FDR and CPV. The details are shown in Table 4.2. Based on Table 4.2, observed FDR cases are more than observed CPV cases, and FDR cases are distributed more widely than CPV cases.

4.4.2 AS-level Analysis

We observe FDR cases in 1,458 ASes globally. The statistics of AS distribution prove that FDR transparent proxies are spread in many ASes. On the other hand, the results also show that the distribution is very imbalanced. Most of the observed cases are located in only a few ASes. The distribution of ASes is the long-tail distribution that 681 (46.7%) of ASes only have one observed transparent proxy. Table 4.3 shows the top 20 AS that FDR transparent proxies belong to. AS9318 SK Broadband Co Ltd, AS17552 True Online and AS45758 Triple T Internet/Triple T Broadband are the Top 3 AS that have the most observed FDR transparent proxies. The distributions of AS of FDR transparent proxies show that HTTP interception by transparent proxies is very common on the global Internet.

4.4.3 Prefix-level Analysis

We observed FDR transparent proxies in 2,542 /24 prefixes and 474 /16 prefixes. Table 4.4 presents the prefix distributions of observed FDR transparent proxies. For

Table 4.3: Top 20 ASes that have the most FDR transparent proxies

AS number	Organization	#IP
AS9318	SK Broadband Co Ltd	7,833
AS17552	True Online	3,586
AS45758	Triple T Internet/Triple T Broadband	1,850
AS45629	JasTel Network International Gateway	1,417
AS4766	Korea Telecom	758
AS45758	Triple T Broadband Public Company Limited	758
AS58224	Iran Telecommunication Company PJS	631
AS44208	Farahoosh Dena PLC	626
AS25019	Saudi Telecom Company JSC	600
AS17858	LG POWERCOMM	509
AS5384	Emirates Telecommunications Corporation	488
AS13206	1 Realmove Company Limited	476
AS8402	PJSC Vimpelcom	404
AS3462	Data Communication Business Group	360
AS3216	PJSC Vimpelcom	292
AS23969	TOT Public Company Limited	268
AS39650	Atrin Information & Communications Technology Company PJS	250
AS41881	Fanava Group	219
AS49100	Pishgaman Toseeh Ertebatat Company (Private Joint Stock)	184
AS31549	Aria Shatel Company Ltd	180

the /24 prefix, 185.164.75.0/24 contains the most observed transparent proxies (75). In this case, 29.3% of this prefix has observed FDR transparent proxies. For the /16 prefix, 124.122.0.0/16 contains the most observed transparent proxies (587). In this case, 0.89% of this prefix has observed FDR transparent proxies. The distributions of prefixes show that clients in these subnets suffer higher HTTP interceptions than other subnets' clients.

4.4.4 Country-level Analysis

We observed FDR transparent proxies in 98 Countries and areas globally, which means almost half of the countries have this type of vulnerable transparent proxies. The wide distributions of vulnerable transparent proxies demonstrate that this vulnerability is a global security problem, not a single region problem. Table 4.5 presents the top 20 countries which have the most observed transparent proxies. Most observed transparent proxies are in South Korea, Thailand, Iran, Russia, and India.

Table 4.4: Top 20 /24 prefixes and /16 prefixes that have the most observed FDR transparent proxies

/24 prefix	#IP	/16 prefix	#IP
185.164.75.0/24	75	124.122.0.0/16	587
103.237.58.0/24	68	183.88.0.0/16	512
217.66.223.0/24	54	94.74.0.0/16	480
2.188.217.0/24	50	183.89.0.0/16	469
95.38.79.0/24	49	124.120.0.0/16	412
94.241.165.0/24	48	180.183.0.0/16	384
94.241.167.0/24	45	14.207.0.0/16	360
212.16.73.0/24	43	223.205.0.0/16	335
94.74.177.0/24	43	49.49.0.0/16	304
2.188.216.0/24	42	223.206.0.0/16	290
45.132.173.0/24	41	171.96.0.0/16	276
1.247.124.0/24	40	58.8.0.0/16	272
87.107.36.0/24	39	223.204.0.0/16	263
2.188.238.0/24	38	223.24.0.0/16	255
103.237.56.0/24	36	171.97.0.0/16	252
103.237.57.0/24	36	58.11.0.0/16	249
1.247.0.0/24	36	5.190.0.0/16	245
210.16.88.0/24	36	2.188.0.0/16	243
81.12.124.0/24	36	171.100.0.0/16	240
103.207.7.0/24	35	95.38.0.0/16	237

4.4.5 Domain Selection Analysis

In our experiment, 201 domain names are selected to detect FDR transparent proxies. Table 4.6 presents the Top 20 Domain names which trigger FDR transparent proxies. Some of the top domain names are adult websites. Transparent proxy owners might use the domain names to censor and block the content.

4.5. Transparent Proxy Cache Poisoning Analysis

In this section, we report observed CPV transparent proxies in large-scale measurement. In total, 11,017 IPs are vulnerable to CPV transparent proxies.

Table 4.5: Top 20 Countries which have the most observed FDR transparent proxies

Country	#IP
South Korea	10,265
Thailand	8,942
Iran	4,137
Russia	2,924
India	1,447
Bangladesh	782
Saudi Arabia	775
United Arab Emirates	490
Taiwan	389
China	376
Hong Kong	342
United States	234
Japan	224
Ukraine	115
Turkey	84
Oman	74
Singapore	70
Kuwait	59
Canada	57
Qatar	57

4.5.1 AS-level Analysis

We observe CPV cases in 226 ASes globally. The statistics of AS distribution prove that CPV transparent proxies are spread in many ASes. The Internet management organization should notice this threat. The distribution is very imbalanced. Most of the observed cases are located in only a few ASes. The distribution of ASes is the long-tail distribution that 149 (65.5%) of ASes only have one observed transparent proxy. Table 4.7 shows the top 20 AS that transparent proxies belong to. AS45629 Jas-Tel Network International Gateway, AS45758 Triple T Internet/Triple T Broadband and AS23969 TOT Public Company Limited are the Top 3 AS that have the most observed transparent proxies, and all the ASes are Thailand ASes. AS4766 Korea Telecom and AS30722 Vodafone Italia S.p.A have observed more than 50 transparent proxies, and they belong to South Korea and Italy, respectively. That means clients in

Table 4.6: Top 20 Domain names which trigger FDR transparent proxies

Domain name	#Detection scans
xhamster.com	116,660
chaturbate.com	113,063
xnxx.com	108,724
bet365.com	108,476
bongacams.com	108,040
pornhub.com	107,805
xvideos.com	101,129
bet9ja.com	22,896
livejasmin.com	18,670
6.cn	12,412
rednet.cn	10,518
vk.com	8,324
weibo.com	6,280
duckduckgo.com	6,183
apple.com	5,685
tiktok.com	5,528
linkedin.com	4,881
yahoo.com	4,340
amazon.co.uk	4,239
naver.com	4,224

these ASes have a higher possibility to use those vulnerable CPV transparent proxies.

