2019-09

SUNUP: ICMP TIMESTAMP BEHAVIORS IN FINGERPRINTING

Kvitchko, Terrence

Monterey, CA; Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

SUNUP: ICMP TIMESTAMP BEHAVIORS IN FINGERPRINTING

by

Terrence Kvitchko

September 2019

Thesis Advisor:	Robert Beverly
Second Reader:	Erik Rye (USNA)

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2019	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
SUNUP: ICMP TIMESTAMP BEHAVIORS IN FINGERPRINTING

5. FUNDING NUMBERS

6. AUTHOR(S) Terrence Kvitchko

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

Previous work on a subset of the internet has identified a number of unique behaviors in ICMP timestamp responses. In this study we expand to studying the entire routable IPv4 address space, surveying ICMP timestamp responsiveness and the prevalence of specific response behaviors. We introduce a new behavior called "uptime" to the existing behavioral classification taxonomy. Additionally, we combine banner-grabbing scan data with our ICMP response classifications to evaluate the utility of ICMP timestamp behaviors in device fingerprinting.

14. SUBJECT TERMS
ICMP

15. NUMBER OF PAGES
103

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

SUNUP: ICMP TIMESTAMP BEHAVIORS IN FINGERPRINTING

Terrence Kvitchko
Civilian, CyberCorps, Scholarship for Service
BS, University of California, Merced, 2017

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2019

Approved by: Robert Beverly
Advisor

Erik Rye
Second Reader

Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Previous work on a subset of the internet has identified a number of unique behaviors in ICMP timestamp responses. In this study we expand to studying the entire routable IPv4 address space, surveying ICMP timestamp responsiveness and the prevalence of specific response behaviors. We introduce a new behavior called “uptime” to the existing behavioral classification taxonomy. Additionally, we combine banner-grabbing scan data with our ICMP response classifications to evaluate the utility of ICMP timestamp behaviors in device fingerprinting.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

AS	Autonomous System (networking)
BAC	Building Automation and Control
CAIDA	Center for Applied Internet Data Analysis
CPE	customer-premises equipment
CSAIL	Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory
CSV	comma-separated value
CVE	Common Vulnerabilities and Exposures
CWMP	CPE WAN Management Protocol
DOS	denial of service
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
IoT	internet of things
IP	Internet Protocol
ITDK	Internet Topology Data Kit
MAC	media access control
NPS	Naval Postgraduate School
OUI	organizationally unique identifier

RFC Request For Comment

SSH Secure Shell Protocol

TCP Transmission Control Protocol

UDP User Datagram Protocol

UTC Coordinated Universal Time

WAN wireless area network

Acknowledgments

I would like to thank Robert Beverly and Erik C. Rye for their enthusiasm, patience, support, and good humor. I would also like to thank Alex, Alexis, Ryan, and Tony for providing moral support and putting up with me in the throes of thesis writing.

This material is based on work supported by the National Science Foundation under Grant No. 1565443. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

In computing, fingerprinting is a way to identify a computing entity (such as a device or file) by using only a restricted amount of data. Fingerprinting has a wide range of purposes, including network surveys and identification of vulnerabilities. As such, many fingerprinting methods and tools exist—for example, Nmap is a tool which identifies operating systems through IP and TCP probing [1]. Fingerprinting generally leverages implementation-specific differences, such as slightly different file contents or responses to unanticipated situations.

Traditional active fingerprinting techniques focus primarily on operating systems and end nodes, and can be less effective for devices like routers. Fortunately, there remain promising options for such devices. One is the long-standing internet protocol ICMP due to its presence on all networked IPv4 devices.

Recent work [2] has discovered a new form of information leakage in a disused feature of ICMP—timestamp messages. Devices respond to timestamp requests with a number of discrete, classifiable behaviors rather than one consistent standard. This work seeks to build on this discovery and understand whether different devices, manufacturers, or operating systems can be fingerprinted through these distinctive behaviors.

1.1 Internet Control Message Protocol (ICMP)

ICMP is a baked-in feature of the Internet Protocol (IP). It is used primarily to relay error and diagnostic messages in the IP environment, such as for failed transmissions and router discovery/selection. All IP-participating nodes must implement ICMP in order to allow for proper connectivity and participation in higher-level protocols and services [3], including not only the World Wide Web but also internal services like Active Directory. Furthermore, many basic software utilities rely on ICMP messages, such as ping and traceroute [4].

Some network administrators may want to opt out of ICMP in order to more stringently secure their networks. Primarily this is because ICMP pings allow for an easy, low-

bandwidth device discovery by prompting them to respond with an ICMP echo message. This capability has been used for full-scale surveys of the internet [5], but can be deployed equally well by an adversary to find attack surfaces. Similarly, traceroute returns a list of router hops a packet went through to reach a certain IP address; administrators may be wary of letting potential adversaries discover network topologies. ICMP is easily spoofed, allowing for covert reconnaissance or denial of service (DOS) attacks [6]. ICMP can even be used for tunneling of arbitrary information [7].

However, it is impractical for internet-connected hosts to opt entirely out of ICMP in IPv4.[1] Pings allow a network administrator to easily determine whether a device of their own is up. ICMP messages communicate where a route may be broken, or whether a packet is taking a route it should not. They report when packets are too large for a link to support and must be fragmented; this same capability can be used to discover the largest permissible packet for a certain route [3]. Without ICMP, network troubleshooting loses efficacy, and communication may fail altogether.

At best a firewall can be configured to block certain types of messages, or restrict specifically out- or inbound packets. Many network administrators neglect to do even this.

Among the lesser-known—but not yet deprecated—ICMP message types are timestamp and timestamp reply messages. As a broad description, timestamp messages send out an originating timestamp and expect two timestamps in response from the recipient. We further detail this feature in section 2.1.

This functionality was previously used in the Internet Clock Service [8] for clock synchronization and one-way delay measurements; it has long-since been replaced by the UDP-based Network Time Protocol [9] in common usage. In practice, there is little legitimate usage today, and when left open to arbitrary hosts it is listed in MITRE'S Common Vulnerabilities and Exposures (CVE) as CVE-1999-0524 [10]. Allowing ICMP timestamp messages leads to information leakage and associated vulnerabilities, among them the ability to attack random number generators seeded with a device's current time [10].

[1] ICMP is even more crucial to IPv6, but the updated IPv6 version of ICMP does not implement timestamps in the same way; as such IPv6 is excluded from this study.

1.2 Scope

Surprisingly, Rye and Beverly found that 15% of a probed subset of internet devices do respond to ICMP timestamp messages [2]. This is more than we expected given warnings like CVE-1999-0524, and the lack of legitimate usage for the feature. Thus, we seek to extend previous work by taking a full scan of the IPv4 internet and determining what overall percentage is responsive.

Additionally, we will classify all responsive IP addresses into a taxonomy previously developed in Rye and Beverly's work [2]. We will use banner and header data from application-layer protocols in order to identify addresses, then analyze them for correlations between identity and behavior. With this information, we will seek to determine whether device fingerprinting is possible from ICMP timestamp behaviors.

1.3 Summary of Findings

- 6% of routable IPv4 addresses respond to ICMP timestamp requests.
- Only 30-35% of routable IPv4 addresses respond to ICMP timestamp requests in a way conformant to RFC 792 [3].
- A majority of unique device models (as implied by unique banner responses) are at least 50% consistent in responding with only a single timestamp behavior. Some are 100% consistent.
- Certain timestamp behaviors are only seen in devices from one manufacturer, and certain models of device only show a single behavior (as implied by banner responses).
- A newly described behavior known as "uptime" shows up in about 4% of ICMP timestamp responsive devices.

1.4 Thesis Structure

1. Chapter 1 introduces fingerprinting, ICMP timestamps, and the scope of our research.
2. Chapter 2 discusses related works, including summarizing Sundial's taxonomy for ICMP timestamp behaviors, and gives a background in the kinds of data we will be collecting.
3. Chapter 3 describes challenges we faced and summarizes our methodology, including the ways we collected data, reduced complicating factors, and chose boundaries for

our analysis.

4. Chapter 4 relays our results and our analyses, including both the prevalence of ICMP timestamp responses and the consistency of behaviors among responses and manufacturers.
5. Chapter 5 discusses our conclusions and ideas for future work.

CHAPTER 2:
Background

This chapter discusses ICMP timestamps in more detail, summarizes previous works on ICMP timestamp fingerprinting, describes and enumerates ICMP timestamp behaviors, and covers message banners and headers used for identification.

2.1 ICMP Timestamps

Both timestamp and timestamp reply messages consist of eight fields.

The first five fields of a timestamp message are all headers. The type field specifies whether this is a request (13) or a reply (14). ICMP timestamps have no associated codes, so the code should always be zero. The checksum field is the Internet Checksum [11], and is itself treated as zero when its contents are being calculated. Both the id and sequence are intended for use by the request sender and should be duplicated, unchanged, by the respondent.

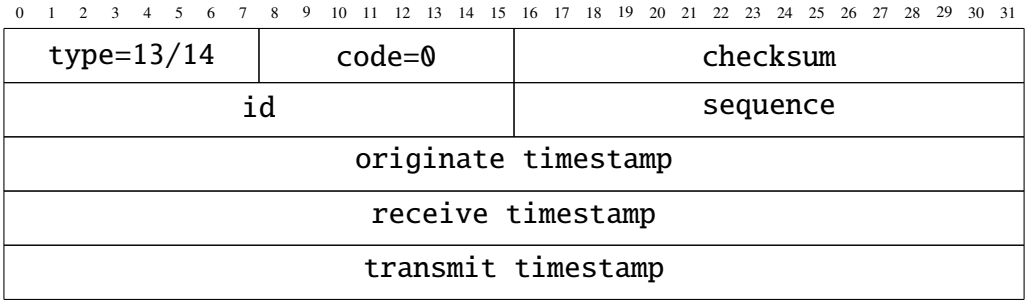


Figure 2.1. ICMP Timestamp Message Fields.
Adapted from [2], [3].

The last three fields of a timestamp message contain timestamps. The first timestamp is an Originate Timestamp, filled in by the timestamp requester when they last interact with the message. The next two timestamp fields are intended to be filled out by the respondent; the Receive Timestamp upon the first time they interact with it, and the Transmit Timestamp upon the last time before the message is sent back. The timestamps are intended to be

in milliseconds since midnight Coordinated Universal Time (UTC), but can be substituted with another format if the high order bit of the timestamp is set [2], [3].

2.2 Related Works

A number of previous studies investigated fingerprinting with ICMP timestamps. Notable examples of this include Kohno et al.'s work on remote physical device fingerprinting [12] and Cristea and Groza's work on fingerprinting smartphones [13].

However, neither of these are applicable to behavioral fingerprinting. Kohno et al. relied upon clock skew, a phenomenon in which device clocks do not progress at the exact same rate due to miniscule variations in hardware (rather than implementation). They also focused primarily on Transmission Control Protocol (TCP) timestamps rather than ICMP, despite outlining a possible method for the latter. Cristea and Groza reliably differentiate between individual devices by their clock skew. They do not, however, examine differences between operating systems, manufacturers, and product models. Additionally, their method has only been tested on smartphones and was not tested in measurement periods of less than 15 minutes.

Furthermore, all previous studies of ICMP timestamps we are aware of analyze comparatively small amounts of public nodes, even when they do look into ICMP timestamp responsiveness. Anagnostakis et al. probed 400k routers, and reported a 93% success rate in eliciting timestamp responses [14]; this work was published in 2003, over 15 years ago relative to this study, and before network security was nearly as emphasized as it is in the modern day. Buchholz and Tjaden investigated a list of approximately 8k web servers over a period of six months when studying clock management and reported a 41% response rate to ICMP requests [15]; their work was published in 2007. The Information Sciences Institute performs ICMP ping scans of the IPv4 address space every few months, but does not test other ICMP message types [5]. The only work which studies ICMP timestamp responsiveness in the modern internet, within the past few years, is Rye and Beverly's 2019 "Sundial" work. Sundial scanned 14.5 million hosts across the internet, including one address for every /24 routable subnet [2]. It elicited responses from approximately 2.2 million, 15% of what was probed.

Sundial made an additional contribution to the field of ICMP timestamp research by focusing

on response behavior rather than clock skew. Although there is a defined expected behavior in RFC 792 [3], Sundial discovered many different implementations and behaviors in practice.

This work seeks to make two improvements on Sundial work. First, we introduce "Sunup." Sunup scans the full routable IPv4 space for ICMP responsiveness and behavior inference. Second, Sunup seeks to compare ICMP timestamp behavior classifications and header responses from scanned IP addresses to see if this data is useful for fingerprinting.

2.3 Sundial Behavioral Taxonomy

The ICMP timestamp behaviors discovered by Sundial are believed to be due to differing implementations on the routers and end devices involved. As ICMP timestamps are not deprecated, operating systems are required to implement them. However, they are a functionally unused feature that are likely to be left unmaintained. As a result, a device's implementation of these ICMP messages can be neglected even when it results in the device being non-conformant to RFC standards [3].

Regardless of the cause, devices ("respondents") are classifiable into a taxonomy of one or more groups based on their responses to four specific ICMP timestamp requests (sent by a "prober").

These behaviors are as follows, with some added or reorganized since Sundial [2]:

1. **Correct**: A behavior that is conformant to RFC 792. The receive and transmit timestamps are both in milliseconds, and are both nonzero except at the appropriate time (midnight UTC). They may have identical timestamps if otherwise conformant, given we make the assumption that processing time took less than a millisecond. Denoted in results as "correct."
2. **MSB-Correct**: Timestamp values have the most significant bit set, but otherwise conform to correct behavior. Denoted in results as "correctMSB."
3. **MSB**: A response that has the most significant bit set. Responses that do not provide a timestamp relative to midnight UTC or do not provide it in milliseconds are still allowed by RFC 792, as long as they have the most significant bit set. Denoted in results as "msb."

4. **Lazy**: Receive and transmit timestamps are equal due to the device simply duplicating the two, or due to genuinely processing a packet in under a millisecond, as assumed in correct. As there is no way of differentiating these two groups with our data, devices with identical timestamps—with the exception of those stuck at 0 or 1—are always classified as lazy. Denoted in results as "lazy."
5. **Normal**: A behavior which responds with receive and transmit timestamps that are not identical, and are also non-zero—i.e., not lazy and potentially correct. Denoted in results as "normal."
6. **Checksum-lazy**: A response is provided even when the request's checksum is incorrect. Denoted in results as "checksumLazy."
7. **Stuck**: The respondent always replies with the same values regardless of input and when requests are sent. (This includes subsets that only respond with 0, 1, or little-endian 1.) Denoted in results as "stuck," "stuck0," "stuck1," and "stuckLE1."
8. **Reflection**: The respondent duplicates whatever is in the receive and transmit timestamps in the request. Denoted in results as "reflect."
9. **Linux htons() bug**: Replies are truncated to a 16-bit value. Denoted in results as "buggy."
10. **Uptime**: Respondents that appear to be responding with their uptime rather than time relative to UTC. Denoted in results as "uptime."
11. **Millisecond**: Respondents which reply with time in milliseconds, as evaluated based on two different responses. Denoted in results as "millisecond."
12. **Second**: Respondents which reply with time in seconds, as evaluated based on two different responses. Denoted in results as "second."
13. **Timezone**: Encode timestamps which appear to be relative to midnight in a non-UTC time. Denoted in results as "timezone."
14. **Epoch reference**: Encode timestamps relative to the Unix epoch. Not present in results.
15. **Little-Endian**: Receive and/or transmit timestamps in little-endian four-byte integers, but otherwise correctly. Denoted in results as "correctLE."
16. **Unknown**: Any responses that cannot be classified into the above types. Denoted in results as "unknown."

Some devices display a certain behavior only in the second of the three timestamps, or only

in the third. Such responses can be denoted with "Rx" for the second and "Tx" for the third. Note also that groups are not necessarily mutually exclusive—e.g., a device can send a packet which is simultaneously a "timezone" and a "checksumLazy" packet.

2.4 Banners and Protocols

As this work is an attempt to see if fingerprinting via ICMP behaviors is possible, we collect a number of headers/banners from scanned IP addresses. These are retrieved through Censys [16], an internet-wide scanning service that provides free data access to researchers.

Banners are messages presented to a user upon interacting with a device on a specific port/through a specific protocol. Headers, on the other hand, tend to be at the beginning of complete segments of network data—they contain metadata about the packet or message. Both leak information which can itself be used for fingerprinting, such as manufacturer or model names. However, their reliability is not absolute; banners can be modified by the owner of a device, and headers can be entered incorrectly or deliberately manipulated.

If a device can be fingerprinted through banners and responds to timestamp requests, and all devices of that model consistently show the same behavior, we can tentatively conclude the behavior is a trait associated with the model. Then this ICMP behavior can be used for fingerprinting when banners are not present or are deliberately modified.

Certain headers/banners are responses to commonly used protocols, but others are more uncommon. The full list of header/banner types collected is as follows:

1. **File Transfer Protocol (FTP)**: A common protocol used for transferring files which requires two port connections: one for control and one for the data. FTP responds with a banner on the first connection, generally asking for login and authentication.
2. **Secure Shell Protocol (SSH)**: A common protocol used for secure remote logins and command line access. It generally responds with the network utility a server is using to run SSH.
3. **Telnet**: An older protocol generally used for remote command line access, Telnet is less secure than SSH. It is intended more broadly for unencrypted communication and generally responds with a login prompt.
4. **Hypertext Transfer Protocol (HTTP)**: Used primarily for supporting the web and

providing web content. HTTP headers provide a wealth of information useful for web functionality, most of which is less useful for fingerprinting. In this case, just the server portion of the HTTP header was collected.

5. **CPE WAN Management Protocol (CWMP)**: A protocol for communicating between configuration servers and customer equipment, i.e., equipment used by a subscriber to carrier, such as telephones and routers. CWMP authentication responses were collected.
6. **BACnet**: A protocol for communication with BAC systems, such as heating, lighting, and access control. BACnet response metadata was collected—namely the manufacturer and product fields.

These banners were selected due to commonly presenting identifying information when they exist. e.g., FTP and Telnet headers often mention the model of the device the protocols are hosted on, and the two BACnet fields, when present, implicitly include this information.

CHAPTER 3:
Methodology

Our methodology involves the collection of timestamp responses, classification, and combination with Censys data. We discuss challenges in analysis, including manufacturer inconsistencies, banner noisiness, and IP churn. We describe methodological measures taken to overcome these challenges. Additionally, we leverage public IEEE data on manufacturers for use in classification.

3.1 Data Collection

We perform ICMP timestamp scanning ourselves in order to accommodate for behavior classification probing—namely, four ICMP timestamp requests, all with different parameters. All IP scanning aside from scans for the fourth data set used the existing "sundial" ZMap [17] module [18].

Four sets of data were collected for analyses. The first group, described as the "primary" group, included all IP addresses from the entire routable IPv4 space which responded to ICMP timestamps. The second set focused on all IPs from the IPv4 space which displayed uptime behavior. The third set focused on routers with uptime behavior. Finally, the fourth set was a smaller subset of the routable IPv4 space, collected within approximately a single day.

Dataset	**Contents**	**Date Initiated**	**# of IPs (thousands)**
Primary	Full IPv4 scan	March 1, 2019	197415.2
Second	Uptime scan (primary set)	May 11, 2019	8255.5
Third	Uptime scan (ITDK)	December 3, 2018	88.9
Fourth	One-day IPv4 scan	August 22, 2019	34.1

Table 3.1. Dataset Summary.

In scanning, we followed ethical guidelines previously used in Sundial work; probing was done at comparatively slow rates, distributed in time and among networks, and used only

ICMP packets. These measures were taken to avoid appearing as attack traffic or noticeably degrading service on any networks. We also provided a method for opting out of scans and coordinated with local network administrators.

3.1.1 Primary set

Scanning for the primary group of data was done in two stages, and was performed from CSAIL's Advanced Network Architecture lab, located at the Massachusetts Institute of Technology. The vantage point is connected by Ethernet cable to the local area network, which resides behind a firewall appliance. While the vantage has a publicly routable IP address and thus requires no NAT, firewall state maintenance limited experiment probe rate to several thousand packets per second.

The first stage sent only a single ICMP timestamp request of the "standard" form discussed in Sundial [2]. Scanning commenced on January 16, 2019, with requests sent at a rate of 1000 packets per second, and ended on February 28, 2019. The intention was to narrow down the scope of of IPs that were responsive and minimize the number of total probes required. Non-responsive addresses were discarded prior to the second stage.

In the second stage of scanning, the remaining addresses underwent the full series of four requests each. Each type of request was performed in an individual round of probing, in the following order: "standard," "bad clock," "bad checksum," and"duplicate timestamp." Rounds were seeded with an identical seed and thus followed the same order of addresses after initial randomization; a round needed to complete before the next one launched. This stage ran from March 1, 2019 to March 16, 2019, and requests were sent at the same rate as the first stage.

Scan data was classified using the previous Sundial methodology. Responses to the standard probe identified whether a device was normal, lazy, non-UTC, and/or little-endian. Stuck and reflection devices were identified by standard and duplicate probes. Checksum-lazy behavior was detected by the bad checksum probe, and buggy behavior was identified by standard and bad clock probes. Any two probes were suitable for detecting time units used, and any single probe was sufficient for timezones [2].

Note that this did not classify uptime addresses, which would have been spread through

other classifications.

3.1.2 Second set: Uptime

The second data set was the group of IP addresses—using addresses from the primary set—that were determined to have uptime behavior. This was detected in two stages: first, any address that was classified as correct from the primary set was discounted. The remaining addresses were probed 5 times over a period of 24 hours, starting May 11, 2019, and had to continuously increase their timestamps at a steady rate to be classified as uptimers. They were also required to respond with at least one timestamp greater than milliseconds possible in UTC, but not report a time relative to the Unix epoch.

Probing is ongoing, run from an Amazon AWS instance in "US-East;" it repeats approximately every 15 minutes and runs at a rate of 10500 packets per second.

3.1.3 Third set: ITDK-Uptime

The third data set is the ITDK-Uptime set. Data from CAIDA's inferred router IP aliases from March 2018 [19] was cut down to one address each, for a result of 2,586,711 unique addresses. These addresses are interesting because they are drawn specifically from router-level topologies, and can be used in comparing routers and the overall IPv4 space.

Classification probing began on December 3, 2018, and was performed in two stages—in the first, anything that gave a seemingly correct response to an initial probe was eliminated. In the second stage, remaining IP addresses were probed 5 times each, with probes 6 hours apart. This covered a 24-hour time period for each unique address. Anything that was defined as uptime was kept for the third stage.

The third stage is ongoing; ITDK-Uptime addresses are probed approximately every 15 minutes, with a rate of 2000 packets per second. Theoretically, ITDK-uptimer addresses should be a subset of the overall uptimers in section 3.1.2.

3.1.4 Fourth set: August 22

The final data set was retrieved starting mid-day of August 22, 2019, and probed a random sample of 215k IP addresses that responded to the first stage of 3.1.1. It performed all four

timestamp requests on a single IP address before moving on to the next; scanning was run through a custom C executable rather than Zmap to allow for this.

This scan was significantly slower than previous scans, taking approximately 30 hours total for about 0.1% of the addresses in section 3.1.1. The scanner was made to sleep for half a second between every address to avoid problems with dropped packets.

The fourth dataset was obtained for reasons to be detailed in section 3.5. Censys data for this scan was retrieved separately, on August 23, 2019.

3.1.5 Opt-outs

We received opt-out requests only for the ITDK-Uptime scan, the third data set. AS7922 opted-out on March 20, 2019, excluding 3,976 addresses, and one individual from AS4713 opted-out on June 25, 2019.

3.2 Challenges

Once classified and stripped to IP and classification, scan data was combined with Censys banner data.[2] Entries that existed in the scan data but not Censys were noted as ICMP-responsive but not present in Censys. Entries found in Censys but not the scan data were skipped.

Headers retrieved included FTP from port 21; SSH from port 22; HTTP from ports 80, 8080, 8888, and 16992; Telnet from ports 23 and 2323, CWMP from port 7547; and BACnet from port 47808. There were numerous complications in analyzing the resulting data.

3.2.1 Responsiveness and accuracy

Sundial probing was restricted to the four packets necessary for classification and did not itself scan for additional banners, in order to avoid making an already intrusive scan more so. Censys historical data was retrieved from the same date that the primary group's Sundial scan began.[3] This means classifications from late in the scan are a full 16 days behind the

[2]Censys data was stripped of newlines, null characters, and ~ (tilde) characters for easy CSV storage and manipulation.

[3]The fourth set of data described in section 3.1.4 was the one exception, with Censys historical data retrieved from August 23.

banners we associated with those IPs. The second and third sets of data are even further off of the Censys date. Because of this, we anticipate some IP churn to have occurred in the intervening time.

IP churn occurs when an IP address that previously pointed to one device is reassigned and now points to another. Generally this happens to dynamic addresses, which are meant to be transferred constantly as devices connect to and disconnect from a network. (This is in contrast to static addresses, which are intended to be tied to one device.)

In Xie et al.'s study of dynamic addresses [20], they analyzed a /16 block (a total of 65,536 addresses) and evaluated 7,045 as dynamic addresses—approximately 11%. In a "dynamic-heavy" sample of 155 million addresses that had email user logins, they evaluated that 75% were dynamic. They further investigated the volatility of these addresses, concluding that of dynamic addresses, only 15% have an inter-user time of 7 or more days; i.e., all but 15% take less than 7 days to be used by another device. This indicates dynamic addresses will be present and will be reassigned during our scan time.

It is important to note that Xie et al. stated that most dynamic addresses belong to consumer networks, and are used by personal computers or workstations in small enterprises. These devices rarely run services like HTTP or FTP, and will not respond with banners—as such, we assume that a number of these dynamic addresses will be immediately unidentifiable. However, there are exceptions—many of which were not present in 2007, the time of the study. Most residential gateways run HTTP for configuration purposes, as do a large amount of internet of things (IoT) devices, and the service is usually accessible on a standard port for ease of the user.

We expect to encounter churn both from these common exceptions and from the rarer static IP reassignment.

Furthermore, not all (or even a majority) of addresses had Censys entries on that date; this means that while they did respond to ICMP requests at the time Sundial probing occurred, they did not respond to any of Censys's queries on that scan date. In the primary and uptime sets, 31.5 million were found in Censys out of 197 million from the Sundial scan, approximately 16%—see Figure 3.1. In the ITDK group, 16.2k out of 88.9k were found.

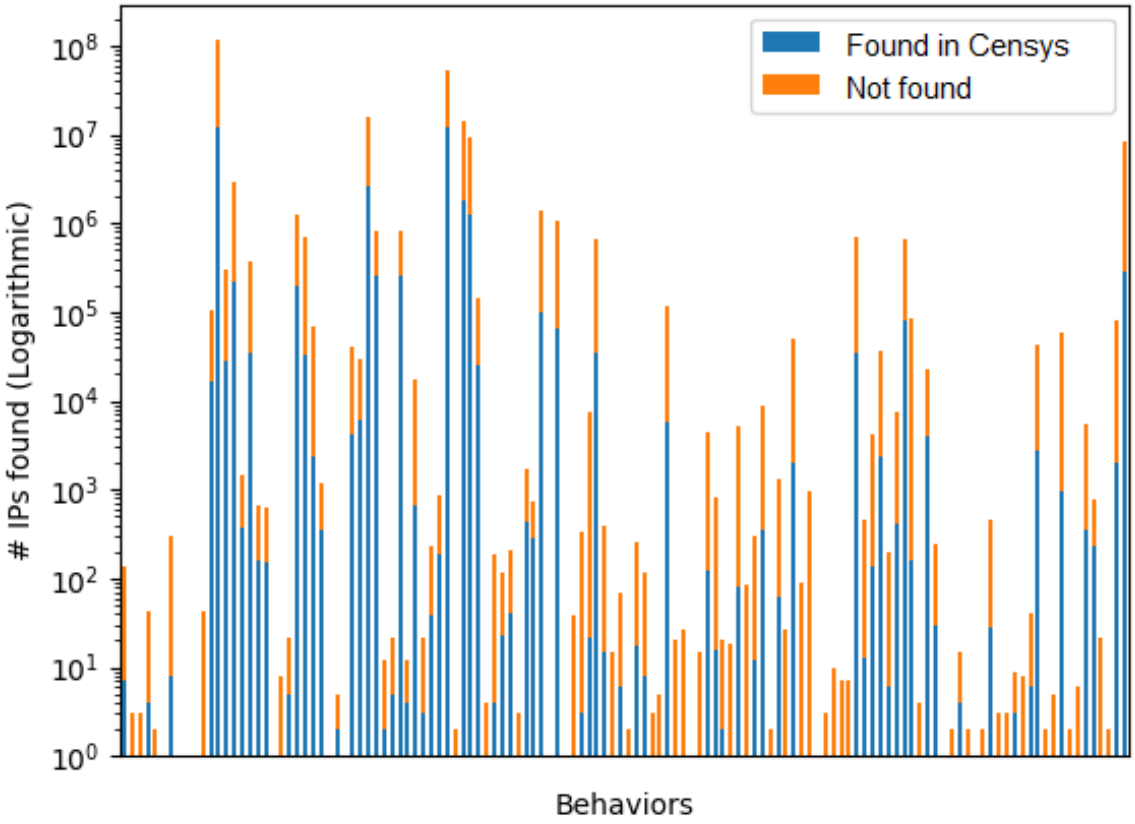


Figure 3.1. IPs Found in Censys Data, by Unique Combination of Behaviors.

Of addresses for which Censys had an entry, some had no responses to any of the protocols we were interested in. Of 78.8 million unique addresses that were found overall in Censys—not all of which corresponded to Sundial scan results—26.7 million were functionally blank. This rendered identification of an IP address impossible for this study.

Of course, when IP addresses had responses, they did not have them for each individual protocol. For example, of the 31.5 million in the primary set 3.6 million responded to FTP, while only 3.6k responded to BACnet.

3.2.2 Noisiness and applicability

The banners gathered had significant amounts of noise, and the variety of noise mechanisms made it difficult to match functionally-identical banners automatically. For example, the majority of CWMP banners had a unique nonce, FTP headers often reported active online

user counts, and several different headers commonly listed their own IP address.

Additionally, banners were not necessarily illuminating or internally consistent. Banners that did not immediately report model or manufacturer were not feasible to research by hand due to the volume of unique responses. The lack of banner standardization meant it was impractical to retrieve manufacturer and model names where a list of known names did not already exist. Some manufacturers lacked even a single standard way of printing their own name.

Also, when banners did list manufacturers, they were not necessarily internally consistent—more than one could be listed across the full array of banners. (For example, Asus could be listed in one banner, while Hewlett Packard was listed in another.) For manufacturers known for both hardware and software products, there was no practical way of differentiating whether a banner was indicating one or the other. This meant it was equally infeasible to determine which of the manufacturers reported by a device was responsible for the ICMP implementation.

For case studies of banner variety and clarity, see Appendix A.

3.3 Manufacturer Classification

In order to search banners for manufacturers, we first needed to obtain a list of potential manufacturers. For this we referred to the IEEE Registration Authority's list of organizationally unique identifier (OUI)s [21]. The purchase of an OUI provides the right to generate identifiers such as media access control (MAC) addresses, which are required for interfacing with networks. As such, nearly every vendor or manufacturer has ownership of an OUI. Furthermore, larger manufacturers require multiple such OUIs, as a single OUI can only cover 16.8 million network interfaces. Apple, for example, has 352 distinct OUIs as of 2016 [22].

Our OUI list was retrieved from linuxnet.ca [23] rather than directly from the IEEE Registration Authority. This version is sanitized, removing incorrect or unnecessary data artifacts and fixing spelling errors or vendor name inconsistencies.

For the purposes of our analysis, we decided to focus on "major" manufacturers, which we

defined as those which had purchased a minimum of ten OUIs under a (sanitized) identical name.[4] When filtered in this way, we obtained with a list of just over 100 names.

These names were sorted and subsidiaries were combined by hand (e.g., Sony Mobile Communications Inc and Sony Interactive Entertainment Inc. were grouped as Sony). Additionally, common variations of names were noted for searching purposes (e.g., Hewlett-Packard also had Hewlett Packard and HP listed as applicable hits) and shortened for maximum hits (e.g., Nintendo Ltd was reduced to Nintendo). Finally, MikroTik was added by hand, due to its prevalence in the previous Sundial study.

The final list of manufacturers resulted in 55 entities; see Appendix B.

3.4 Noise Reduction

Preliminary frequency analysis was done on all banner fields by classification—e.g., the frequency of a specific BACnet response was counted for each unique combination of ICMP timestamp behaviors. This initial analysis did not attempt to differentiate by noise and treated a number of functionally identical, but "noisy," banners as unique entries. Even so, it cut down significantly on the amount of data to be processed for individual protocols; 5.6 million lines of data were reduced to 0.9 million.

We considered using some form of fuzzy string matching to further cut down on noise. Unfortunately, upon examination of the data, we concluded that any fuzzy string matching algorithm would cut out differences we wanted to preserve as unique banners—such as model numbers—as well as those we wanted to discard.

As such, noise reduction was done through the use of regular expressions. Patterns matching the IP address format (including port, if they were immediately followed by a colon and port-sized number) were stripped and replaced with a generic substring marking their location. The same was done with hour-minute-second timestamps where fields were separated by colons (overwhelmingly the most common time format). Weekdays and their common contractions were stripped when followed by a delimiter (such as a white space). Similarly,

[4]Manufacturers with multiple subsidiaries were not filtered for at this stage. Unless a subsidiary had more than 10 OUIs it was not counted, e.g., a manufacturer with exactly 10 registered OUIs across two subsidiaries would have been excluded.

dates were stripped, including months in either numbered or named form, and including contractions.

Dates were the most difficult of the "standard" forms of noise to strip, due to the number of accepted date formats and orders. While all common formats were replaced, some small fraction of uncommon banners may retain dates and be counted separately.

Further manual examination led to more specialized cleanup. CWMP header responses regularly had (more or less) unique nonces, so all CWMP headers with nonces had them removed. Additionally, both FTP and Telnet had very common noisy strings in specific formats—e.g., FTP servers commonly responded with a banner including the phrase "You are user [number] of [number] allowed." Nullification of these artificial difference was also automated with regular expressions and string literal searches.

Final, reduced-noise frequency counts resulted in 0.4 million lines of data. This version of the data was used for all frequency analyses.

For specific regular expressions used in noise reduction, see Appendix C.

3.5 Counteracting IP Churn

In order to counteract any noise or inaccuracies caused by IP churn, two strategies were used.

First, minimum occurrence thresholds for banner inclusion were used in nearly all analyses. These thresholds applied by header/banner protocol. For example, if a minimum threshold of 3 was used for BACnet when checking responses for simultaneously lazy and checksumLazy IPs, any BACnet responses that occurred less than 3 times for that group were removed.