4.5.2 ISP-level Analysis

We observe CPV cases in 257 ISPs globally. Those ISPs may deploy those transparent proxies, but they do not know there are vulnerabilities in these transparent proxies. The distribution is very imbalanced. Most of the observed cases are located in only a few ISPs. The distribution of ISP is the long-tail distribution that 182 (70.1%) of ISPs only have one observed transparent proxy. Table 4.8 shows the top 20 ISP that transparent proxies belong to. Triple T Internet Company Limited is the ISP that contains most of the transparent proxies. 93.4% (10,556) of CPV are observed in this ISP. Triple T Broadband Public Company Limited is a telecommunications company based in Bang Phlat District, Bangkok, Thailand. The Company provides

Table 4.7: Top 20 ASes that have the most observed CPV transparent proxies

AS number	Organization	#IP
AS45629	JasTel Network International Gateway	8,255
AS45758	Triple T Internet/Triple T Broadband	1,739
AS45758	Triple T Broadband Public Company Limited	596
AS23969	TOT Public Company Limited	86
AS4766	Korea Telecom	78
AS30722	Vodafone Italia S.p.A.	58
AS131090	CAT TELECOM Public Company Ltd,CAT	31
AS133481	AIS Fibre	26
AS4760	HKT Limited	21
AS17552	True Online	21
AS49847	Pardazeshgar Ray Azma Co. Ltd.	14
AS12389	PJSC Rostelecom	12
AS852	TELUS Communications Inc.	9
AS17676	Softbank BB Corp.	8
AS131429	MOBIFONE Corporation	7
AS15895	Kyivstar PJSC	7
AS8544	Primetel PLC	7
AS9931	The Communication Authoity of Thailand, CAT	6
AS3462	Data Communication Business Group	5
AS9808	Guangdong Mobile Communication Co.Ltd.	5

telephone, data communication. and Internet services. This ISP should identify those CPV transparent proxies and mitigate the vulnerabilities.

4.5.3 Prefix-level Analysis

We observed CPV transparent proxies in 2,542 /24 prefixes and 474 /16 prefixes. Table 4.9 presents the prefix distributions of observed CPV transparent proxies. For the /24 prefix, 223.205.232.0/24 contains the most observed transparent proxies (40). In this case, 15.6% of this prefix has observed transparent proxies. For the /16 prefix, 223.205.0.0/16 contains the most observed transparent proxies (1,079). In this case, 1.64% of this prefix has observed transparent proxies. The distribution of prefixes shows that clients in these subnets suffer higher vulnerability of CPV than other subnets' clients. Attackers can target the clients in those specific IP prefixes to launch cache poison attacks. The network management teams should eliminate those vulnerable

Table 4.8: Top 20 ISPs which have most observed CPV transparent proxies

ISP	#IP
Triple T Internet Company Limited	10,270
Triple T Broadband Public Company Limited	286
TOT Public Company Limited	85
Korea Telecom	68
Vodafone	44
TRIPLETNET	33
CAT-BB	26
AIS-Fibre	26
Hong Kong Telecommunications (HKT) Limited Mass Internet	18
IP VDF	12
True Internet Corporation CO. Ltd.	11
KORNET	10
Pardazeshgar Ray Azma Co. Ltd.	10
Charter Communications	10
CAT Telecom Public Company Limited	9
Softbank BB Corp.	8
TRUEBB	8
TELUS Communications Inc.	7
China Mobile communications corporation	7
Rostelecom networks	5

transparent proxies as soon as possible to protect the users in those prefixes.

4.5.4 Country-level Analysis

We observed CPV transparent proxies in 51 Countries and areas globally, which means almost 25% of countries have this type of vulnerable transparent proxies. The wide distributions of vulnerable transparent proxies demonstrate that this vulnerability is a global security problem, not a single region problem. Table 4.10 presents the top 20 countries which have the most observed transparent proxies. Most observed transparent proxies are located in Thailand which observes 95.44% of CPV transparent proxies, followed by South Korea, Italy, Ukraine, and Russia.

4.5.5 Domain Selection Analysis

In our experiment, 201 domain names are selected to detect vulnerable transparent proxies. Table 4.11 presents the Top 20 Domain names which trigger vulnerable

Table 4.9: Top 20 /24 prefixes and /16 prefixes that have the most observed CPV transparent proxies

/24 prefix	#IP	/16 prefix	#IP
223.205.232.0/24	40	223.205.0.0/16	1,079
223.205.222.0/24	39	183.88.0.0/16	1,029
223.205.221.0/24	38	223.206.0.0/16	1,008
223.205.249.0/24	37	183.89.0.0/16	925
223.205.223.0/24	36	180.183.0.0/16	790
223.206.233.0/24	36	49.49.0.0/16	781
223.205.219.0/24	35	14.207.0.0/16	729
223.205.234.0/24	33	171.5.0.0/16	659
223.205.251.0/24	33	171.4.0.0/16	657
223.206.220.0/24	33	223.204.0.0/16	641
223.205.220.0/24	32	49.48.0.0/16	632
223.205.246.0/24	32	171.6.0.0/16	542
223.205.248.0/24	32	171.7.0.0/16	464
223.206.246.0/24	32	223.207.0.0/16	463
223.205.216.0/24	30	110.164.0.0/16	25
223.205.218.0/24	30	109.118.0.0/16	20
223.205.235.0/24	30	101.51.0.0/16	12
223.206.238.0/24	30	37.159.0.0/16	11
223.206.222.0/24	29	159.192.0.0/16	10
49.49.216.0/24	29	45.132.0.0/16	10

transparent proxies. Netflix.com tops the list, followed by Spotify.com, Speedtest.net, Instagram.com, and Microsoft.com. Some websites provide audio and video streaming and picture sharing services. Netflix provides video streaming services, Spotify provides audio streaming service, and Instagram provides picture sharing services. Those websites produce a huge amount of traffic on the Internet. We can speculate that ISPs configure the transparent proxies to cache the content of those domain names to save bandwidth, decrease the traffic, and lower cost.

4.5.6 Cached File Type Analysis

We conduct experiments to study how file type affects the caching of CPV transparent proxies. JPG and CSS are selected as test file types. In each experiment,

Table 4.10: Top 20 Countries which have most observed CPV transparent proxies

Country	#IP
Thailand	10772
South Korea	87
Italy	64
Ukraine	41
Russia	40
Japan	36
Hong Kong	28
United States	23
Canada	22
Iran	16
Taiwan	13
China	12
United Kingdom	12
Brazil	10
Iraq	9
Sweden	9
Germany	8
Vietnam	8
Cyprus	7
Malaysia	7

we issue queries simultaneously with the same UUID, domain name, and IP address but two different file types. The experiment period is 50 days. In total, we observed 102 CPV cases. Among them, 7 of them were triggered by both JPG and CSS file types, 92 of them were triggered by JPG file type, and 5 of them were triggered by CSS file. Based on the result, we can speculate that the picture file (or video file) is more likely cached by transparent proxies than the CSS file. The reason might be caching object files such as pictures or videos can save much larger traffic than caching CSS files. Attackers could utilize this cache configuration to launch cache injection and poison attacks. Transparent proxy owners must configure very carefully to defend against such cache poisoning attacks.

Table 4.11: Top 20 Domain names which trigger vulnerable transparent proxies

Domain name	#Detection scans
netflix.com	14,854
spotify.com	7,625
speedtest.net	2,374
instagram.com	1,708
microsoft.com	1,079
vk.com	875
wordpress.com	638
ikea.com	628
tribunnews.com	177
csdn.net	175
msn.com	168
alipay.com	147
panda.tv	144
ebay.com	134
aliexpress.com	134
bongacams.com	126
office.com	123
aparat.com	118
17ok.com	117
twitch.tv	107

4.5.7 Transparent Proxy Server Analysis

To achieve the information of transparent proxies, we use Nmap to perform scans. We present the results of the transparent proxy server – OS, service, and product in the following parts.

Operating System. 139 operating systems are identified among transparent proxies. The top 20 Operating Systems of transparent proxy servers are shown in Table 4.12. HP P2000 G3 NAS device OS is the Top OS in transparent proxies. Linux and Microsoft Windows are popular OS. A lot of those OS are not up to update, so attackers can use the founded vulnerabilities to exploit the transparent proxies.