We assumed very rare banners for a behavior were theoretically due to IP churn replacing the devices originally at that address, where the new devices with those banners did not display the behavior. Thresholds were therefore used to eliminate these. Minimums were decided for each behavioral group based loosely on the size of the group, with the assumption that larger groups would experience more churn. (Thresholds also filtered entirely unique, customized banners out.)

Second, the entire fourth set of data described in section 3.1.4 and subsets of primary (3.1.1) and uptime (3.1.2) data were also put through the above noise reduction. The primary/uptime data subsets, henceforth referred to as March 1 data, covered approximately the first two-fifths of scanned addresses. The March 1 dataset diverged from the Censys scan's date at maximum 14 instead of 16 days. The 3.1.4 dataset, henceforth referred to as August 22 data, covered a small fraction of addresses that were probed with all 4 ICMP requests consecutively. As this scan concluded on August 23, Censys historical data for it was pulled from that date, and the August 22 dataset diverged less than a day at maximum.

We adopted the premise that scan data closer to the Censys data would have experienced less churn. Therefore, we decided significant differences between March 1 data, August 22 data, and full primary/uptime data would be noticeable if churn influenced results heavily.

CHAPTER 4:
Results

This chapter discusses our results, including how much of the routable IPv4 space we found to be ICMP timestamp responsive, the number and consistency of behaviors for unique banners, and the number of manufacturers that displayed specific behaviors. We summarize how many addresses were found in Censys for our analyses.

We also discuss differences between overall uptimers and (router) ITDK-uptimers, and compare data from partial Sundial scans to the full scan.

4.1 Responsiveness of Internet to Timestamp Requests

Responsiveness of the internet can be evaluated through the primary set mentioned in section 3.1.1. Although the IPv4 address space consists of 2^{32} total IP addresses, a number of these addresses are not publicly routable; they are allocated for private and service provider addressing [24], [25]. As such, our initial probe covered all routable IPs—approximately 3.7 billion.

218,651,485 addresses were responsive to the initial probe, a response rate of about 6%. This is a lower percentage than the original Sundial paper's timestamp responsive 15% [2]; however, this can be explained by Sundial's targets. The original study used a hitlist [5] covering the most ping-responsive address in each /24 network—i.e., those addresses that were most likely to reply to any ICMP messages. The full IPv4 scan covered addresses that were not as responsive.

Once an address responded to an initial ICMP timestamp probe, it was added to the list for the full four-probe Sundial scan [2]. Following Sundial scanning and classification, we ended with a list of 197,415,236 successfully classified addresses—including those classified as "unknown." The 21,236,249 missing addresses—about 9.7%—did not respond to any ICMP timestamp probes following the initial scan and were dropped from our study. (Possible reasons for this are discussed in Chapter 5.) Functionally speaking, however, this means approximately 5% of routable IPv4 addresses are responsive to ICMP timestamps

consistently enough to be classified by Sundial methodology. The occurrences of large behavioral groups can be seen in Table 4.1.

Behavioral Group	Total Addresses	% of Overall
stuck	66,704	0.04%
lazy	186,656,147	98.72%
msb	9,521,539	5.04%
normal	2,222,078	1.18%
checksumLazy	4,839,045	2.56%
correct	57,769,954	30.55%
timezone	2,374,256	1.26%
unknown	78,954	0.04%

Table 4.1. Broad Sundial Behavioral Groups Found in IPv4 Scan.

Note that there is overlap of behavioral groups—for example, something could be marked both lazy and correct.

Responses overwhelmingly displayed lazy behavior. This suggests that a large percentage of devices have a processing time of less than one of their chosen time units, that a large percentage of devices duplicate their receive and transmit timestamps, or that both are true. The original Sundial study discovered that the Linux and BSD operating systems both have a genuinely lazy implementation of ICMP timestamp responses [2], so this is unsurprising.

Only 30.55% of responses were classified as appearing correct. The union of correct and msb responses—i.e., responses that could theoretically be conformant—was 35.28%. This means a full 64.72% of ICMP timestamp responsive devices definitively did not conform to RFC 792 [3] and could have faulty implementations. Up to 1.26% of this may instead be due to misconfigured local times on devices, and an unclear amount may be due to clock drift and network congestion causing timestamps to fall outside of our margin of error. Additionally, this behavior could be caused by middleboxes, as originally theorized in Sundial [2].

There were no addresses that could be classified as “epoch” respondents in the IPv4-wide scan. This is interesting, as the original Sundial study—on an address space approximately

256 times smaller than Sunup—did find one epoch respondent. We theorize that either the device was taken down or that clock drift unsynchronized it from the Sundial scanner's time. Sundial's classification standards allow for an error margin of one second for epoch time and would fail to evaluate a device more than one second off.

We also checked for theoretical "echo" behavior, a behavior where the respondent echoed what it received in the originate timestamp into its receive and transmit timestamps. No occurrences of this behavior were found.

For specific behavioral classifications and exact numbers, see Appendix D.

4.1.1 Comparison to August 22 dataset

The August 22 dataset described in section 3.1.4 was not a full scan of the routable IPv4 internet, and therefore cannot be used to evaluate overall responsiveness. It did, however, use addresses from the ICMP timestamp single-probe—which may be a useful indicator of churn.

Of the roughly 218 million addresses that were initially responsive to a single probe in March, 215k were randomly selected. They resulted in 34,125 IP addresses responsive to the full four-probe scan, or about 15.9%; significantly less than the 90.3% found in March. This is potentially caused by IP churn over 4 months, by firewalls attempting to block network scans, or by a mix of both.

Broad behavioral groups in this scan are summarized in Table 4.2. They seem to be roughly comparable to the primary dataset, which is especially striking given the difference in magnitude. This may be an indicator that—given that the August 22 data should have experienced far less—IP churn *during* our 16-day Sundial run is not a highly influential factor in our data.

Behavioral Group	Total Addresses	% of Overall
stuck	64	0.02%
lazy	33,679	98.70%
msb	1,805	5.29%
normal	414	1.21%
checksumLazy	853	2.50%
correct	11,577	33.93%
timezone	799	2.34%
unknown	0	0%

Table 4.2. Broad Sundial Behavioral Groups Found in August 22 Scan.

4.2 Censys-ICMP Join

As previously discussed in section 3.2.1, only 16% of our classified IP addresses had entries in Censys at all. This means that out of ICMP-responsive addresses, a full 84%—or approximately 4% of the overall routable address space—were not Censys-responsive even though they were responsive to timestamp requests.

Notably, these addresses had not been responsive for at minimum a week—as per Censys, data on an IP address is removed if the address has not replied to probing for that long. We propose three explanations for the 84% of addresses not included.

First, these addresses may not respond on all the common ports Censys scans, but also not block ICMP. Second, this may be due to IP churn; an address could have been non-responsive during the time of Censys's scan, but responsive by the time of the Sundial scan. Third, the administrators of those addresses may have opted-out of Censys scans. Censys's opt-out mechanism involves dropping traffic from their /23 measurement subnet, and Censys provides no data on whether a lack of response is due to their traffic being dropped or to a device with closed ports rejecting packets.

If the first explanation is true, it seems to suggest either a much greater amount of dynamic IP addresses than Xie et al.'s study indicates [20], a large amount of static IP addresses that provide no public services on common ports, or a mixture of both.

Within Censys data for the full IPv4 address space on March 1, 2019, there were 26,743,002 addresses found that had no entries relevant to us—approximately 20.2% were not publicly running anything on the 10 ports we evaluated for common services.

Within Sundial data, of the 31,500,486 successfully found in Censys, there were 6,563,115 addresses that did not respond to the 10 ports we evaluated, approximately 20.8%.

The full IPv4 data Censys scans and the Sundial-Censys subset seem to have comparable activity on common ports. It must be noted that Censys addresses are those which are open to port scanning in general, as Censys does not retain an IP unless it responds on some port. This means the 6.5M Sundial addresses remaining must have a response on some other port. However, given they respond on a port at all, ICMP timestamp responsive addresses appear to be broadly representative of common ports in the IPv4 address space.

Of addresses found in Censys, incidences of response on specific ports were as shown in Table 4.3.

Service/Port	**Total Responsive Addresses**	**% of Addresses**
FTP (20)	3,571,167	11.3%
SSH (22)	5,912,382	18.8%
Telnet (23)	1,061,692	3.4%
HTTP (80)	16,407,985	52.1%
Telnet (2323)	47,897	0.2%
CWMP (7547)	3,506,160	11.1%
HTTP (8080)	1,214,261	3.9%
HTTP (8888)	445,039	1.4%
HTTP (16992)	2,857	<0.1%
BACnet (47808)	3,555	<0.1%

Table 4.3. Banner Frequency In ICMP Timestamp Responsive Addresses Found in Censys.

Note that there is overlap, so percentages will not add up to 100; i.e., certain addresses had HTTP running on all of the HTTP ports listed instead of just one of them, or were running more than one type of service. Note also that something running on a specific port does

not necessarily mean it was the standard protocol. A small subset of IPs ran protocols on non-standard ports, such as FTP on the Telnet port. Due to banner noisiness these were not filtered out and percentages were not measured.

4.2.1 Comparison to August 22 dataset

Of the 34,125 IP addresses in the August 22 dataset, only 1,760 were found in Censys. This is approximately 5.2%, a significant difference from the 16% for the primary dataset, and is something that should have been unaffected by IP churn or firewalls. However, the massive difference in magnitudes makes it difficult to say anything conclusive. Incidences of response are shown in Table 4.4.

462 of these available-in-Censys addresses, or 26.3%, had none of the banner responses we were evaluating. All but FTP, Telnet on the standard port 23, and HTTP on port 8080 had lower response rates, but only in CWMP was the response rate lowered significantly.

Service/Port	**Total Responsive Addresses**	**% of Addresses**
FTP (20)	225	12.8%
SSH (22)	311	17.7%
Telnet (23)	81	4.6%
HTTP (80)	884	50.2%
Telnet (2323)	3	<0.1%
CWMP (7547)	95	5.4%
HTTP (8080)	102	5.8%
HTTP (8888)	22	1.3%
HTTP (16992)	0	0%
BACnet (47808)	0	0%

Table 4.4. Banner Frequency In August 22 Addresses Found in Censys.

4.3 Uptimers

Of the 197,415,236 responsive IP addresses in the primary dataset, 8,255,481 were classified as uptime when scanned approximately two months later—roughly 4.18% of all responsive

addresses. Broad behavioral groups, as they were classified in the primary dataset, can be seen in Table 4.5.

About 59.7% of these were labeled as only lazy in the primary dataset. These, and the less than 1% of those labeled only normal, can probably be considered "true" uptimers. However, 98.2% of uptimers had lazy within their behavioral classification—i.e., including those that also had classifications such as checksumLazy. This number seems roughly similar to the whole IPv4 space.

A large subset of uptimers—3,200,546, or about 38.8%—had their most significant bit set. This deviates greatly from the 5.1% in the overall IPv4 space, and suggests that uptimers are aware their behavior is non-conformant, or that they may be using the most significant bit for another purpose.

Investigation of uptime behavior in the long-running probes demonstrates why msb behavior is so prevalent. Two different versions of uptime were noted. One of them treats timestamp fields as 32-bit unsigned integers, and therefore has the most significant bit set half of the time. This behavior wraps roughly every 49 days. The second behavior always has the most significant bit set, and uses the remaining 31 bits; it wraps roughly every 25 days.

Only 15 uptimers were classified in the primary dataset as unknown.

Broad Behavior	**% Overall**	**% checksumLazy**	**% msb**	**% checksumLazy,msb**
lazy	98.22%	0.69%	37.95%	0.17%
normal	1.78%	<0.01%	0.81%	<0.01%
unknown	<0.01%	0%	0%	0%

Table 4.5. Notable Sundial Behavioral Groups Found in Uptimers.

4.4 ITDK-Uptimers

Of 2,586,711 IP addresses in the ITDK list, 1,255,579 or 48.5% responded to ICMP timestamp probing. Even compared to the 218.7M (≈6%) that were initially responsive in section 4.1 rather than the final 197.4M (≈5%) pool, this is a huge difference. ITDK-surveyed routers appear to be more responsive to ICMP timestamp requests than the full

IPv4 space. Note that this should be taken with a grain of salt: ITDK routers are a subset of the IPv4 space that is already known to be traceroute-responsive [19]. However, this is expected behavior; at a minimum, routers are less likely to be behind a firewall.

In the 1.3M responsive ITDK addresses, 88,884 ITDK-uptimers were discovered. This is approximately 7.1%, a greater percentage than overall uptimers. Routers seem marginally more likely to be uptimers.

ITDK-uptimer IP addresses were compared to the addresses in the primary set in order to classify their behavior. 86,291 were found; 2,593 were not present at all. Broad behavioral groups of those that were found can be seen in Table 4.6.

Interestingly, when compared to the uptimers, a full 23,163 addresses were in the ITDK group but not among the uptime addresses. Given the ITDK group's status as a subset of overall IPv4 addresses, both should have been classified as uptimers. As with the August 22 set, the 2.6k missing addresses and the 23.2k non-uptime addresses may or may not be due to IP churn over several months. Addresses in either group may also have not responded to an initial probe and been dropped from the scan list.

35,547 ITDK-uptimers had their most significant bit set, or about 40.0%. This is quite similar to the overall uptimers. No ITDK-uptimers were classified as unknown.

Broad Behavior	**% Overall**	**% checksumLazy**	**% msb**	**% checksumLazy,msb**
lazy	90.03%	0.05%	39.93%	0.01%
normal	2.3%	0.01%	0.08%	<0.01%
unknown	0%	0%	0%	0%

Table 4.6. Notable Sundial Behavioral Groups Found in ITDK-Uptimers.

4.5 Behavior Volume in Unique Banners

All unique banners had a certain behavioral volume—a number of classifications that the banner appeared in. For example, if 100 IP addresses responded on the FTP port with the same banner, and half of them showed "lazy" behavior, while the other half were classified with both "lazy" and "checksumLazy" behavior, then the volume would be two.

Banners were measured separated by common protocol/port. Common protocols on different ports were combined—e.g., HTTP banners were combined across all four measured HTTP ports.

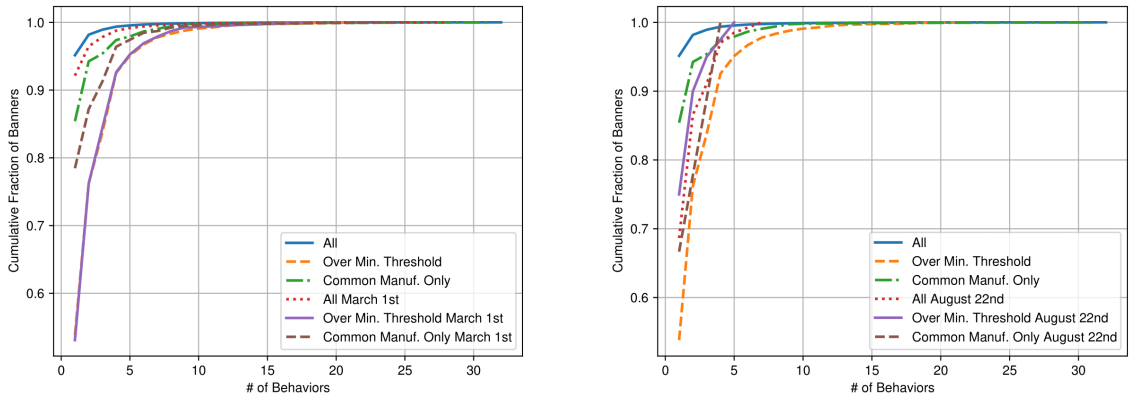
Behavioral volume was measured with a different threshold for each protocol. Thresholds were roughly proportional to the amount of responses a certain protocol had as seen in Fig 4.3, with a minimum threshold of two. They were intended to eliminate those banners that were categorized erroneously due to IP churn, as well as those banners that were customized singletons; any banners that had fewer instances than the threshold were discarded. Through this we hoped to estimate what proportion of banners were both accurately classified and possible to fingerprint. We made the assumption that protocols with more responses were likely to have more inaccurate responses due to IP churn and more customized unique responses, and so gave these protocols higher thresholds. Thresholds are listed in Table 4.7.

Protocol	**Threshold**	**Threshold August 22**
BACnet	2	2
CWMP	10	2
FTP	10	2
HTTP	50	2
SSH	18	2
Telnet	3	2

Table 4.7. Thresholds for Behavior Volume Analysis.

Finally, behavioral volume was separately measured for only those banners that listed a common manufacturer, where common manufacturers include those chosen through the method described in section 3.2.2. This was another attempt to estimate what proportion of banners was possible to fingerprint. The manufacturer-inclusion analysis did not use thresholds.

Results from the primary set (see section 3.1.1) were compared with both March 1 and August 22 results (see section 3.5) to further estimate the effect of IP churn. The BACnet protocol was the one exception: there were no BACnet results in the small August 22 dataset, so BACnet was only compared to the March 1 dataset.



(a) Primary and March 1

(b) Primary and August 22

Figure 4.1. Behavior Volume for FTP Banners

An example of results can be seen in Fig. 4.1. For graphic results for all protocols, see Appendix E.

In primary data, when banners below the threshold of required instances are allowed, a significantly higher percentage of banners have a behavior volume of one. This distribution is very pronounced in FTP banners, where—without the threshold of 10 instances—more than 90% of banners have a single behavior. With the threshold, the percentage shrinks to approximately 50%.