Service and product. Even though we can only identify a few transparent proxies, we still can get a little information from Nmap scans. For ISP transparent

Table 4.12: Top 20 Operating System of transparent proxy servers

Operating System	# of IP
HP P2000 G3 NAS device	419
MikroTik RouterOS 6.36	227
Linux 2.6.18 - 2.6.22	94
OpenWrt Kamikaze 7.09 (Linux 2.6.22)	75
Linux 3.10 - 4.11	62
Fortinet FortiGate 100D firewall	41
ProVision-ISR security DVR	41
Linux 2.6.32 - 3.13	32
OpenWrt 0.9 - 7.09 (Linux 2.4.30 - 2.4.34)	25
Crestron XPanel control system	24
HP ProCurve MSM422 WAP	19
DD-WRT v24 or v30 (Linux 3.10)	17
Linux 2.6.32	17
Linux 3.2	15
Linux 4.4	14
Linux 2.6.32 - 3.10	13
Linux 2.6.18 (Debian 4.0, x86)	11
DD-WRT v24-sp2 (Linux 3.10)	10
iPXE 1.0.0+	9
Linux 3.0	8

proxies, It isn't easy to detect the service and product information. We can identify some client-side transparent proxies. The top service and product information are shown in Table 4.13. HTTP is the top service of these transparent proxy servers. MikroTik, Huawei, Apple and Hikvision is the top product of the CPV transparent proxies.

4.5.8 Case Study: Characteristics of SP-CPV Transparent Proxy

In the prior parts, we report the observed FDR and CPV transparent proxies. In most cases, the front-end IP address may be the same as the back-end IP address. That is because the transparent proxies may hide in an inner network. However, we identify the cases in which front-end IP addresses are different from back-end IP addresses and denoted as **SP-CPV** transparent proxies. We report these cases as a case study to report this special group of transparent proxies. In total, we confirmed 63 SP-CPV

Table 4.13: Service and products of CPV transparent proxy servers

Service	# IP	Product	# IP
http	723	MikroTik bandwidth-test server	289
domain	294	MikroTik router config httpd	148
bandwidth-test	289	MikroTik	135
unknown	232	Hikvision IPCam control port	123
tcpwrapped	195	Huawei Home Gateway telnetd	104
rtsp	191	Apache httpd	101
telnet	184	Apple AirTunes rtspd	93
pptp	167	Hikvision Network Video Recorder http admin	84
ssh	128	Dropbear sshd	73
ipcam	123	nginx	55
http-alt	89	Unbound	46
ftp	66	OpenSSH	43
https	56	SSL/TLS ClientHello	40
hosts2-ns	50	lighttpd	37
ms-wbt-server	49	Portable SDK for UPnP devices	30
upnp	47	dnsmasq	30
reverse-ssl	40	Microsoft IIS httpd	30
http-proxy	38	MikroTik router ftpd	28
sdr	36	Boa HTTPd	26
vnc	32	MySQL	23

transparent proxies in 5 countries. Next, we analyze SP-CPV vulnerable transparent proxies in several aspects.

4.5.8.1 Geo-distribution of SP-CPV Transparent Proxies

Our collected SP-CPV vulnerable transparent proxies are distributed in five countries which are Thailand, Iraq, Vietnam, Canada, and Italy. The details of geo-distribution are shown in Table 4.6. About 63% of vulnerable transparent proxies are located in Thailand. To see the fine results of geo-distributions, we characterize proxies at the province level. In Thailand, 40 transparent proxies distribute in 17 provinces. The distributions in Thailand are shown in Table 4.14. Bangkok, Chon, and Buri have the most vulnerable transparent proxies. For Iraq, all found vulnerable proxies are in the Baghdad region. For Vietnam, 7 of 8 found vulnerable proxies are in Hanoi, and

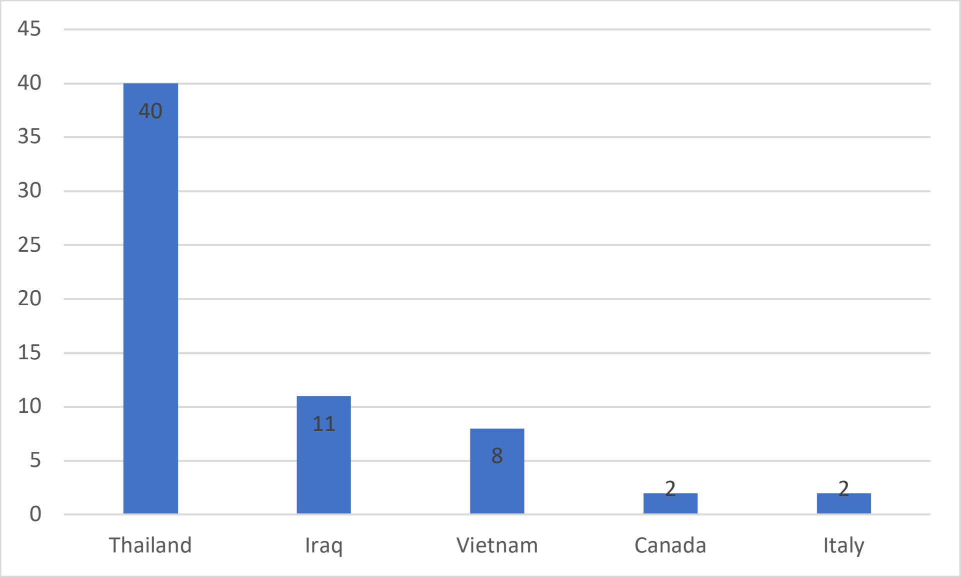


Figure 4.7: Geo-distribution of SP-CPV vulnerable transparent proxies

1 proxy is in Ho Chi Minh. For Canada, all 2 proxies are in Quebec. For Italy, all 2 proxies are in the Lombardy province.

4.5.8.2 AS distribution of SP-CPV Transparent Proxies

The AS distribution of vulnerable SP-CPV transparent proxies is shown in Table 4.15. We found 5 ASes have deployed such transparent proxies. The AS distribution aligns with geo-distribution. We also list the AS rank of such ASes based on the data provided by CAIDA. The highest-ranking AS is AS852 TELUS Communications Inc – a Canada AS, and the lowest-ranking AS is AS207786 super network for internet service ltd – an Iraq AS.

4.5.8.3 ISP distribution of SP-CPV Transparent Proxies

The ISP distribution of SP-CPV transparent proxies is shown in Table 4.15. We found that 9 ISPs have deployed such transparent proxies.

4.5.8.4 Prefix distribution of SP-CPV Transparent Proxies

The /16 prefix and /24 prefix distributions are shown in Figures 4.9 and 4.10, respectively. Based on the figures, these SP-CPV cases are mainly located in only a

Table 4.14: Geo-distribution of SP-CPV transparent proxies in Thailand

Province of Thailand	number
Bangkok	8
Chon Buri	8
Phuket	7
Nonthaburi	3
Songkhla	2
Chachoengsao	1
Chiang Mai	1
Nakhon Sawan	1
Nong Khai	1
Kalasin	1
Phetchaburi	1
Khon Kaen	1
Prachin Buri	1
Mae Hong Son	1
Surat Thani	1
Nakhon Nayok	1
Nakhon Ratchasima	1

few prefixes.

4.5.9 Case Study: CPDOS on Transparent Proxy Detection

In this study, we investigated three CPDoS attack vectors: HHO, HMO, and HMC.

CPDoS detection methodology. We set a server as the target server to get the requests. When the requests of CPDoS requests are accepted by the target server, the target server returns default error messages. When the normal requests are accepted by the target server, the target server returns the designed static content. In our experiments, we send pairs of requests, one is for the CPDoS attacks, and one is for the normal requests. We compare two responses of each pair. If the first response matches the second response, and the response matches the default error message, we label this CPDoS attack as successful. Next, we need to make sure whether these successful attacks are caused by transparent proxies. We compare the vantage point

Table 4.15: AS-distribution of SP-CPV transparent proxies

AS	number	AS-Rank
AS45758 Triple T Internet Company Limited	40	1052
AS207786 super network for internet service ltd	11	38447
AS131429 MOBIFONE Corporation	8	13643
AS852 TELUS Communications Inc.	2	166
AS30722 Vodafone Italia S.p.A.	2	886

Table 4.16: ISP distribution of SP-CPV transparent proxies

ISP	number
Triple T Internet Company Limited	20
Triple T Broadband Public Company Limited	18
super network for internet service ltd	11
MOBIFONEKV9	5
MOBIFONEKV5	2
Vodafone	2
TRIPLETNET	2
TELUS Communications Inc.	2
MOBIFONE Corporation	1

IP addresses with the IP address in the Apache log of the target server. If these two IP addresses are different, we think that it is caused by transparent proxies.