However, a deeper dive into the relevant data shows that a threshold of 2 would be nearly as effective. FTP banners which display a single behavior overwhelmingly do so because they only appear once; of 115,258 single-behavior FTP banners, 111,430 (or 98.7%) are singletons. Many of these singletons are customized banners and will only ever appear on one device, which means a single behavior by default.

This pattern appears to hold in all primary data banners except CWMP and BACnet. (CWMP has no single-behavior banners.) However, the threshold's effect is least pronounced in HTTP; HTTP also has only 53,995 single-behavior responses, and 42,241 of those responses (or 78.2%) appear only once. This is unsurprising, as HTTP is conventionally unlikely to have the server portion of its header uniquely customized compared to FTP, SSH, and Telnet. The latter three protocols often have banners that include welcome messages and

have no conventional format; HTTP's server section is meant only to identify the server software and has a conventional format [26].

March 1 data mirrors primary data in having proportionately fewer banners with a behavior volume of one after a threshold is applied. In August 22 banners, the pattern holds only in SSH and Telnet.

When thresholds are applied, primary and March 1 data have near-identical distributions in FTP, HTTP, and Telnet, a similar one in CWMP, and a partially similar one in SSH. This seems to indicate that after 14 days of potential churn have already passed, two days of IP churn are ineffectual enough to be eliminated with thresholds. August 22 data is not similar, but it is difficult to say whether this is due to less IP churn or simply a much smaller data set.

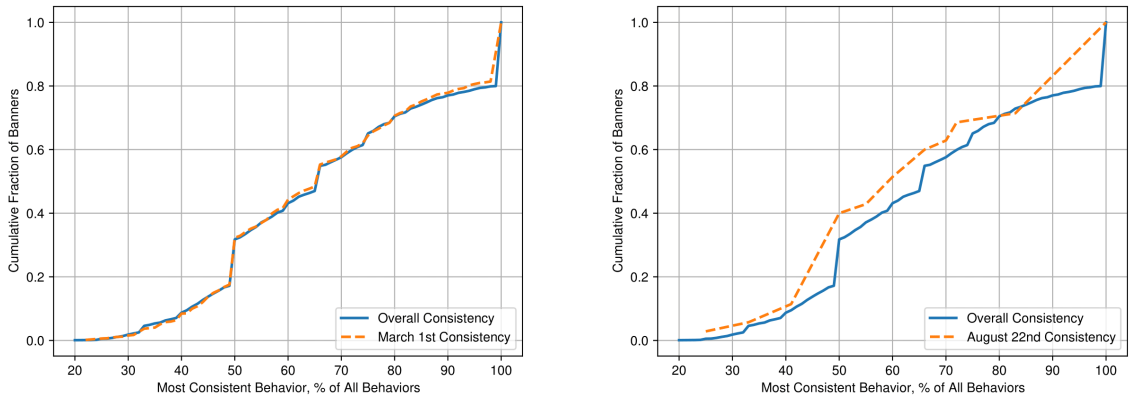
Requiring common manufacturers improves or roughly equals average volume in BACnet and Telnet, as well as August 22 CWMP and primary/March 1 HTTP. This is encouraging in the cases of BACnet and Telnet at a minimum, as it indicates that likely prospects for fingerprinting have more easily "identifiable" banners.

The banners with the most unusual distributions are CWMP and BACnet. BACnet is the only protocol which has an average lower volume the more constrained the data set is. CWMP has the most discrete behavior volumes.

In general, while uniquely customized responses are useless for our purposes, behavior volumes seem to indicate that ICMP timestamp behavior testing will not apply everywhere but can help in fingerprinting. After thresholds are applied, a minimum of 20% of primary data banners are consistent in their timestamp behavior. This number grows to 50% in the August 22 data; the real number is likely somewhere between the two.

4.6 Behavior Consistency in Unique Banners

All unique banners also had a behavioral consistency—the percentage of their most common classification out of all classifications. For example, assume 100 IP addresses responded on the FTP port with the same banner, where 40% of them showed "lazy" behavior, 25% showed both "lazy" and "checksumLazy" behavior, and all other classifications had lower



(a) Primary and March 1

(b) Primary and August 22

Figure 4.2. Behavior Consistency for SSH Banners

percentages. The behavioral consistency for that banner would be 40%.

As with behavioral volume, banners were segregated by common protocol/port, and common protocols on different ports were combined. We also compared the primary set (see section 3.1.1) with both March 1 and August 22 results (see section 3.5) to estimate the effect of IP churn. Once again the exception is BACnet, due to the lack of BACnet responses in the August 22 dataset.

We judged that consistency should be less affected by IP churn, as it was based on percentages rather than direct quantities. Therefore, we eliminated only those banners which showed up a single time across all behaviors; this was intended to get rid of just completely unique customized banners. We used only banners over this threshold, rather than comparing banners "over minimum threshold" to all banners.

For graphic results for all protocols, see Appendix F.

Unexpectedly, all August 22 data was *less* consistent than primary data. We anticipated it would have less IP churn and therefore more consistency. However, this is easily explained through the scale of data collected. Any network hiccup in 1.8k data points is much more significant than in 37M data points. Similarly, due to less data points, August 22 data had an absolute minimum consistency higher than the primary data.

March 1 data is nearly identical to primary data in SSH consistency, and arguably similar in HTTP and CWMP. FTP, BACnet, and Telnet are slightly less consistent than primary data.

In general, out of primary data, a minimum of 60% of banners are at least 50% consistent in behavior. This is very encouraging for the use of ICMP timestamps in fingerprinting. Even assuming that there is no loss in consistency due to IP churn, anything that is truly over 50% consistent can be used in fingerprinting by probability. Furthermore, approximately 50% of Telnet respondents are nearly 100% consistent, and other protocols such as FTP are not far behind.

4.7 Behaviors by Manufacturer

To follow up on the original Sundial [2], we analyze the presence of manufacturers using a list derived as described in section 3.2.2. Analyses are separated by broad behavioral groups as listed in Table 4.1, excluding "timezone" and "unknown." Due to many classifications' presence in multiple behavioral groups, classifications may show up twice.

It is important to note that this analysis is based on individual IP addresses, not on unique banners. Furthermore, if more than one manufacturer's name shows up in a single address's responses, both are counted. A device can therefore count for both Microsoft and Asus, for example. Among the primary set, there were 3,349 addresses that showed multiple manufacturers; 13 showed more than two manufacturers. The majority were combinations of Microsoft and Cisco.

Analysis was performed two ways—using the density of manufacturers in a certain behavioral group's behaviors, and using the density of individual behavioral group's behaviors in manufacturers. For the purposes of filtering out IP churn and to make results decipherable, we used minimum thresholds for all groups as described in section 3.5. Thresholds were chosen roughly proportionate to the amount of results for each group as seen in Fig 4.1, then adjusted for better graph legibility. They are listed in Table 4.8.

Broad Behavioral Group	Threshold
stuck	0
reflect	0
lazy	100
msb	30
normal	10
checksumLazy	10
correct	50

Table 4.8. Thresholds for Manufacturer Analysis.

Over the entirety of the primary group, the second most common manufacturer is nearly an order of magnitude less than the most common, and the sixth manufacturer is approximately an order of magnitude lower than the fifth. The five most common manufacturers are listed in Table 4.9.

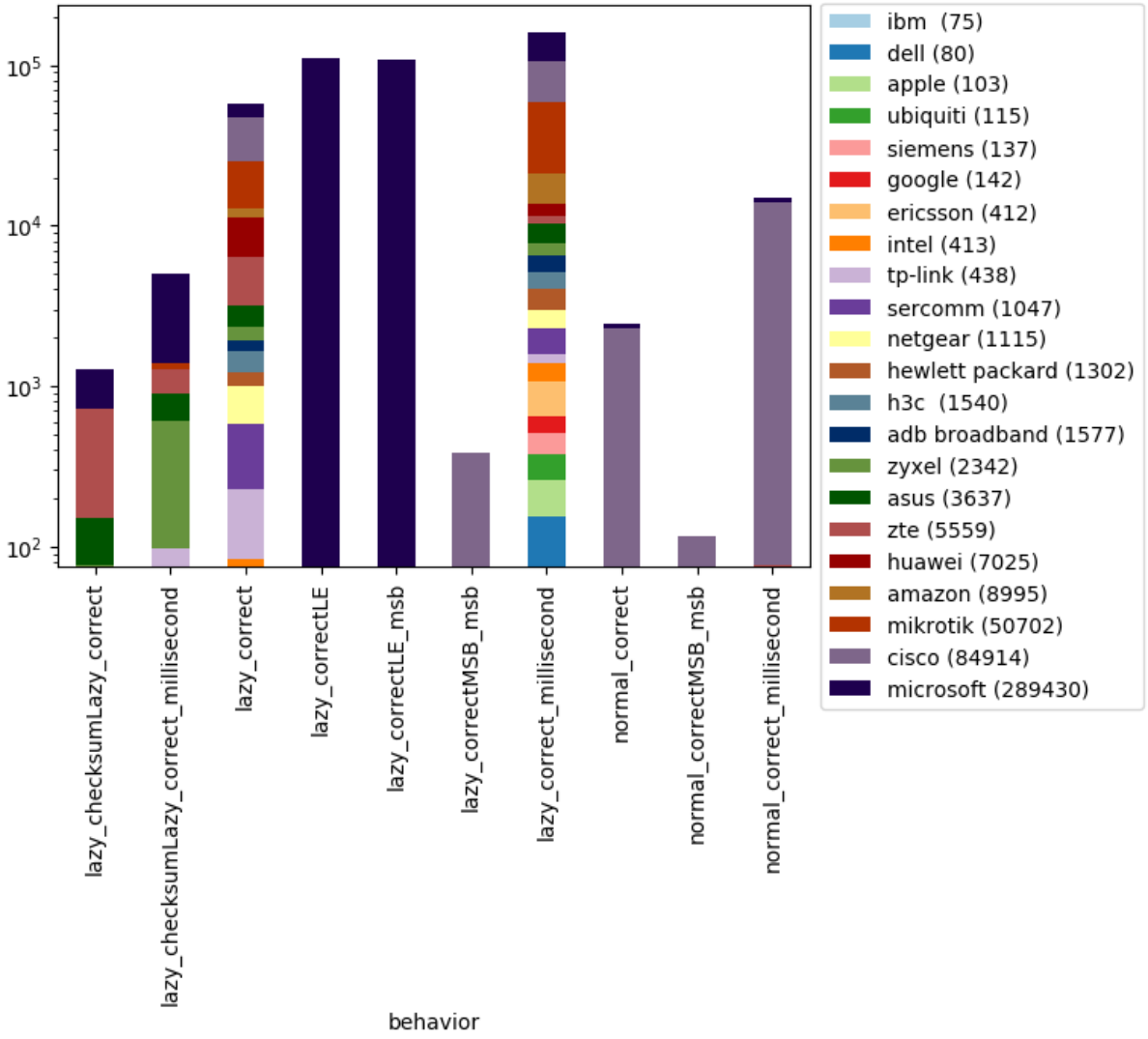
Manufacturer	Instances	% Addresses
Microsoft	1,389,008	0.70%
Cisco	335,009	0.17%
ZTE	159,167	0.08%
MikroTik	149,406	0.08%
Huawei	113,856	0.06%

Table 4.9. Most Common Manufacturers in ICMP Responsive Devices.

In general, even when accounting for complicating factors like IP churn, manufacturers very rarely have a monopoly over some form of behavior. There are exceptions, however. As can be seen in Fig 4.3, Microsoft does seem to be the only manufacturer to display correctLE in a meaningful capacity, and Cisco seems to be the only manufacturer to demonstrate correctMSB. Certain other msb or normal behaviors seem to only appear in Cisco or Microsoft.

Only Microsoft seems to show stuck behavior without stuck0 or stuck1; only Samsung displays stuck0 more than once; only Huawei demonstrates stuck1 or stuckLE1. However,

Figure 4.3. Manufacturers by "Correct" Behavior



with the possible exception of Huawei, we cannot say whether these are meaningful labels given that the sample size is so small. Similarly, Hewlett Packard appears to be the only manufacturer to show normal, reflectRx, and msbTx combined—but this behavior only appears six times total.

Of correct behaviors, nearly all that are normal rather than lazy are Cisco and Microsoft—but it is difficult to say whether this is because of lazy implementations by everyone else, because certain Cisco and Microsoft devices process significantly slower than average, or just due to their prevalence. Interestingly, Microsoft is the only manufacturer to heavily display buggy

behavior. This is strange, as the htons() bug is specific to Linux kernel 3.18 according to [2]. It seems to suggest that there is some subset of Linux 3.18 devices running a Microsoft service, or that a similar bug exists in some Microsoft devices.

Lenovo/Motorola is also unusual, as the only time it shows up with a significant fraction of a behavior is in reflect. However, Lenovo appears in lazy addresses more times total than it does in reflect.

For graphic results for all behavioral groups, see Appendix G.

4.8 Uptime and ITDK-Uptime Behaviors

Manufacturer presence was also analyzed in Uptime (see section 3.1.2) and ITDK-Uptime (see section 3.1.3) datasets, as in section 4.6. A threshold of five was used for both uptime datasets.

The five most common manufacturers in the Uptime dataset were as seen in Table 4.10.

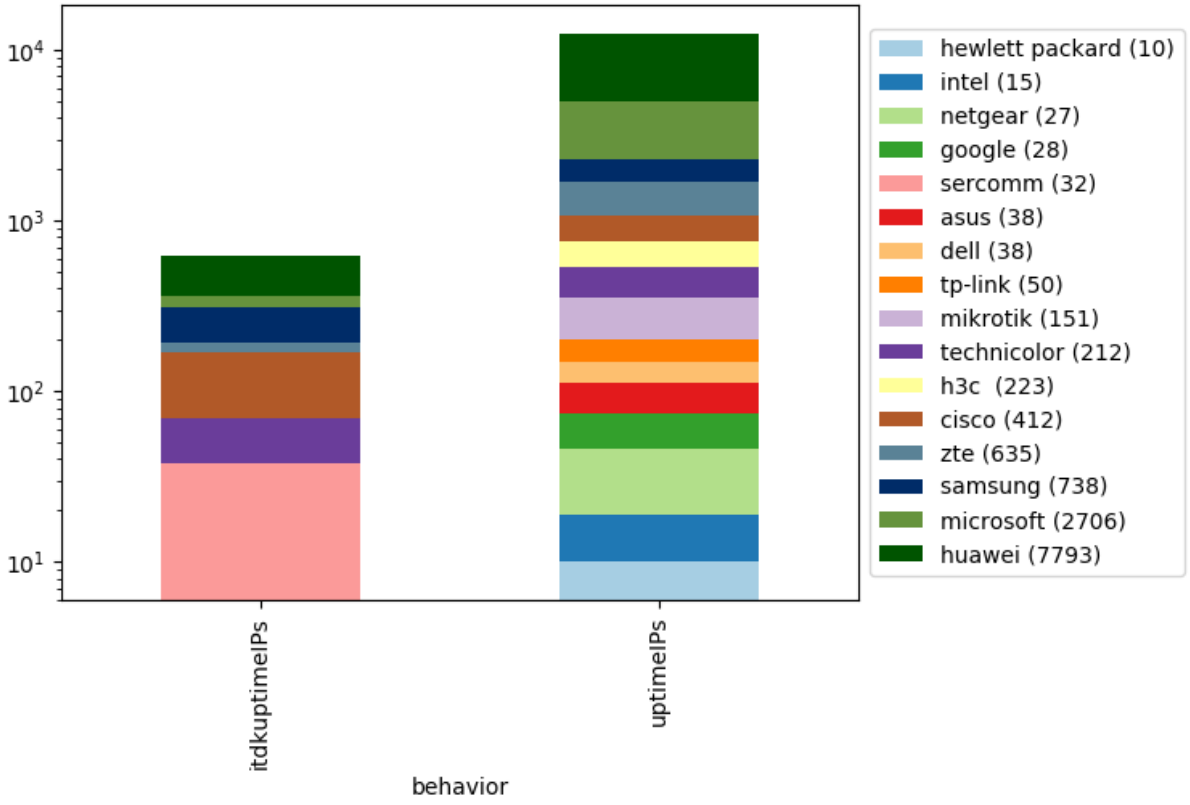
Manufacturer	**Instances**	**% Uptime Addresses**
Huawei	7,524	0.09%
Microsoft	2,656	0.03%
Samsung	621	<0.01%
ZTE	614	<0.01%
Cisco	312	<0.01%

Table 4.10. Most Common Manufacturers in "Uptime" Behavior.

Even the most common manufacturer made up less than 0.1% of all addresses classified as uptime. However, where uptime manufacturers do exist, there is a definite difference compared to overall manufacturer responses. Huawei is overwhelmingly the most likely to occur, and is slightly more likely to occur in uptime devices than all ICMP responsive devices; furthermore, Samsung has replaced MikroTik. (There were only 151 instances of MicroTik among the Uptime dataset.)

Samsung is particularly interesting, as only 2767 instances of Samsung existed in the primary dataset. This means 22.44% of ICMP timestamp responsive Samsungs are classified as

Figure 4.4. Manufacturers by "Uptime" Behavior



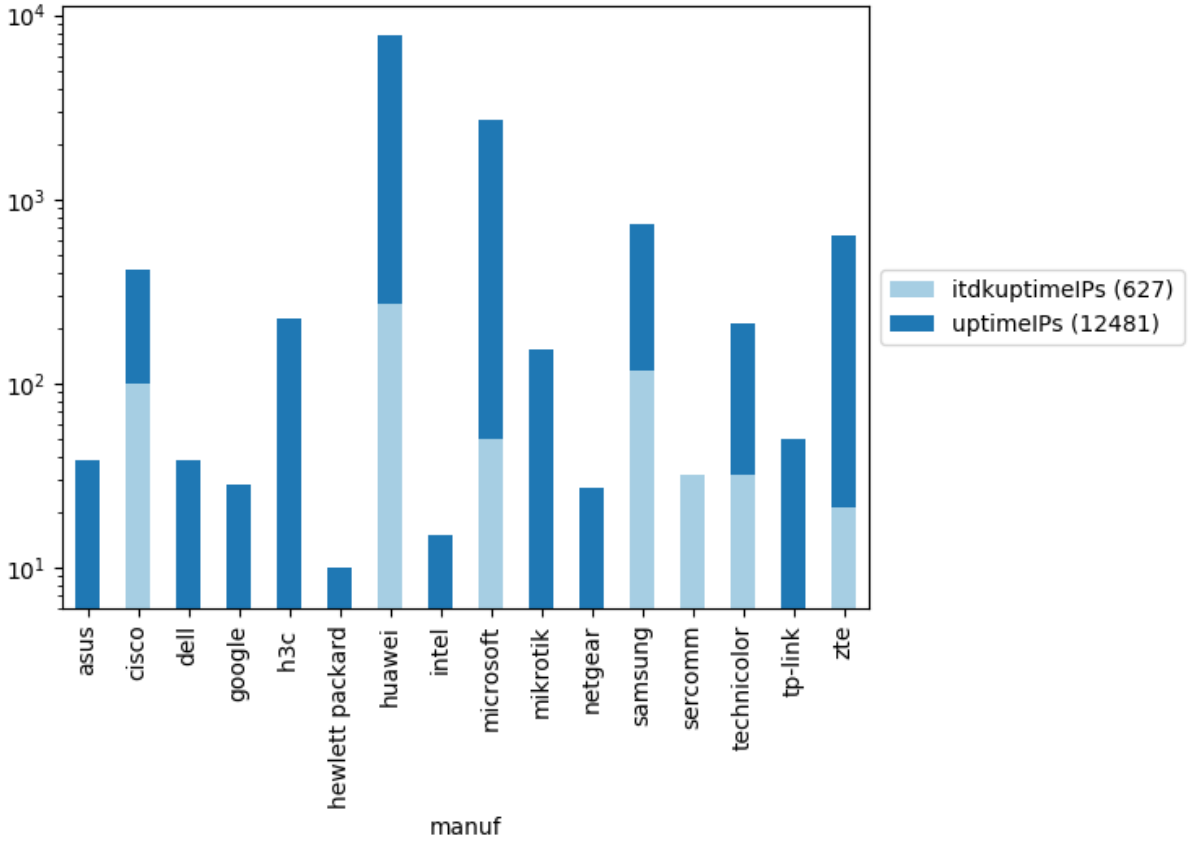
uptime. Of manufacturers with more than 10 instances classified as uptime, only Technicolor is similar, with 180 instances for 31.9% of Technicolor devices.

ITDK-Uptime data was expected to be a subset of all IP addresses classified as uptime, though not all uptime-responding routers were expected to be present in ITDK-Uptime data. (See section 4.3 for initial evidence that this was incorrect.)

As ITDK-Uptime was not a comprehensive selection of all routers in the IPv4 space, manufacturers that appeared in Uptime results and not ITDK-Uptime are inconclusive. In fact, certain manufacturers—such as MikroTik and Netgear—produce nearly exclusively routers, and were not present in the ITDK-Uptime dataset.

However, as can be seen in Fig. 4.4, Sercomm appeared a full 32 times in the ITDK-Uptime group without appearing even once in Uptime. (More accurately, Sercomm appeared three times in the Uptime group, and was then filtered out as IP churn by the threshold.)

Figure 4.5. "Uptime" Behaviors by Manufacturer



4.9 Device and Router Fingerprinting

Certain device models *can* be determined to display only a single behavior. For example, Samsung SWL-3300AP—a wireless access point—only appears in stuck0 addresses. Six Samsung printer models—C140x, C145x, C1860, CLP-680, CLX-4190, and CLX-6260—only ever display lazy and lazy_msb behaviors in the primary set, but also appear in uptime addresses. This means they are showing the 32-bit unsigned integer uptime behavior described in section 4.2.1.

Per-manufacturer behaviors, however, may be more valuable. One goal of this work was to determine whether ICMP timestamps were a useful method of fingerprinting for routers, which are often far less responsive to fingerprinting techniques than end devices. They provide non-routing services less often and drop most traffic directed specifically to them. As routers regularly use ICMP for control and route diagnostics, we hypothesized they

would be more likely to respond to ICMP timestamps.

Even knowing the manufacturer may be incredibly helpful in narrowing down a device's identity. As mentioned in section 4.6, certain behaviors are only associated with one manufacturer, as evidenced by the banners they are associated with. CorrectMSB indicating Cisco and stuck1/stuckLE1 indicating Huawei are probably the most useful of these for router fingerprinting purposes. It is also useful to know that Microsoft and Cisco are functionally the only manufacturers that display correct and normal behavior rather than correct and lazy.

Of course, even this evidence is not conclusive; there may be non-Cisco devices that show correctMSB, but do not respond to any of the services we retrieved banners from, or do not name their manufacturer in their banners. It is also important to note that a timestamp behavior would not be enough to fully fingerprint a device, as even a behavior with a single manufacturer applies to multiple devices.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Conclusions and Future Work

Over the course of this work, we determined that—in the routable IPv4 space—approximately 6% of addresses are responsive to ICMP timestamps. When scanned with four different probes over 16 days, the number that consistently answer shrinks to 5%. This may be an explicit response to scanning, an effect of network congestion, a consequence of devices being powered down, or a result of IP churn.

Of responsive addresses, only about one-third were theoretically conformant to RFC 792 [3]. Responsive addresses were also overwhelmingly classified as lazy. This is not unexpected, as the original Sundial work [2] had similar results. Implementation may not necessarily be lazy—normal hosts may appear lazy if their processing time for a message is less than one millisecond, and Sundial reported that many hosts appeared normal from one vantage point and not another. Because of this, we conclude that lazy and normal behaviors are of minimal use in fingerprinting.

A comparatively small amount of timestamp responsive addresses could be associated with Censys header/banner data. This may be because they are ICMP responsive but do not provide any other services, because they provide services on ports other than those we checked, or because they simply block Censys scanning.

Out of those ICMP responsive addresses that had banners and headers, we determined that there is a correlation between certain behavior classifications and manufacturers, and that there are banner responses that are uniquely associated with one classification. Among banner responses that are not consistent to one behavior, a majority are at a minimum 50% consistent to one behavior. We therefore conclude that—while they are not suitable for fingerprinting by themselves—ICMP timestamp behaviors can be used to assist in fingerprinting. They may be particularly useful in the case of routers, which tend to respond to fewer traditional fingerprinting techniques.

We used samples of smaller datasets with tighter time frames to see if behavioral inconsistency could be blamed on IP churn. While this cut down on maximum behaviors seen, it

did not explain the inconsistency.

The uptime behavior, one not discussed in the original Sundial, makes up just over 4% of responsive addresses. It also encompasses two varieties: one where the entire timestamp field is used as a 32-bit unsigned integer counting uptime, and one where the most significant bit is always set and the remaining 31 bits are used as an unsigned integer. Where possible to associate with a manufacturer, devices with uptime behavior are significantly more likely to be Huawei than overall devices that are ICMP timestamp responsive.

5.1 Primary Takeaways

- 6% of routable IPv4 addresses respond to ICMP timestamp requests.
- 1% of routable IPv4 addresses respond to ICMP timestamp requests *and* to at least one of 10 ports running common services.
- Only about one-third of routable IPv4 addresses respond correctly to ICMP timestamp requests. All other addresses have an incorrect implementation, are behind a middlebox that distorts ICMP responses, have an incorrect time configuration, or respond incorrectly for some other reason.
- A majority of unique banners that devices respond with are at least 50% consistent in one timestamp behavior (and mean the behavior can be used in fingerprinting).
- Some unique banners that devices respond with are 100% consistent in one timestamp behavior (and mean the behavior can be used in fingerprinting).
- Certain timestamp behaviors are only seen in devices from one manufacturer (as implied by banner responses).
  - CorrectLE is only meaningfully seen in Microsoft.
  - CorrectMSB is only meaningfully seen in Cisco.
  - Stuck1/stuckLE1 are only seen in Huawei.
- Certain models of device only show a single behavior (as implied by banner responses).
- Uptime devices are about 4% of ICMP timestamp devices.
  - There are two varieties of uptime behavior, a 32-bit unsigned integer version and a 31-bit unsigned integer version.
  - Uptime devices are far more likely to be Huawei devices.

5.2 Future Work

We recommend periodic probing across the IPv4 space to establish patterns of behavior. This would help determine what percentage of the internet is ICMP timestamp responsive on a regular basis, and what caused the decrease to 5% responsiveness in our scans.

While we found some devices associated with a certain behavior, associating banners and classifications with devices on a large scale was infeasible within the scope of this work. Future work which wished to pursue fingerprinting would have to create more sophisticated noise reduction techniques, as well as do manual research to associate banners to devices. Similarly, more devices should be directly obtained and probed in order to establish ground truths, particularly as less than 15% of responsive addresses had useful banner responses. Finally, the two uptime varieties should be investigated as separate behaviors.

We did not investigate the full scope of potentially useful services/ports; future work may wish to investigate additional well-known ports. It may also be prudent to scan banners along with Sundial's ICMP probes, rather than relying on Censys, to minimize IP churn even further.

In the long term, ICMP timestamp fingerprinting should be integrated into network scanning tools such as Nmap [1].

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A:
Case Studies of Banner Noisiness

Banner noisiness made extracting models and analyzing behaviors by banner difficult. Many banners were completely unique but clearly custom-set, and no longer offered any illuminating information for device identification. For example, some banners from FTP and SSH replies:

```
220 This site is restricted for use of Cellebrite Training Staff
    only.

220 Hm...

Look at you, hacker. A pathetic creature of meat and bone, panting
    and sweating as you run through my corridors. How can you
    challenge a perfect, immortal, machine?

NOPE2.0-NOPENOPE
```

These above unique banners were a functionally inconsequential, tiny percentage of the overall banner collection, and were truly unique. More disruptive were banners that were otherwise entirely identical but that displayed their own IP address, the date, or a number of users. Examples of the following included:

```
PacketShaper (193.238.28.126)   Password:

PacketShaper (194.224.251.100)   Password:

QTerm(v1.0.4)  Tuesday, 26 February 2019 08:48:32  ZMM100 login:

QTerm(v1.0.4)  Tuesday, 26 February 2019 09:02:52  ZMM100 login:

QTerm(v1.0.4)  Tuesday, 26 February 2019 09:08:13  ZMM100 login:

220---------- Welcome to Pure-FTPd [privsep] [TLS] ---------- 220-
    You are user number 7 of 50 allowed. 220-Local time is now 16:12.
```

```
 Server port: 23. 220-This is a private system - No anonymous
login 220-IPv6 connections are also welcome on this server. 220
You will be disconnected after 15 minutes of inactivity.

220---------- Welcome to Pure-FTPd [privsep] [TLS] ---------- 220-
You are user number 8 of 50 allowed. 220-Local time is now 01:36.
 Server port: 23. 220-This is a private system - No anonymous
login 220-IPv6 connections are also welcome on this server. 220
You will be disconnected after 15 minutes of inactivity.
```

There were also those that inserted a unique substring (seemingly a username) into each otherwise identical banner. It is very likely not all of these were filtered out, but one that was handled was Netrouter 2G, such as the following:

```
Linux 2.4.30-pre1-p1_01 (DILUMAR) (2)
    ------------------------------------ Welcome to NetRouter 2G from
     Digitel ------------------------------------ DILUMAR login:

Linux 2.4.30-pre1-p1_01 (ECONSULTING) (1)
    ------------------------------------ Welcome to NetRouter 2G from
     Digitel ------------------------------------ ECONSULTING login:

Linux 2.4.30-pre1-p1_01 (FADIPA) (7)
    ------------------------------------ Welcome to NetRouter 2G from
     Digitel ------------------------------------ FADIPA login:
```

The biggest obstacle, however, was the inconsistent presentation of manufacturers and models. As manufacturers did not deign to always have their name in a standardized place in every banner, extraction of manufacturers had to be done off a pre-built list. Furthermore, manufacturers were inconsistent in their own spelling and in whether or not they used contractions of their name. Consider the following examples with Hewlett-Packard:

```
HP Printer  Enter username:

*********************************************************************
    Copyright (c) 2010-2017 Hewlett Packard Enterprise Development LP
         * * Without the owner's prior written consent,
    * * no decompiling or reverse-engineering shall be allowed.
```

```
                        *
    ******************************************************************
      Password:

2J?7l3;23r?6l1;1H?25l1;1HHP J4813A ProCurve Switch 2524 Firmware
    revision F.05.17  Copyright (C) 1991-2003 Hewlett-Packard Co.
    All Rights Reserved.                              RESTRICTED
    RIGHTS LEGEND   Use, duplication, or disclosure by the Government
     is subject to restrictions  as set forth in subdivision (b) (3)
    (ii) of the Rights in Technical Data and  Computer Software
    clause at 52.227-7013.           HEWLETT-PACKARD COMPANY, 3000
    Hanover St., Palo Alto, CA 94303  1;24r20;1H24;1HPress any key to
     continue20;1H?25h
```

As may clearly be seen, slightly different punctuation was used; this may have been a product of minor re-brandings, but did make distinguishing and sorting manufacturers more difficult. Manufacturers with multi-word names often had to be cut down to just the first word in order to be reliably found.

Manufacturers also had no consistent way of reporting model names relative to their name. Even if all model names had immediately followed manufacturer names, they would have been difficult to extract (consider "Hewlett-Packard Co" versus "Hewlett Packard Enterprise Development LP" above). However, as the following sampler from Telnet shows, model names could be nearly anywhere in the banner:

```
Welcome to DASAN Zhone Solutions                Model: ZNID-GE-2426
    A1-UK Router                Release: S4.1.048  Copyright (C)
    2009-2017 by DASAN Zhone Solutions, Inc.  All Rights Reserved.
    Confidential, Unpublished Property of DASAN Zhone Solutions.
    Rights Reserved Under the Copyright Laws of the United States.
    Login:

"Linux 2.4.30-pre1-p1_01 (XXXXX)
    ---------------------------------- Welcome to NetRouter 2G from
     Digitel ---------------------------------- XXXXX login:"

*********************************************          Welcome
    to ZXR10 5928PS Switch of ZTE Corporation
```

```
    ********************************************  Username:

BLM1500 - Broadband Loop Multiplexer Copyright 2001-2007 Entrisphere
     Inc. Fabric & Control Processor (ttyp0)

WAVECAST(TM) MMC Technology, Inc.(http://www.mmctech.com) Wireless
    Accesspoint MW-1700AP (3.5.2) GSM-4F-406-MMC-1 login:
```

This meant that models could not be extracted without an existing list of models or to be entirely sorted through by hand, something which was outside of the scope of this study's ability to do.

There were many banners which ran on non-standard ports - for example, we found instances of SSH running on the standard Telnet port, FTP running on the standard SSH and Telnet ports, and HTTP running on the SSH port. This list is non-comprehensive and covers only those banners which used non-standard ports often enough to be easily spotted. There were likely other combinations.

CWMP responses had one particularly odd behavior. While they tended to have nonces unless they had only a "basic" realm, which we treated as noise, not all nonces were completely unique. For example, the following CWMP responses recurred a full 1702 and 1694 times, respectively:

```
"Digest realm=""IgdAuthentication"", domain=""/"", nonce=""
    IGEwNDNlMTA6NTMzOGRlNjA6OWY4ODQzMzA="", qop=""auth"", algorithm=
    MD5"

"Digest realm=""IgdAuthentication"", domain=""/"", nonce=""
    YjY3Yjg5NDA6ODYzNzFkOTA6MzIxMTBlMzA="", qop=""auth"", algorithm=
    MD5"
```

For researcher illumination, the following partial banner is offered. This banner was not unique aside from the server name in the final line:

```
------------------------------------------------------
ATTITUDE ADJUSTMENT (Attitude Adjustment, r1327)
------------------------------------------------------
* 1/4 oz Vodka       Pour all ingredients into mixing
* 1/4 oz Gin         tin with ice, strain into glass.
* 1/4 oz Amaretto
* 1/4 oz Triple sec
* 1/4 oz Peach schnapps
* 1/4 oz Sour mix
* 1 splash Cranberry juice
------------------------------------------------------
mimosa_20B5C60347D8 login:
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B:
Manufacturer List

The table following lists the final 55 manufacturers used for searching within banners, including any variations of a single manufacturer used. Name variations shorter than 4 letters generally required at least one trailing whitespace to be counted as an instance.

Table B.1. Manufacturer List Extracted From OUI.

3com	huawei
actiontec	ibm
adb broadband	innotek
aerohive	intel
alcatel-lucent	juniper networks
amazon	lenovo, motorola
ampak	lg electronics
apple	microsoft
arcadyan	mikrotik
arris group	netgear
askey	nokia
asus	nortel network
avm	rim
belkin	samsung
buffalo	seiko epson
canon	sercomm
china mobile	sharp corporation
ciena	siemens
cisco, cisco-linksys	sony
d-link	technicolora
dell	toshiba
ericsson	tp-link
extreme networks	ubiquiti
fiberhome	xerox
google	zhejiang dahua
h3c	zte
hewlett packard, hp, hewlett-packard	zyxel
htc	

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C:
Noise Reduction Expressions

The following regular expression was used for trimming standard IP addresses and ports.