Result. In this study, we identified two types of CPDoS attacks on transparent proxies: HMC and HHO. There are 434 HMC cases and 32 HHO cases in the transparent proxy study. Our studies show that transparent proxies have potential CPDoS vulnerabilities and transparent proxy owners should mitigate CPDoS vulnerabilities as soon as possible.

4.5.10 Summary of Findings

Our measurement findings in the global analysis are summarized below.

- Ten thousand of transparent proxies are performing potentially harmful HTTP interceptions. More seriously, thousands of transparent proxies are vulnerable to

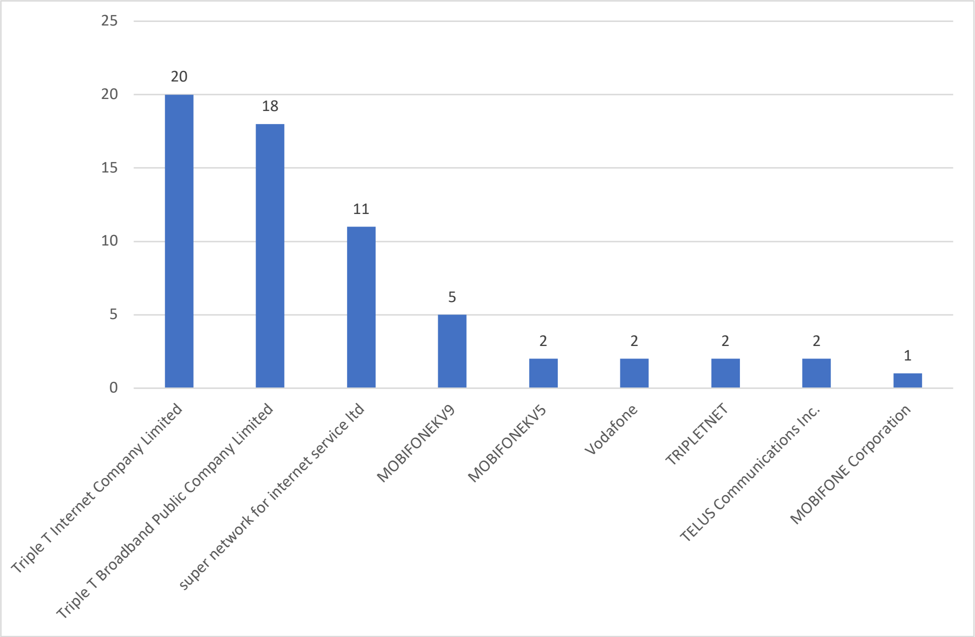


Figure 4.8: ISP distribution of SP-CPV transparent proxies

cache-poisoning attacks.

- HTTP interceptions are distributed globally, and we find FDR transparent proxies in 1,458 ASes and 98 countries.

- CPV transparent proxies are found to exist in 51 countries and 226 ASes.

- CPV transparent proxies may cause serious damage. Damage might be significant if attackers target popular websites and the vulnerable transparent proxies serve many clients.

- Transparent proxies are also vulnerable to other attacks such as CPDoS (Cache Poisoned Denial of Service). We identified 434 HMC and 32 HHO cases in our study.

4.6. Threats

A transparent proxy is difficult to be detected at the client side, and thus Internet users might not realize their traffic is intercepted. First, when HTTP requests from clients are handled by transparent proxies, it is possible to monetize illegally from

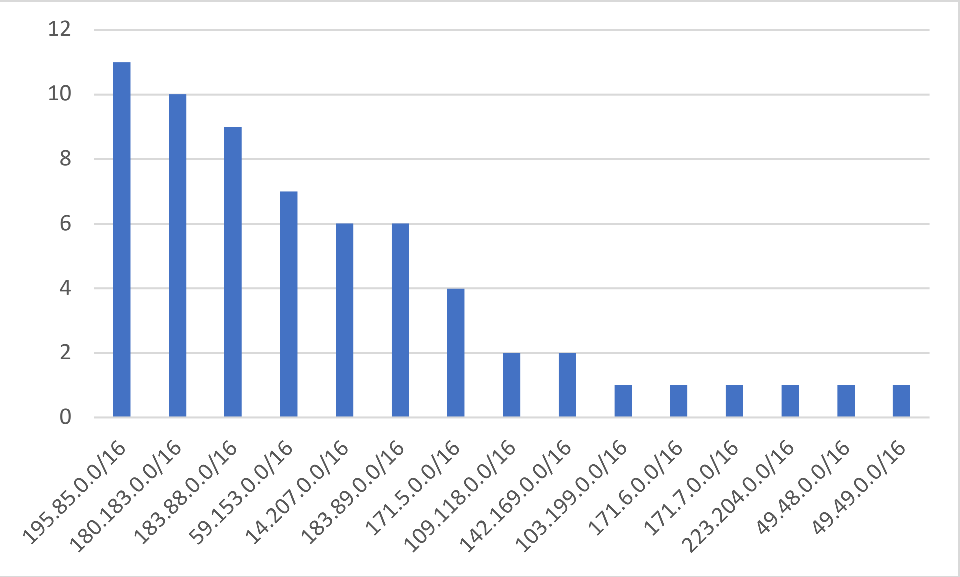


Figure 4.9: /16 prefix distribution of SP-CPV transparent proxies

traffic. Second, as it is difficult for Internet users to detect transparent proxies merely from clients, requested websites can be wrongly blamed when undesired results (e.g., advertisement sites or even malware) are returned. Third, CPV brings severe cache poisoning vulnerability. Attackers might utilize such a vulnerability to inject designed content into transparent proxies. Other clients who share the same transparent proxies may also not get the original content. Moreover, if attackers inject content similar to an online bank or other financial websites, it may cause significant financial damage to clients. Finally, intercepted HTTP requests can be snooped on by untrusted third parties, leading to the leak of private data. Therefore, we believe that transparent proxies potentially induce ethical, privacy, and security risks to Internet users.

4.7. Mitigation Discussion

At present, in our study, almost all HTTP packets are sent unencrypted, which makes them vulnerable to snooping and manipulation. This problem has already been noticed by the Internet community, and RFC 2818 [100], which describes the specification of HTTP over Transport Layer Security (TLS), is released to address this problem. Many popular websites have deployed HTTPS as the major protocol to communicate

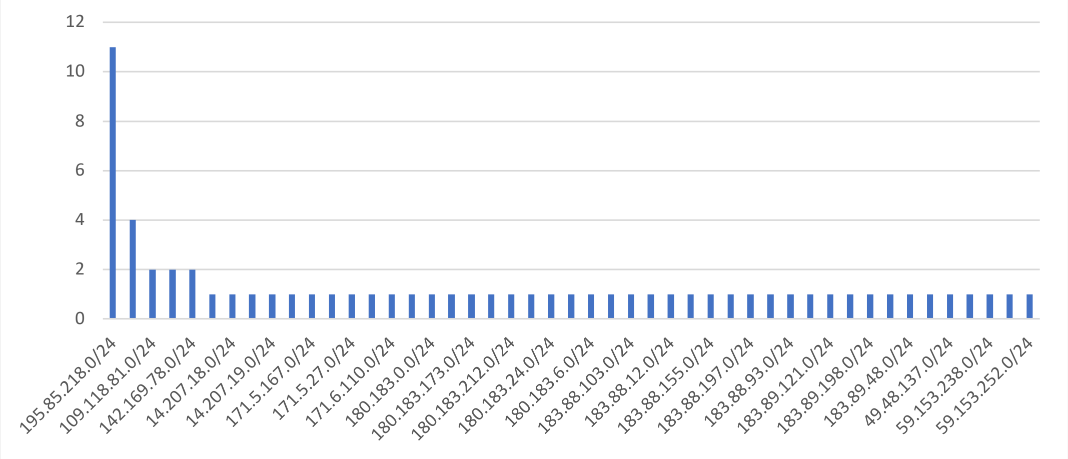


Figure 4.10: /24 prefix distribution of SP-CPV vulnerable transparent proxies

with clients. HTTPS can provide authentication of the accessed website, and protect the privacy and integrity of the exchanged data while in transit. It protects against man-in-the-middle attacks, and the bidirectional encryption of communications between a client and its server protects the communications against eavesdropping and tampering. Unfortunately, the deployment of HTTPS is sophisticated for web servers. As such, the wide deployment of this initiative could take a long time. We highly recommend using HTTPS over HTTP to prevent potential interceptions.