```
\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}(:\d{2,5})?
```

The following regular expression was used for trimming standard timestamps.

```
\d{1,2}\:\d{1,2}\:\d{1,2}
```

The following three regular expressions were used for trimming dates and weekdays.

```
\d{1,4})? (?i)(January|February|March|April|May|June|July|August|
    September|October|November|December|Jan|Feb|Mar|Apr|Jun|Jul|Aug|
    Sep|Oct|Nov|Dec) (\d{1,4})?([,\s]{1,2}\d{4})?

\d{1,4}[\/-]\d{1,2}[\/-]\d{1,4}

\s(?i)(Monday|Tuesday|Wednesday|Thursday|Friday|Mon|Tue|Tues|Wed|Thu
    |Thurs|Fri|Sat|Sun)[,\.\s]
```

Additionally, the following substrings were used in some individual banner types to detect and replace specific noisy strings. Note leading whitespace in some substrings.

```
Last login time is

Ubiquoss L2 Switch

Ubiquoss L3 Switch

Ubiquoss VDSL

Ubiquoss Switch
```

```
Ubiquoss QoS Switch

PlusID

Sorry, Dear! telnet service is still in lock-time, You have to wait

Welcome to NetRouter 2G from Digitel

 You are user

220-You are user number

Local time

Welcome to

 Server (

 FTP server (

 FTP server ready.

nonce

domain
```

APPENDIX D:
Sundial Classification Percentages

The table following lists the entirety of ICMP responsive addresses, enumerated by Sundial classification [2]. This data is from the primary set described in 3.1.1 only. It excludes the "uptime" classification, which would be scattered through multiple classification groups.

Certain classifications may have contradictory behaviors, such as the four instances of "reflect_timezone." In these cases, the classifier was unable to parse which classification was correct - e.g., the scanner happened to compute hashes that in timestamp form were timezone-offset from the current time. It is impossible to determine if the respondent reflected the receive and transmit timestamps or genuinely reported a non-UTC time.

Other classifications may be redundant. For example, any classification with "correctMSB" must by necessity also include "msb" - correct-msb requires the most significant bit to be set. (However, an address that is classified msb is not necessarily correct-msb.)

Table D.1. Full Sundial Behavioral Groups Found in IPv4 Scan.

Classification	Total Addresses	% of Overall Addresses
lazy	103148960	54.5557127%
lazy_correct_millisecond	41578431	21.9909239%
lazy_correct	12790905	6.7651379%
lazy_millisecond	12391283	6.5537769%
lazy_msb	8047270	4.2562189%
lazy_checksumLazy	2644085	1.3984624%
lazy_timezone	1300000	0.6875729%
lazy_checksumLazy_correct_millisecond	1062315	0.5618607%
lazy_timezone_millisecond	1014377	0.5365062%
normal_correct_millisecond	677225	0.3581858%
lazy_checksumLazy_millisecond	673931	0.3564436%
normal	628536	0.3324341%
normal_msb	584627	0.3092105%
lazy_correctLE	564773	0.2987097%
lazy_correctLE_msb	563123	0.2978370%
lazy_checksumLazy_correct	328455	0.1737206%
lazy_buggy	276715	0.1463552%

continued on next page

Table D.1 – continued from previous page

Classification	Total Addresses	% of Overall Addresses
lazy_msb_buggy	121976	0.0645134%
normal_correct	111747	0.0591032%
normal_msbRx	84979	0.0449456%
unknown	78954	0.0417589%
lazy_checksumLazy_msb	66986	0.0354290%
stuck_stuck0	59001	0.0312058%
normal_correctTx_millisecondTx	47617	0.0251847%
reflect	40376	0.0213550%
lazy_checksumLazy_timezone	35388	0.0187168%
normal_millisecond	34464	0.0182281%
lazy_checksumLazy_timezone_millisecond	23970	0.0126778%
normal_msbTx	18140	0.0095943%
lazy_correctMSB_msb	17201	0.0090976%
normal_correctTx	8247	0.0043619%
msbRx	7607	0.0040234%
normal_millisecondTx	7182	0.0037986%
stuck_stuck1	5240	0.0027714%
normal_correctRx_millisecond	5158	0.0027281%
normal_correctMSB_msb	4298	0.0022732%
normal_correct_millisecondTx	4036	0.0021346%
normal_correctTx_millisecond	1265	0.0006691%
lazy_stuck_msb	1265	0.0006691%
lazy_checksumLazy_buggy	1062	0.0005617%
normal_correctTx_msbRx_millisecondTx	978	0.0005173%
lazy_checksumLazy_msb_buggy	861	0.0004554%
normal_correctRx	806	0.0004263%
lazy_correct_correctLE_millisecond	653	0.0003454%
stuck_stuckLE1	550	0.0002909%
lazy_checksumLazy_correctLE	499	0.0002639%
lazy_checksumLazy_correctLE_msb	475	0.0002512%
normal_correct_millisecondRx	454	0.0002401%
lazy_stuck_msb_buggy	453	0.0002396%
normal_timezone	432	0.0002285%
normal_checksumLazy	377	0.0001994%
msb	327	0.0001730%
correct	290	0.0001534%
normal_correctRx_millisecondTx	281	0.0001486%

continued on next page

Table D.1 – continued from previous page

Classification	Total Addresses	% of Overall Addresses
normal_checksumLazy_msb	242	0.0001280%
normal_msbTx_millisecondRx	212	0.0001121%
lazy_correct_correctLE	193	0.0001021%
normal_millisecondRx	189	0.0001000%
lazy_msb_second	180	0.0000952%
lazy_stuck	173	0.0000915%
checksumLazy	131	0.0000693%
normal_checksumLazy_msbRx	107	0.0000566%
lazy_second	96	0.0000508%
normal_correctTx_msbRx	88	0.0000465%
normal_correctRx_millisecondRx	85	0.0000450%
normal_checksumLazy_correct_millisecond	62	0.0000328%
correct_millisecond	43	0.0000227%
millisecond	39	0.0000206%
checksumLazy_reflect	38	0.0000201%
normal_timezone_millisecond	34	0.0000180%
normal_correctTx_millisecondRx	25	0.0000132%
normal_correctLE_msb	25	0.0000132%
timezone	21	0.0000111%
normal_correctLE	20	0.0000106%
normal_correctRx_correctMSBTx_msbTx	18	0.0000095%
lazy_correct_buggy	18	0.0000095%
normal_correctRx_correctMSBTx_msbTx_millisecondRx	17	0.0000090%
lazy_correctLE_correctMSB_msb	17	0.0000090%
lazy_checksumLazy_correct_correctLE_millisecond	17	0.0000090%
normal_correctMSBTx_msb	14	0.0000074%
normal_checksumLazy_correct	14	0.0000074%
normal_reflectRx_msbTx	11	0.0000058%
normal_correct_correctLERx_millisecond	10	0.0000053%
lazy_correctLE_buggy	10	0.0000053%
normal_timezoneTx_msbRx	8	0.0000042%
lazy_correctLE_msb_buggy	8	0.0000042%
normal_correct_correctLETx_millisecond	7	0.0000037%