Based on RFC 2616 [67], transparent proxies should not modify the request or response beyond what is required for proxy authentication and identification. However, in our study, we observed that a large number of transparent proxies do not follow the standard. They perform DNS resolutions to get the destination IP address but ignore the destination IP address in the request. This behavior might cause significant damage to clients or/and servers. Transparent proxy managers should configure the proxy server very carefully to avoid unintended consequences. We also found that many transparent proxies are using outdated operating systems and software with vulnerabilities and exploits like CVE (Common Vulnerabilities and Exposures). Attackers may use CVE to hack the proxy server easily. Transparent proxy owners should keep the OS and software up-to-date to avoid such attacks.

4.8. Related Work

Xu *et al.* [120] conducted an analysis of transparent proxy behavior and how transparent web proxies interact with HTTP traffic in four major US cell carriers. They found that all four carriers use these proxies to interpose on HTTP traffic, but they vary in terms of behaviors. In our study, we mainly focus on residential transparent proxies. Zhang *et al.* [123] performed a measurement study on HTTP traffic manipulation by transparent proxies in China-wide networks. They discovered that transparent proxies modified web page contents by replacing or injecting with advertisements and transparent proxies could inject HTTP headers which raised privacy concerns. Our work performs a measurement study on transparent proxies on a global scale which gives a wider view.

Fanou *et al.* [63] performed a study of web infrastructure in Africa. Their mapping of middleboxes in the region reveals a greater presence of transparent proxies in Africa than in Europe or the US. Our global measurement result is consistent with Fanou's findings about transparent proxies.

Nguyen *et al.* [91] introduced and analyzed a new class of web cache poisoning attacks – Cache Poisoned DoS (CPDoS) attack. They studied how to provoke errors during request processing on an origin server and the case, in which error responses get stored and distributed by caching systems. They identified that one proxy cache product and five CDN services are vulnerable to CPDoS attacks. In this work, we explore whether transparent proxies are vulnerable to CPDoS attacks.

Mirheidari *et al.* [87] present the first large-scale study of web cache deception (WCD) where an attacker tricks a caching proxy into erroneously storing private information transmitted over the Internet and subsequently gains unauthorized access to that cached data. The authors quantify the prevalence of WCD in 340 high-profile sites among Alexa Top 5K. Mirheidari *et al.* [88] proposed a novel WCD detection methodology that can be tested any website. They expand the knowledge of WCD attacks, their spread, and their implications. Tyson *et al.* [113] investigated HTTP header manipulation of proxies and middleboxes and analyzed the factors affecting head manipulation. Chung *et al.* [51] detected end-to-end violations of DNS, HTTP, and HTTPS through a paid residential proxy service. They found that up to 4.8% of nodes are subject to some end-to-end violations.

Nguyen *et al.* [90] proposed a cache testing environment that can be used to analyze shared caches. They analyzed seven different shared caching systems. The results showed that caches did perform differently in many respect and some peculiarities had the potential for future incidents. Nguyen's work was all conducted in an experimental environment, however, our work identified real-world cache security problems and analyzed the real impact on the modern global Internet.

4.9. Summary

In this chapter, we perform a large-scale study on HTTP interceptions by transparent proxies, which induce the security, privacy, and performance issues. We develop a set of techniques to detect the stealthy behavior of transparent proxies by utilizing well-maintained proxy platform with numerous vantage points. Based on our dataset, we observe that HTTP interceptions by transparent proxies exist in 1,458 ASes and 98 countries. In addition, interception characteristics are further analyzed. Our results indicate that the stealthy HTTP interceptions by transparent proxies can potentially introduce new threats in the web ecosystem, and new solutions are needed to address the threat. Furthermore, we study the security problems around transparent proxies such as caching poisoning attack and CPDoS. We find cache-poisoning-prone transparent proxies in 226 ASes and 51 countries. For CPDoS, we identify 434 HMC and 32 HHO cases in our study. In the end, we analyze the threats caused by transparent proxies and discuss mitigation solutions.

Chapter 5
CONCLUSION

This dissertation focuses on using stand-off observations and measurements to understand different aspects of the global Internet. In particular, we (1) explored the passive way to classify anycast prefixes based on BGP information, (2) studied the open proxy ecosystem by collecting the measurement data of a large number of open proxies, and (3) explored the problems of HTTP interceptions and security issues caused by transparent proxies. In this chapter, we summarize these studies presented in this dissertation.

5.1. Summary and Contributions

In our first work, we presented a passive method to study IP anycast by utilizing BGP data. Without using active measurements, we proposed a set of BGP-related features to classify anycast and unicast prefixes. Using the datasets collected from RouteViews and RIPE RIS, we evaluated the effectiveness of our proposed approach. The evaluation results show that our approach achieves high classification accuracy, about 90% for anycast and 99% for unicast. While further delving into the causes of inaccuracy, we found that remote peering has an unintended impact on anycast routing. In our study, 19.6% of anycast prefixes are sensitive to remote peering, and over 62% of such prefixes are further confirmed by traceroute measurements. We revealed that remote peering could increase transmission latency by routing traffic to distant suboptimal anycast sites.

In our second study, we conducted a comprehensive measurement and in-depth analysis of the open proxy ecosystem. We conducted a large-scale measurement that collected more than 436 thousand proxies (including more than 104 thousand responsive proxies) over ten months. We characterized the open proxies' deployment, performance, and behaviors. We collected and analyzed large responses and classified open proxies based on their DOM tree structures. Furthermore, we identified and tracked the owners of open proxy groups by parsing HTML content and extracting identifier information. We analyzed the categories of content modification and deployment as well as the management strategy of malicious open proxies. We found that 76.42% of content modification proxies demonstrate malicious behaviors, among which Ad injection and redirection are the most prevalent activities. Our case studies show that malicious open proxy owners manipulate proxy deployment to increase their impacts by changing the deployment of their proxies (*e.g.*, the ASes and locations). Finally, we studied two specific groups of proxies, cloud-based proxies and long-term proxies. Our analysis shows that cloud-based proxies are a small portion of the open proxy ecosystem, but these proxies are more reliable and have better performance than non-cloud proxies. Meanwhile, long-term proxies demonstrate better performance than short-term proxies.

In our third study, we performed a large-scale study on HTTP interceptions caused by transparent proxies, which induce security, privacy, and performance issues. We developed a suite of techniques to detect such a hidden behavior, i.e., leveraging one proxy platform with numerous vantage points. Based on our collected dataset, we found that HTTP interceptions by transparent proxies exist in many ASes and networks. In addition, the interception characteristics as well as motivations are further analyzed. Our results indicate that the hidden HTTP interceptions by transparent proxies can potentially introduce new security threats in the Web ecosystem, and new solutions are needed to address these threats. Furthermore, we studied the security problems around transparent proxies such as caching poisoning problems and CPDoS.

BIBLIOGRAPHY

[1] a2u. https://github.com/a2u/free-proxy-list.

[2] abuseipdb:113.161.62.81. https://www.abuseipdb.com/check/113.161.62.81.

[3] Blog: hn.kd.ny.adsl: Research, ban. https://dontai.com/wp/2016/06/08/hn-kd-ny-adsl-research-ban/.

[4] clarketm. https://github.com/clarketm/proxy-list.

[5] dailyfreeproxy. https://www.dailyfreeproxy.com/.

[6] Domain reputation: azteca-comunicaciones.com. https://talosintelligence.com/reputation_center/lookup?search=azteca-comunicaciones.com.

[7] Domain reputation: hn.kd.ny.adsl. https://talosintelligence.com/reputation_center/lookup?search=hn.kd.ny.adsl.

[8] Domain reputation: static.vnpt.vn. https://talosintelligence.com/reputation_center/lookup?search=static.vnpt.vn.

[9] fate0. https://github.com/opsxcq/proxy-list.

[10] G-tools. https://github.com/jaxBCD/G-Tools.

[11] Live ip map:200.69.79.170. https://www.liveipmap.com/200.69.79.170.

[12] live-sock. http://www.live-socks.net/.

[13] Norton community: repeated portscan issue with hn.kd.ny.adsl. https://community.norton.com/en/forums/repeated-portscan-issue-hnkdnyadsl.

[14] openproxy.space. https://openproxy.space/list.

[15] opsxcq. https://github.com/opsxcq/proxy-list.

[16] otx.alienvault.com:azteca-comunicaciones.com. https://otx.alienvault.com/indicator/domain/azteca-comunicaciones.com.

[17] proxy-daily. https://proxy-daily.com/.

[18] proxy-list.download. https://www.proxy-list.download/HTTP.

[19] Proxybroker. https://github.com/constverum/ProxyBroker/.

[20] proxylistdaily. https://www.proxylistdaily.net/.

[21] proxyserverlist24. http://www.proxyserverlist24.top/.

[22] Pydnsbl,async dnsbl lists checker based on asyncio/aiodns. https://github.com/dmippolitov/pydnsbl.

[23] sinium. https://seopro.sinium.com/free-proxy-list.

[24] smallseotools. https://smallseotools.com/free-proxy-list/.

[25] Sophos community: Top hacker hn.kd.ny.adsl ?? https://community.sophos.com/utm-firewall/f/network-protection-firewall-nat-qos-ips/39664/top-hacker-hn-kd-ny-adsl.

[26] The spam auditor blog:smtp auth attacks, how big is the problem really? https://spamauditor.org/2019/01/smtp-auth-attacks-how-big-is-the-problem/.

[27] Spicework community: Hi been my network has been comppromised by a known port scammer, hn.kd.ny.adsl,. https://community.spiceworks.com/topic/2301469-hi-been-my-network-has-been-comppromised-by-a-known-port-scammer-hn-kd-ny-adsl.

[28] Suc012 : Chinese blind sql injection – hn.kd.ny.adsl. https://eromang.zataz.com/2010/04/30/suc012-blind-sql-injection-china/.

[29] Thespeedx. https://github.com/TheSpeedX/PROXY-List.

[30] vnpt.vn content scraper: Research, ban. https://dontai.com/wp/2016/06/14/vnpt-vn-content-scraper-research-ban/.

[31] https://github.com/bianrui0315/ccr_Anycast, 2019.

[32] Joe Abley and Kurt Erik Lindqvist. Operation of Anycast Services. RFC 4786, 2006.

[33] Sumayah A Alrwais, Alexandre Gerber, Christopher W Dunn, Oliver Spatscheck, Minaxi Gupta, and Eric Osterweil. Dissecting ghost clicks: Ad fraud via misdirected human clicks. In *Proceedings of the 28th Annual Computer Security Applications Conference*, pages 21–30, 2012.

[34] Anycast Enumeration and Geolocation Dataset. https://anycast.telecom-paristech.fr/dataset/, 2015-2017.

[35] Martin Arlitt, Ludmila Cherkasova, John Dilley, Rich Friedrich, and Tai Jin. Evaluating content management techniques for web proxy caches. *ACM SIGMETRICS Performance Evaluation Review*, 27(4):3–11, 2000.

[36] Sajjad Arshad, Amin Kharraz, and William Robertson. Identifying extension-based ad injection via fine-grained web content provenance. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 415–436, 2016.

[37] Mike Belshe, Roberto Peon, and Martin Thomson. Hypertext transfer protocol version 2 (http/2). Technical report, 2015.

[38] Rui Bian, Shuai Hao, Haining Wang, and Chase Cotton. Shining a light on dark places: A comprehensive analysis of open proxy ecosystem. *Computer Networks*, 208:108893, 2022.

[39] Pierre Bourhis, Juan L Reutter, Fernando Suárez, and Domagoj Vrgoč. Json: data model, query languages and schema specification. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI symposium on principles of database systems*, pages 123–135, 2017.

[40] Sam Burnett and Nick Feamster. Encore: Lightweight measurement of web censorship with cross-origin requests. In *Proceedings of the 2015 ACM conference on special interest group on data communication*, pages 653–667, 2015.

[41] John W Byers. Encore: Lightweight Measurement of Web Censorship with Cross-Origin Requests–Public Review. Technical Report, 2015.

[42] G. Hooghiemstra H. Uijterwaal C. J. Bovy, H. T. Metrodimedjo and P. Van Mieghem. Analysis of end-to-end delay measurements in internet. In *Passive and Active Network Measurement (PAM)*, 2002.

[43] Ramon Caceres, Fred Douglis, Anja Feldmann, Gideon Glass, and Michael Rabinovich. Web proxy caching: The devil is in the details. *ACM SIGMETRICS Performance Evaluation Review*, 26(3):11–15, 1998.

[44] CAIDA AS Relationships Dataset. http://www.caida.org/data/as-relationships/, 2018.

[45] Matt Calder, Ashley Flavel, Ethan Katz-Bassett, Ratul Mahajan, and Jitendra Padhye. Analyzing the Performance of an Anycast CDN. In *ACM Internet Measurement Conference (IMC)*, 2015.

[46] Ignacio Castro, Juan Camilo Cardona, Sergey Gorinsky, and Pierre Francois. Remote Peering: More Peering without Internet Flattening. In *ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2014.

[47] Abdelberi Chaabane, Terence Chen, Mathieu Cunche, Emiliano De Cristofaro, Arik Friedman, and Mohamed Ali Kaafar. Censorship in the wild: Analyzing internet filtering in syria. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 285–298, 2014.

[48] Songqing Chen, Haining Wang, Xiaodong Zhang, B. Shen, and S. Wee. Segment-based proxy caching for internet streaming media delivery. *IEEE MultiMedia*, 12(3):59–67, 2005.

[49] Tzi-cker Chiueh, Harish Sankaran, and Anindya Neogi. Spout: a transparent proxy for safe execution of java applets. *IEEE Journal on Selected Areas in Communications*, 20(7):1426–1433, 2002.

[50] Jinchun Choi, Mohammed Abuhamad, Ahmed Abusnaina, Afsah Anwar, Sultan Alshamrani, Jeman Park, Daehun Nyang, and David Mohaisen. Understanding the proxy ecosystem: A comparative analysis of residential and open proxies on the internet. *IEEE Access*, 8:111368–111380, 2020.

[51] Taejoong Chung, David Choffnes, and Alan Mislove. Tunneling for Transparency: A Large-scale Analysis of End-to-End Violations in the Internet. In *ACM Internet Measurement Conference (IMC)*, pages 199–213, 2016.

[52] Danilo Cicalese, Jordan Augé, Diana Joumblatt, Timur Friedman, and D Rossi. Characterizing IPv4 Anycast Adoption and Deployment. In *ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2015.

[53] Danilo Cicalese, Diana Joumblatt, Dario Rossi, Marc-Olivier Buob, Jordan Augé, and Timur Friedman. A Fistful of Pings: Accurate and Lightweight Anycast Enumeration and Geolocation. In *IEEE INFOCOM*, 2015.

[54] Danilo Cicalese and Dario Rossi. A Longitudinal Study of IP Anycast. *ACM SIGCOMM Computer Communication Review*, 48(1), 2018.

[55] Ian Cooper and John Dilley. Known http proxy/caching problems. Technical report, 2001.

[56] Roger Crandell, James Clifford, and Alexander Kent. A secure and transparent firewall web proxy. In *LISA*, pages 23–30, 2003.

[57] X de Carné de Carnavalet and Mohammad Mannan. Killed by proxy: Analyzing client-end tls interception software. In *Network and Distributed System Security Symposium (NDSS)*, 2016.

[58] Ricardo de Oliveira Schmidt, John Heidemann, and Jan Harm Kuipers. Anycast Latency: How Many Sites Are Enough? In *Passive and Active Network Measurement (PAM)*, 2017.

[59] Wouter B. de Vries, Ricardo de O. Schmidt, Wes Hardaker, John Heidemann, Pieter-Tjerk de Boer, and Aiko Pras. Broad and Load-aware Anycast Mapping with Verfploeter. In *ACM Internet Measurement Conference (IMC)*, 2017.

[60] DNS Root Sever Anycast Trace Data. University of Maryland. http://www.cs.umd.edu/projects/droot/anycast-data.tar.gz, 2018.

[61] Zakir Durumeric, Zane Ma, Drew Springall, Richard Barnes, Nick Sullivan, Elie Bursztein, Michael Bailey, J Alex Halderman, and Vern Paxson. The Security Impact of HTTPS Interception. In *Network and Distributed System Security Symposium (NDSS)*, 2017.

[62] Xun Fan, John Heidemann, and Ramesh Govindan. Evaluating Anycast in the Domain Name System. In *IEEE INFOCOM*, 2013.

[63] Rodérick Fanou, Gareth Tyson, Eder Leao Fernandes, Pierre Francois, Francisco Valera, and Arjuna Sathiaseelan. Exploring and analysing the african web ecosystem. *ACM Transactions on the Web (TWEB)*, 12(4):1–26, 2018.

[64] Aurore Fass, Michael Backes, and Ben Stock. Hidenoseek: Camouflaging malicious javascript in benign asts. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1899–1913, 2019.

[65] Nick Feamster, Magdalena Balazinska, Greg Harfst, Hari Balakrishnan, and David Karger. Infranet: Circumventing web censorship and surveillance. In *11th USENIX Security Symposium (USENIX Security 02)*, 2002.

[66] Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. Hypertext transfer protocol–http/1.1. Technical report, 1999.

[67] Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. Rfc2616: Hypertext transfer protocol–http/1.1, 1999.

[68] Thomas Gerbet, Amrit Kumar, and Cédric Lauradoux. A privacy analysis of google and yandex safe browsing. In *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 347–358, 2016.

[69] Danilo Giordano, Danilo Cicalese, Alessandro Finamore, Marco Mellia, Maurizio Munafò, Diana Zeaiter Joumblatt, and Dario Rossi. A First Characterization of Anycast Traffic from Passive Traces. In *Network Traffic Measurement and Analysis Conference (TMA)*, 2016.

[70] Richard Gomer, Eduarda Mendes Rodrigues, Natasa Milic-Frayling, and MC Schraefel. Network analysis of third party tracking: User exposure to tracking cookies through search. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 1, pages 549–556, 2013.

[71] Shuai Hao, Yubao Zhang, Haining Wang, and Angelos Stavrou. End-Users Get Maneuvered: Empirical Analysis of Redirection Hijacking in Content Delivery Networks. In *USENIX Security Symposium*, 2018.

[72] Shan Huang, Félix Cuadrado, and Steve Uhlig. Middleboxes in the internet: a http perspective. In *Network Traffic Measurement and Analysis Conference (TMA)*, pages 1–9, 2017.

[73] Remote IXP Peering Observatory. http://remote-ixp-peering.net/, 2018.

[74] Xiaohua Jia, Deying Li, Hongwei Du, and Jinli Cao. On optimal replication of data object at hierarchical and transparent web proxies. *IEEE Transactions on Parallel and Distributed Systems*, 16(8):673–685, 2005.

[75] Lin Jin, Shuai Hao, Haining Wang, and Chase Cotton. Your remnant tells secret: Residual resolution in ddos protection services. In *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 362–373, 2018.

[76] Lin Jin, Shuai Hao, Haining Wang, and Chase Cotton. Understanding the practices of global censorship through accurate, end-to-end measurements. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 5(3):1–25, 2021.

[77] Wenquan Jin and DoHyeun Kim. Development of virtual resource based iot proxy for bridging heterogeneous web services in iot networks. *Sensors*, 18(6):1721, 2018.

[78] Ruogu Kang, Stephanie Brown, and Sara Kiesler. Why do people seek anonymity on the internet? informing policy and design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2657–2666, 2013.

[79] Zhihao Li, Dave Levin, Neil Spring, and Bobby Bhattacharjee. Internet Anycast: Performance, Problems, & Potential. In *ACM SIGCOMM*, 2018.

[80] Ari Luotonen. *Web proxy servers*. Prentice-Hall, Inc., 1998.

[81] Doug Madory, Chris Cook, and Kevin Miao. Who Are the Anycasters. *NANOG59*, 2013.

[82] Akshaya Mani, Tavish Vaidya, David Dworken, and Micah Sherr. An extensive evaluation of the internet's open proxies. In *Proceedings of the 34th Annual Computer Security Applications Conference*, pages 252–265, 2018.

[83] MaxMind's GeoLite City Dataset. https://dev.maxmind.com/geoip/geoip2/geolite2/, 2018.

[84] Allison McDonald, Matthew Bernhard, Luke Valenta, Benjamin VanderSloot, Will Scott, Nick Sullivan, J Alex Halderman, and Roya Ensafi. 403 forbidden: A global view of cdn geoblocking. In *Proceedings of the Internet Measurement Conference 2018*, pages 218–230, 2018.

[85] Xianghang Mi, Xuan Feng, Xiaojing Liao, Baojun Liu, XiaoFeng Wang, Feng Qian, Zhou Li, Sumayah Alrwais, Limin Sun, and Ying Liu. Resident evil: Understanding residential ip proxy as a dark service. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1185–1201, 2019.

[86] Xianghang Mi, Xuan Feng, Xiaojing Liao, Baojun Liu, XiaoFeng Wang, Feng Qian, Zhou Li, Sumayah Alrwais, Limin Sun, and Ying Liu. Resident Evil: Understanding Residential IP Proxy as a Dark Service. In *IEEE Symposium on Security and Privacy (S&P)*, 2019.

[87] Seyed Ali Mirheidari, Sajjad Arshad, Kaan Onarlioglu, Bruno Crispo, Engin Kirda, and William Robertson. Cached and confused: Web cache deception in the wild. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 665–682, 2020.

[88] Seyed Ali Mirheidari, Matteo Golinelli, Kaan Onarlioglu, Engin Kirda, and Bruno Crispo. Web cache deception escalates. In *USENIX Security Symposium*, 2022.

[89] Giovane Moura, Ricardo de O Schmidt, John Heidemann, Wouter B de Vries, Moritz Muller, Lan Wei, and Cristian Hesselman. Anycast vs. DDoS: Evaluating the November 2015 root DNS event. In *ACM Internet Measurement Conference (IMC)*, 2016.

[90] Hoai Viet Nguyen, Luigi Lo Iacono, and Hannes Federrath. Mind the cache: large-scale explorative study of web caching. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 2497–2506, 2019.

[91] Hoai Viet Nguyen, Luigi Lo Iacono, and Hannes Federrath. Your cache has fallen: Cache-poisoned denial-of-service attack. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1915–1936, 2019.

[92] George Nomikos and Xenofontas Dimitropoulos. traIXroute: Detecting IXPs in traceroute paths. In *Passive and Active Network Measurement (PAM)*, 2016.

[93] Georgios Nomikos, Vasileios Kotronis, Pavlos Sermpezis, Petros Gigis, Lefteris Manassakis, Christoph Dietzel, Stavros Konstantaras, Xenofontas Dimitropoulos, and Vasileios Giotsas. O Peer, Where Art Thou?: Uncovering Remote Peering Interconnections at IXPs. In *ACM Internet Measurement Conference (IMC)*, 2018.

[94] Mark O'Neill, Scott Ruoti, Kent Seamons, and Daniel Zappala. TLS proxies: Friend or foe? In *ACM Internet Measurement Conference (IMC)*, pages 551–557, 2016.

[95] Chiara Orsini, Alistair King, Danilo Giordano, Vasileios Giotsas, and Alberto Dainotti. BGPStream: a Software Framework for Live and Historical BGP Data Analysis. In *ACM Internet Measurement Conference (IMC)*, 2016.

[96] Vivek S Pai, Limin Wang, KyoungSoo Park, Ruoming Pang, and Larry Peterson. The dark side of the web: an open proxy's view. *ACM SIGCOMM Computer Communication Review*, 34(1):57–62, 2004.

[97] Diego Perino, Matteo Varvello, and Claudio Soriente. Proxytorrent: Untangling the free http (s) proxy ecosystem. In *Proceedings of the 2018 World Wide Web Conference*, pages 197–206, 2018.

[98] ProxyRack. https://www.proxyrack.com/, 2022.

[99] Yakov Rekhter, Tony Li, and Susan Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271, 2006.

[100] Eric Rescorla. Rfc2818: Http over tls, 2000.

[101] RIPE Atlas. https://atlas.ripe.net.

[102] RIPE Geoloc. https://stat.ripe.net/widget/geoloc.

[103] RIPE RIS. https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris.

[104] Root DNS Servers. http://www.root-servers.org/.

[105] Route Views Project. http://www.routeviews.org/.

[106] Scikit-Learn: Machine Learning Library for the Python. http://scikit-learn.org/.

[107] Will Scott, Ravi Bhoraskar, and Arvind Krishnamurthy. Understanding Open Proxies in the Wild. *Chaos Communication Camp*, 2015.

[108] Philippe Skolka, Cristian-Alexandru Staicu, and Michael Pradel. Anything to hide? studying minified and obfuscated code in the web. In *The World Wide Web Conference*, pages 1735–1746, 2019.

[109] Kurt Thomas, Elie Bursztein, Chris Grier, Grant Ho, Nav Jagpal, Alexandros Kapravelos, Damon McCoy, Antonio Nappa, Vern Paxson, Paul Pearce, et al. Ad injection at scale: Assessing deceptive advertisement modifications. In *2015 IEEE Symposium on Security and Privacy*, pages 151–167, 2015.

[110] Altug Tosun, Michele De Donno, Nicola Dragoni, and Xenofon Fafoutis. Resip host detection: Identification of malicious residential ip proxy flows. In *2021 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–6. IEEE, 2021.

[111] traIXroute. https://github.com/gnomikos/traIXroute, 2018.

[112] Giorgos Tsirantonakis, Panagiotis Ilia, Sotiris Ioannidis, Elias Athanasopoulos, and Michalis Polychronakis. A Large-scale Analysis of Content Modification by Open HTTP Proxies. In *Network and Distributed System Security Symposium (NDSS)*, 2018.

[113] Gareth Tyson, Shan Huang, Felix Cuadrado, Ignacio Castro, Vasile C Perta, Arjuna Sathiaseelan, and Steve Uhlig. Exploring http header manipulation in-the-wild. In *International Conference on World Wide Web (WWW)*, pages 451–458, 2017.

[114] Limin Wang, KyoungSoo Park, Ruoming Pang, Vivek S Pai, and Larry L Peterson. Reliability and Security in the CoDeeN Content Distribution Network. In *USENIX Annual Technical Conference (ATC)*, pages 171–184, 2004.

[115] Nicholas Weaver, Christian Kreibich, Martin Dam, and Vern Paxson. Here be web proxies. In *International Conference on Passive and Active Network Measurement*, pages 183–192, 2014.

[116] Lan Wei and John Heidemann. Does Anycast Hang up on You? In *Network Traffic Measurement and Analysis Conference (TMA)*, 2017.

[117] Zachary Weinberg, Shinyoung Cho, Nicolas Christin, Vyas Sekar, and Phillipa Gill. How to catch when proxies lie: Verifying the physical locations of network proxies with active geolocation. In *Proceedings of the Internet Measurement Conference 2018*, pages 203–217, 2018.

[118] Zachary Weinberg, Mahmood Sharif, Janos Szurdi, and Nicolas Christin. Topics of controversy: An empirical analysis of web censorship lists. *Proceedings on Privacy Enhancing Technologies*, 2017(1):42–61, 2017.

[119] Mengjun Xie, Indra Widjaja, and Haining Wang. Enhancing cache robustness for content-centric networking. In *2012 Proceedings IEEE INFOCOM*, pages 2426–2434, 2012.

[120] Xing Xu, Yurong Jiang, Tobias Flach, Ethan Katz-Bassett, David Choffnes, and Ramesh Govindan. Investigating transparent web proxies in cellular networks. In *International Conference on Passive and Active Network Measurement*, pages 262–276. Springer, 2015.

[121] Tarun Kumar Yadav, Akshat Sinha, Devashish Gosain, Piyush Kumar Sharma, and Sambuddho Chakravarty. Where the light gets in: Analyzing web censorship mechanisms in india. In *Proceedings of the Internet Measurement Conference 2018*, pages 252–264, 2018.

[122] Hui Zhang, Ashish Goel, and Ramesh Govindan. An empirical evaluation of internet latency expansion. *ACM SIGCOMM Computer Communication Review*, 35(1), January 2005.

[123] Mingming Zhang, Baojun Liu, Chaoyi Lu, Jia Zhang, Shuang Hao, and Haixin Duan. Measuring privacy threats in china-wide mobile networks. In *8th {USENIX} Workshop on Free and Open Communications on the Internet ({FOCI} 18)*, 2018.

[124] Yihe Zhang, Hao Zhang, Xu Yuan, and Nian-Feng Tzeng. Pseudo-honeypot: Toward efficient and scalable spam sniffer. In *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 435–446, 2019.

Appendix

PERMISSIONS

The anycast research presented in Chapter 2 was previously published in the ACM journal *SIGCOMM Computer Communication Review*, Volume 49, Issue 3, pp. 18-25. https://doi-org.udel.idm.oclc.org/10.1145/3371927.3371930. The inclusion of this material in the dissertation is in accordance with ACM's author rights, which is available at https://authors.acm.org/author-services/author-rights, and with the terms of the Copyright Transfer, i.e., "*Authors can include partial or complete papers of their own (and no fee is expected) in a dissertation as long as citations and DOI pointers to the Versions of Record in the ACM Digital Library are included.*"

The research presented in Chapter 3 was previously published in the Elsevier journal *Computer Networks*, Volume 208, 8 May 2022, 108893. https://doi.org/10.1016/j.comnet.2022.108893. The inclusion of this material in the dissertation is in accordance with the Elsevier's Guide for Authors, which is available at https://www.elsevier.com/journals/computer-networks/1389-1286/guide-for-authors, i.e., "*In general, an author should not submit for consideration in another journal a paper that has been published previously, except in the form of an abstract or as part of a published lecture or academic thesis or as an electronic preprint.*"

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346 USA