lazy_checksumLazy_correct_correctLE	7	0.0000037%
stuck_stuck0_timezone	6	0.0000032%
normal_timezoneTx	6	0.0000032%
normal_correct_correctLE_millisecond	6	0.0000032%

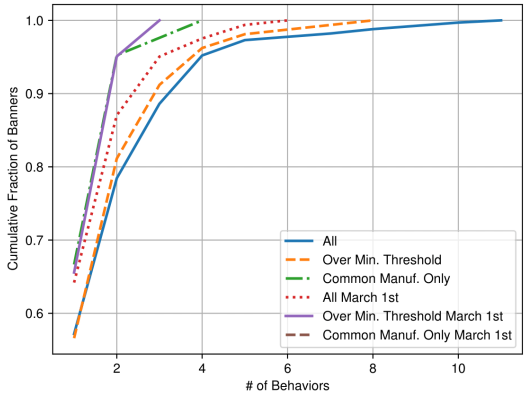
continued on next page

Table D.1 – continued from previous page

Classification	Total Addresses	% of Overall Addresses
normal_checksumLazy_stuckRx_msb	5	0.0000026%
reflect_timezone	4	0.0000021%
lazy_msb_millisecond	4	0.0000021%
normal_timezoneRx	3	0.0000016%
normal_msbRx_millisecondTx	3	0.0000016%
lazy_checksumLazy_stuck	3	0.0000016%
checksumLazy_msb	3	0.0000016%
checksumLazy_correct	3	0.0000016%
timezone_millisecond	2	0.0000011%
stuck_stuck0_correct_correctLE	2	0.0000011%
reflect_second	2	0.0000011%
normal_timezoneRx_msbTx	2	0.0000011%
normal_secondTx	2	0.0000011%
normal_correct_correctLERx	2	0.0000011%
normal_correctTx_correctLETx_millisecondTx	2	0.0000011%
normal_checksumLazy_msbTx	2	0.0000011%
normal_checksumLazy_millisecond	2	0.0000011%
lazy_stuck_buggy	2	0.0000011%
lazy_correct_second	2	0.0000011%
normal_stuckRx_msb	1	0.0000005%
normal_stuckRx_correctTx_msbRx_millisecondTx	1	0.0000005%
normal_reflectRx	1	0.0000005%
normal_reflect	1	0.0000005%
normal_correct_correctLE	1	0.0000005%
normal_correctMSBRx_msb	1	0.0000005%
lazy_timezone_second	1	0.0000005%
lazy_timezone_buggy	1	0.0000005%
lazy_checksumLazy_stuck_msb	1	0.0000005%
lazy_checksumLazy_msb_millisecond	1	0.0000005%
lazy_checksumLazy_correctMSB_msb	1	0.0000005%
correctRx	1	0.0000005%
correctLE_msb	1	0.0000005%
correctLE	1	0.0000005%
checksumLazy_timezone	1	0.0000005%
checksumLazy_stuck_stuck0	1	0.0000005%

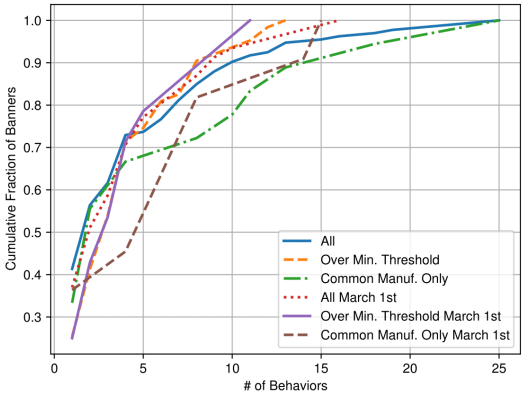
APPENDIX E:
Behavior Volume Graphs

See section 4.4 for thresholds used.

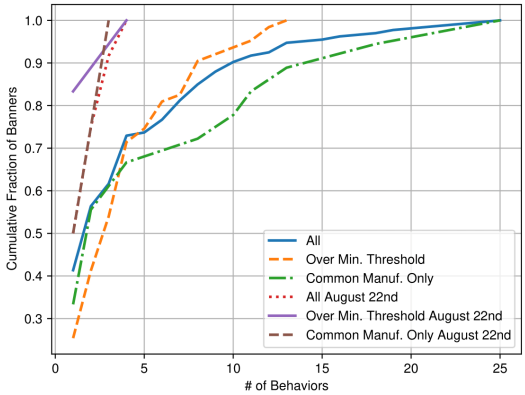


(a) Primary and March 1

Figure E.1. Behavior Volume for BACnet Banners

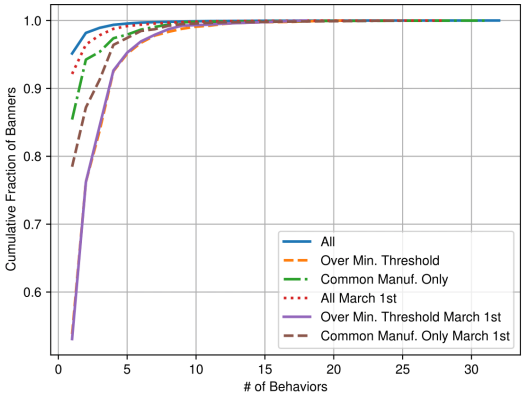


(a) Primary and March 1

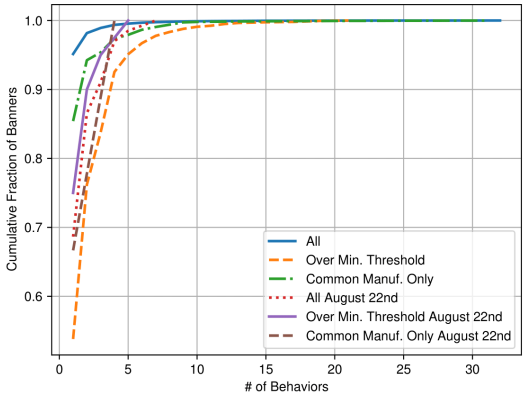


(b) Primary and August 22

Figure E.2. Behavior Volume for CWMP Banners

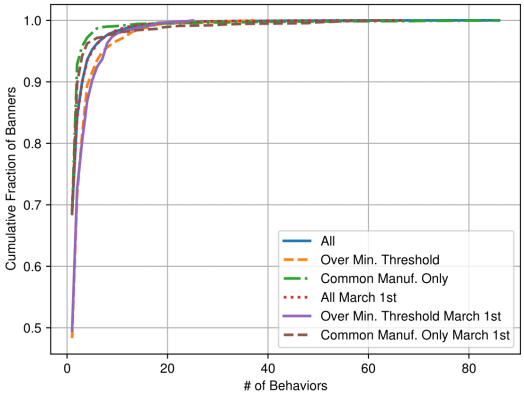


(a) Primary and March 1

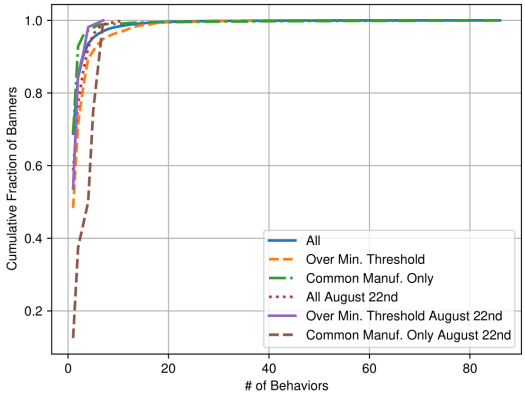


(b) Primary and August 22

Figure E.3. Behavior Volume for FTP Banners (Duplicate)

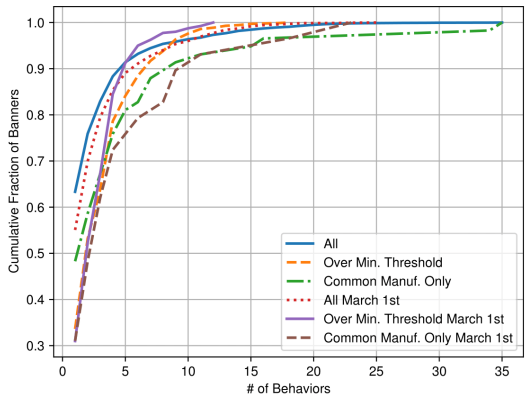


(a) Primary and March 1

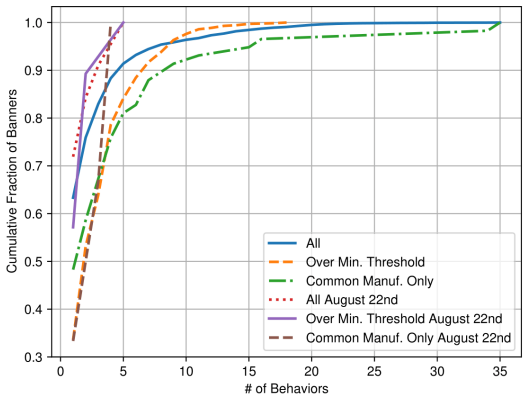


(b) Primary and August 22

Figure E.4. Behavior Volume for HTTP Banners

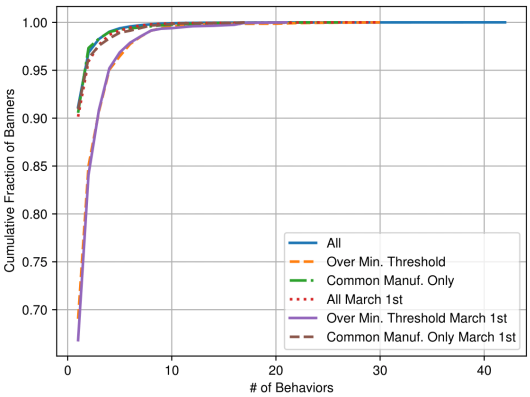


(a) Primary and March 1

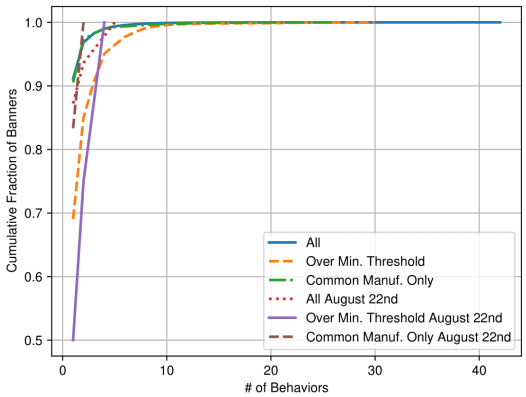


(b) Primary and August 22

Figure E.5. Behavior Volume for SSH Banners



(a) Primary and March 1



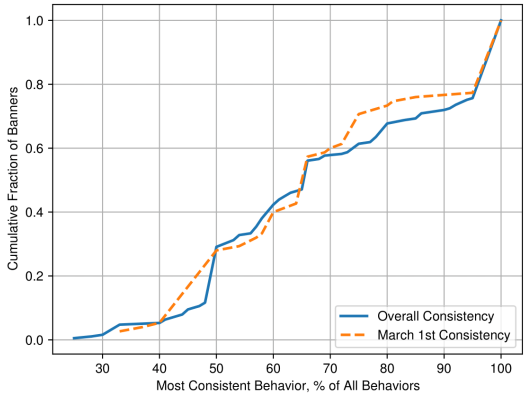
(b) Primary and August 22

Figure E.6. Behavior Volume for Telnet Banners

THIS PAGE INTENTIONALLY LEFT BLANK

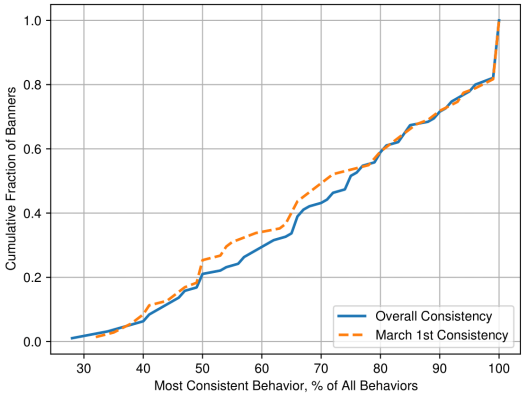
APPENDIX F:
Behavior Consistency Graphs

See section 4.5 for thresholds used.

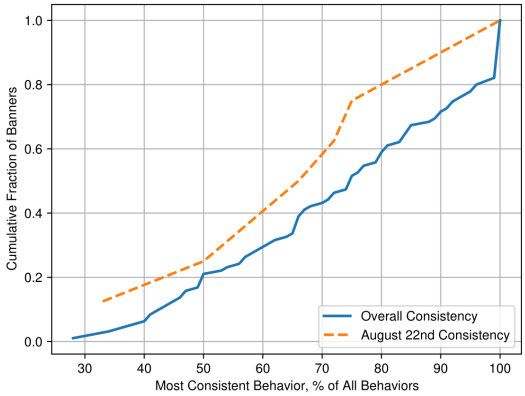


(a) Primary and March 1

Figure F.1. Behavior Consistency for BACnet Banners

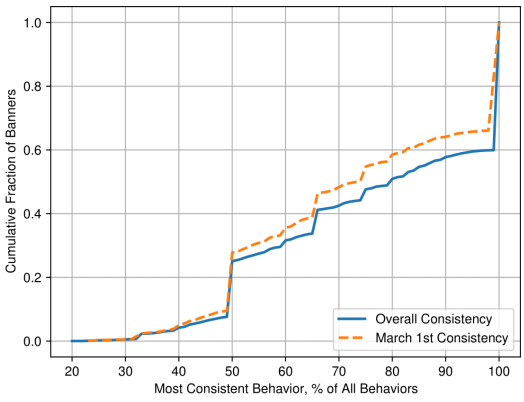


(a) Primary and March 1

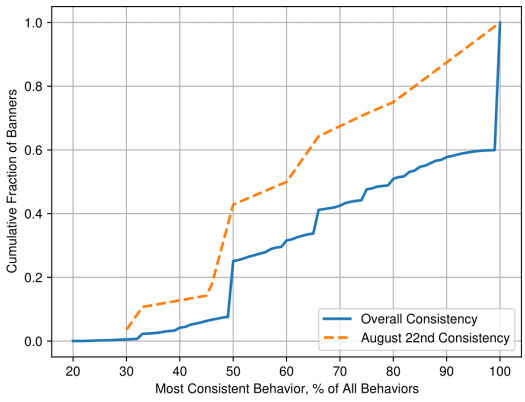


(b) Primary and August 22

Figure F.2. Behavior Consistency for CWMP Banners

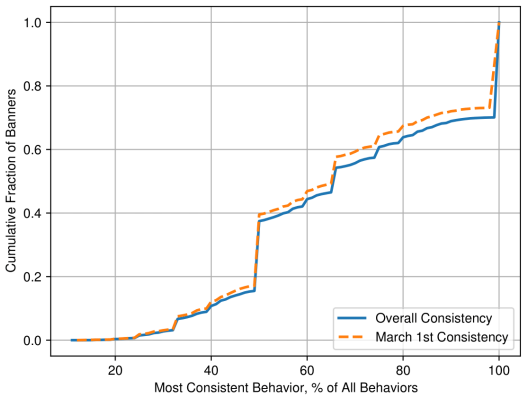


(a) Primary and March 1

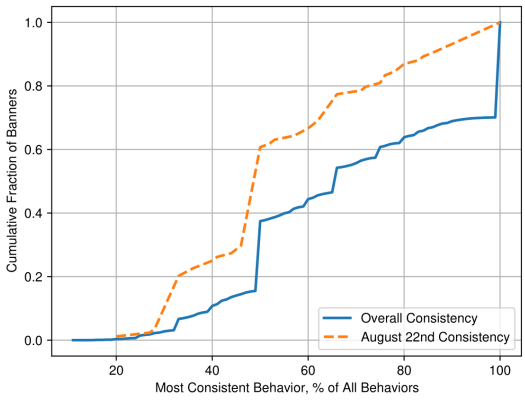


(b) Primary and August 22

Figure F.3. Behavior Consistency for FTP Banners

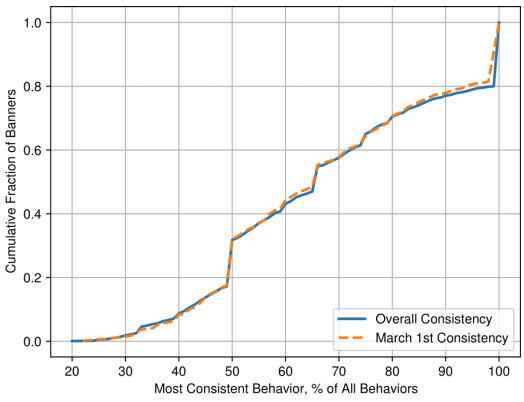


(a) Primary and March 1

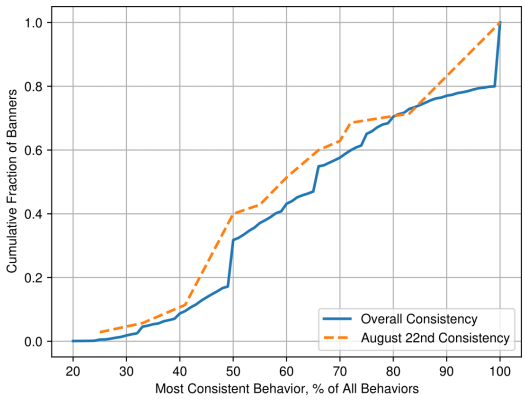


(b) Primary and August 22

Figure F.4. Behavior Consistency for HTTP Banners

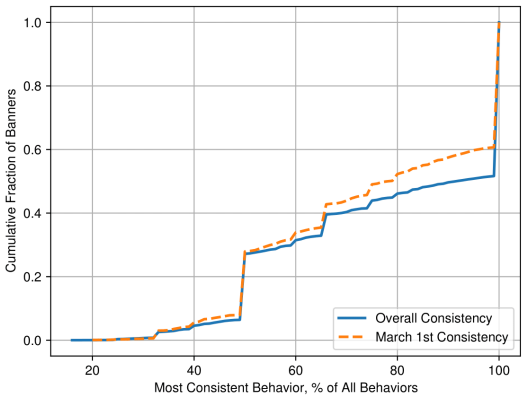


(a) Primary and March 1

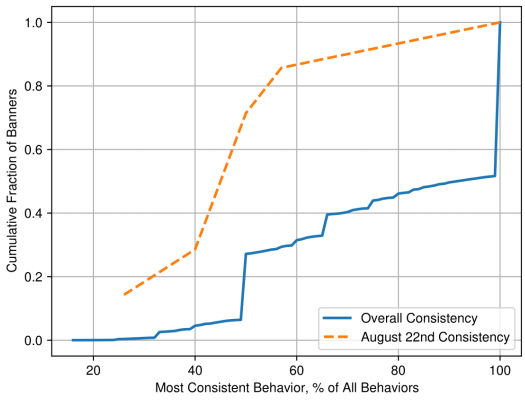


(b) Primary and August 22

Figure F.5. Behavior Consistency for SSH Banners (Duplicate)



(a) Primary and March 1



(b) Primary and August 22

Figure F.6. Behavior Consistency for Telnet Banners

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX G:
Behaviors by Manufacturer

See section 4.6 for thresholds used.

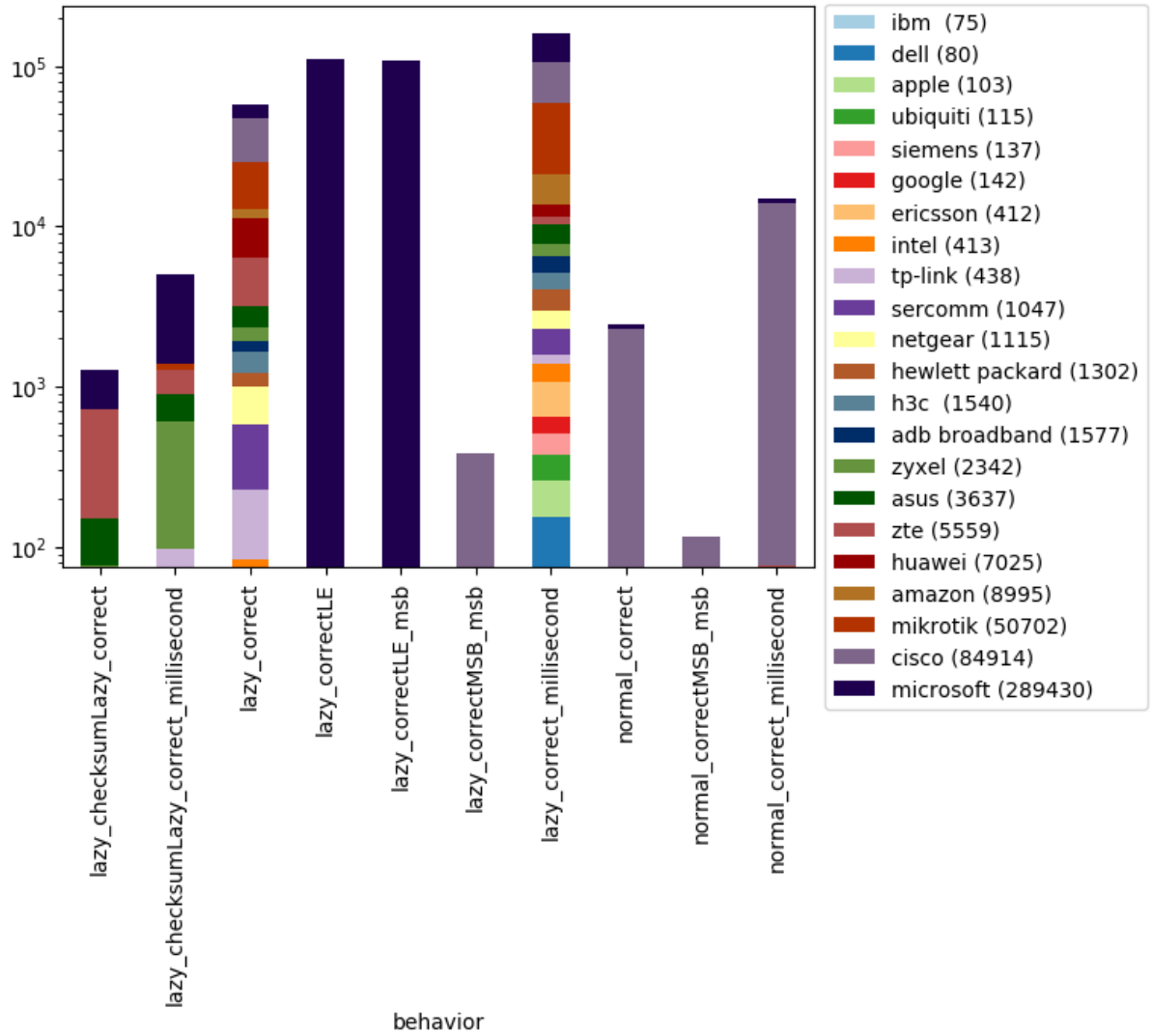


Figure G.1. Manufacturers by "Correct" Behavior (Duplicate)

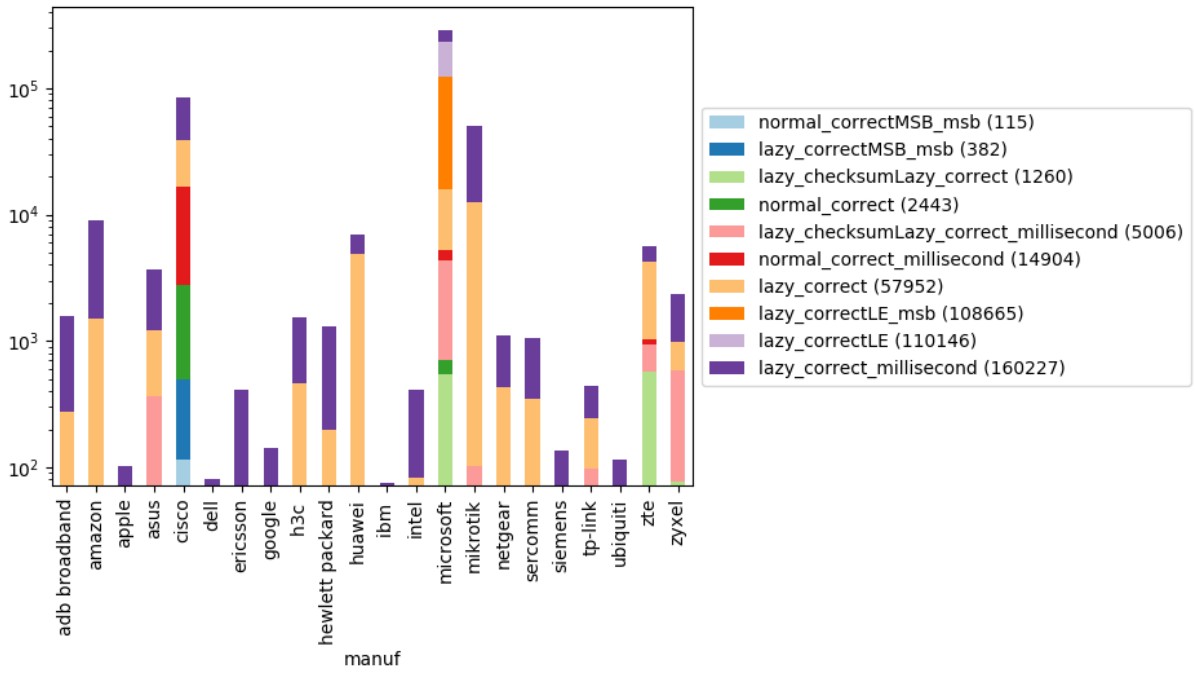


Figure G.2. "Correct" Behaviors by Manufacturer

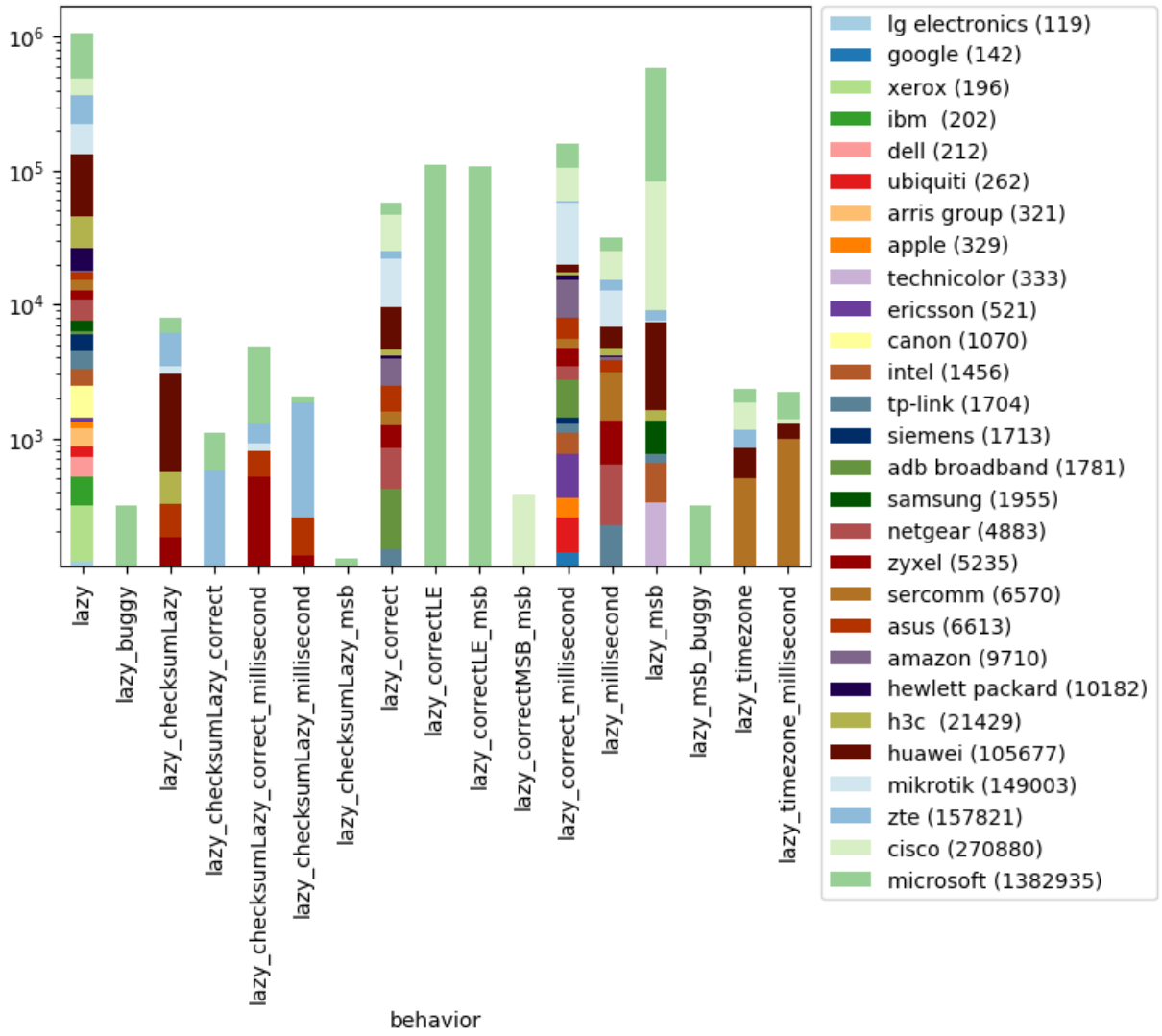


Figure G.3. Manufacturers by "Lazy" Behavior

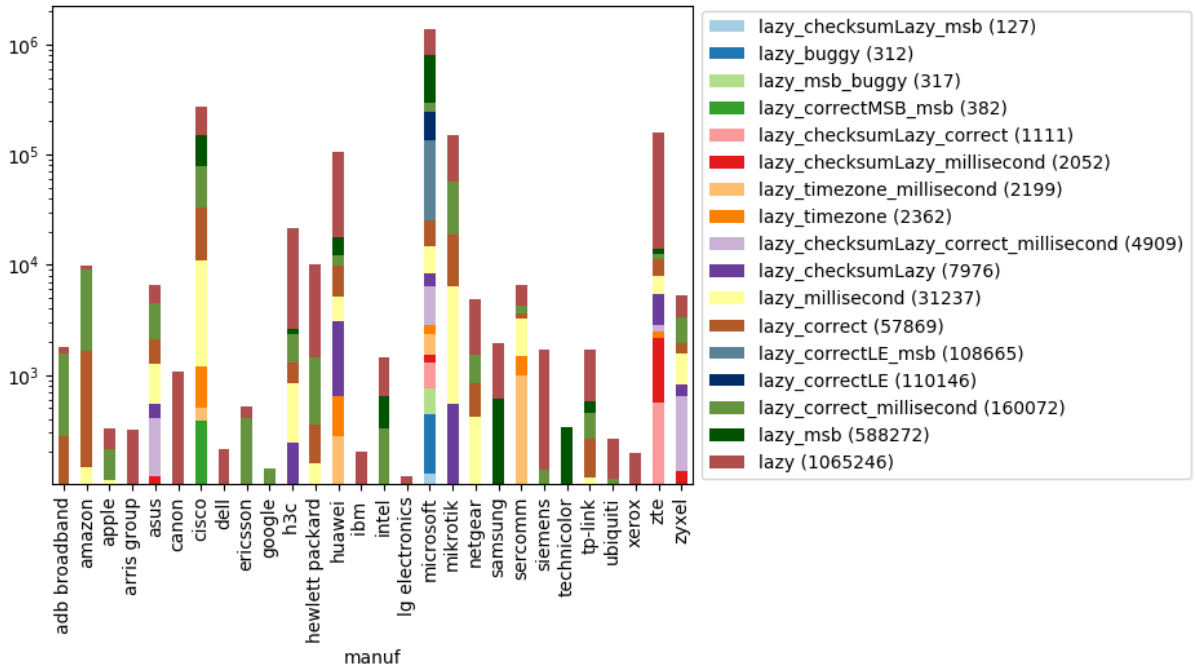


Figure G.4. "Lazy" Behaviors by Manufacturer

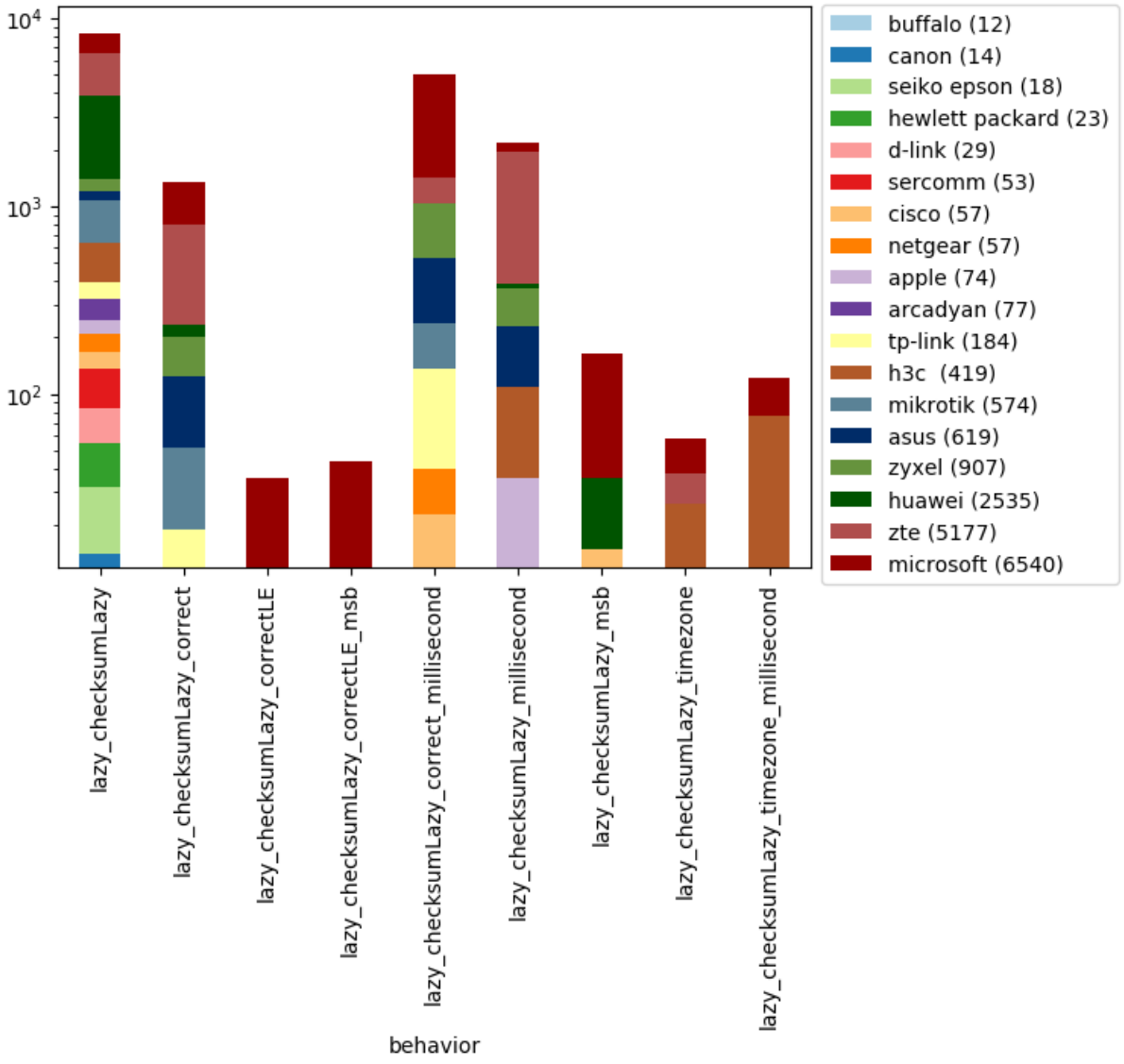


Figure G.5. Manufacturers by "ChecksumLazy" Behavior

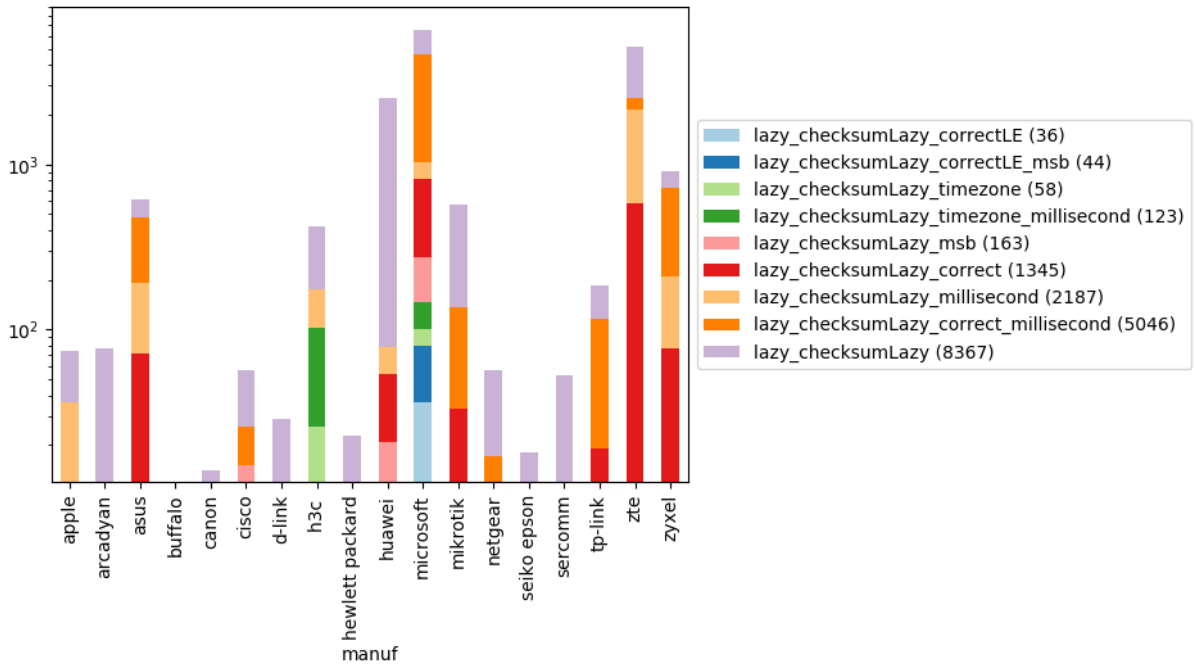


Figure G.6. "ChecksumLazy" Behaviors by Manufacturer

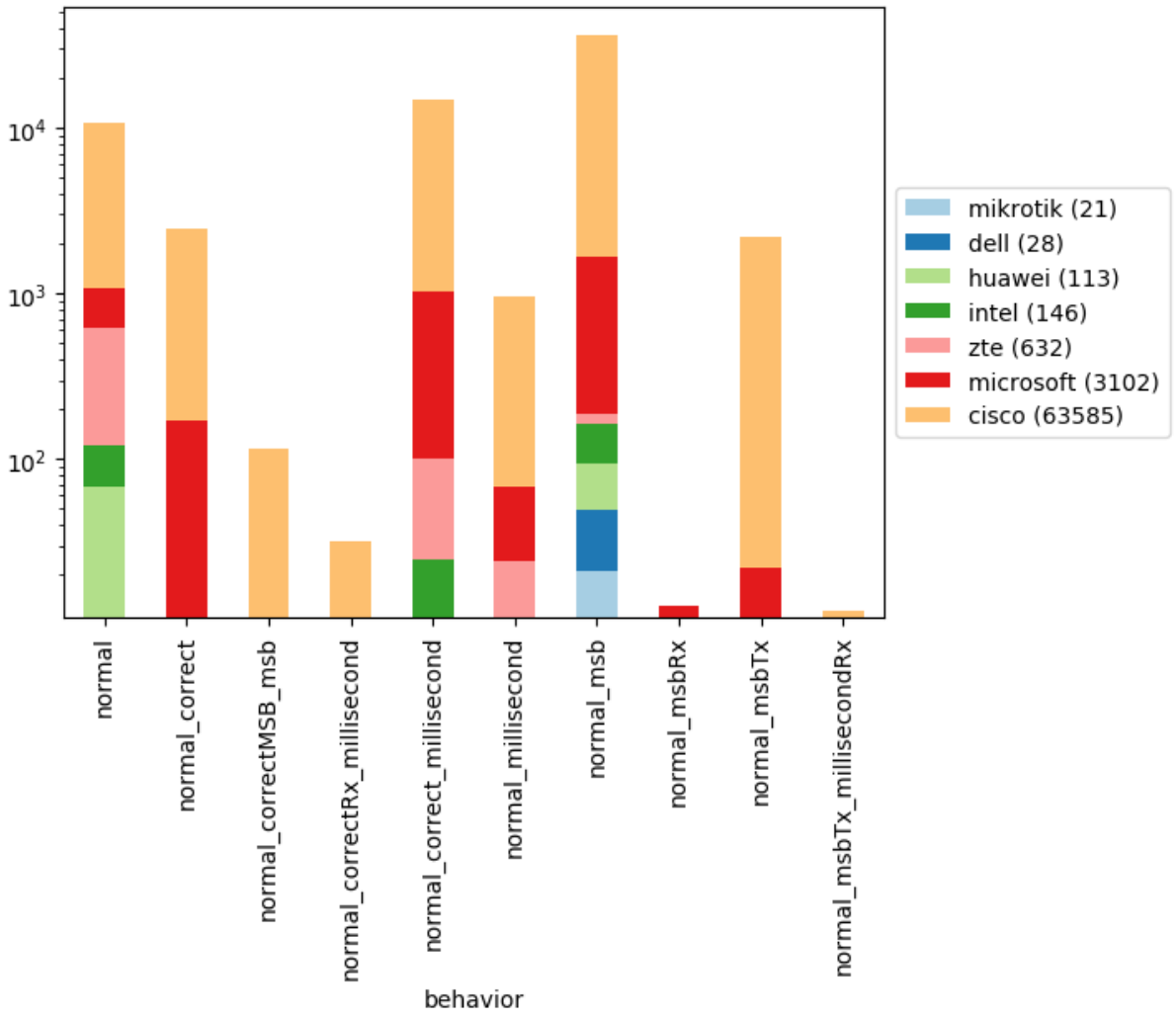


Figure G.7. Manufacturers by "Normal" Behavior

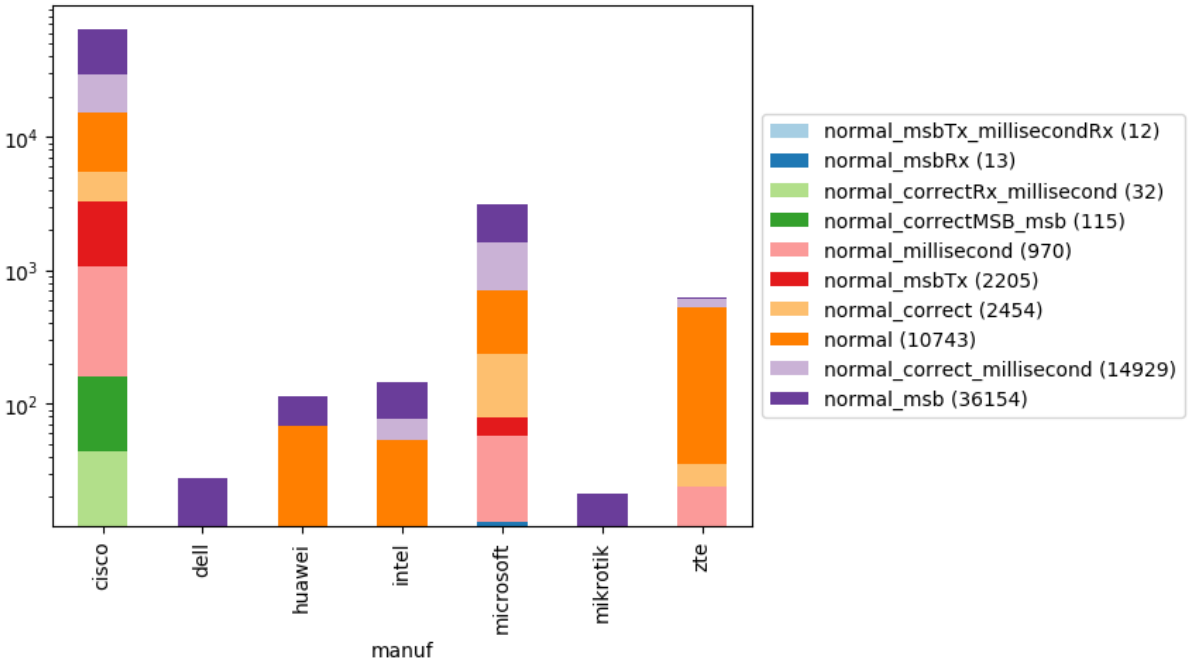


Figure G.8. "Normal" Behaviors by Manufacturer

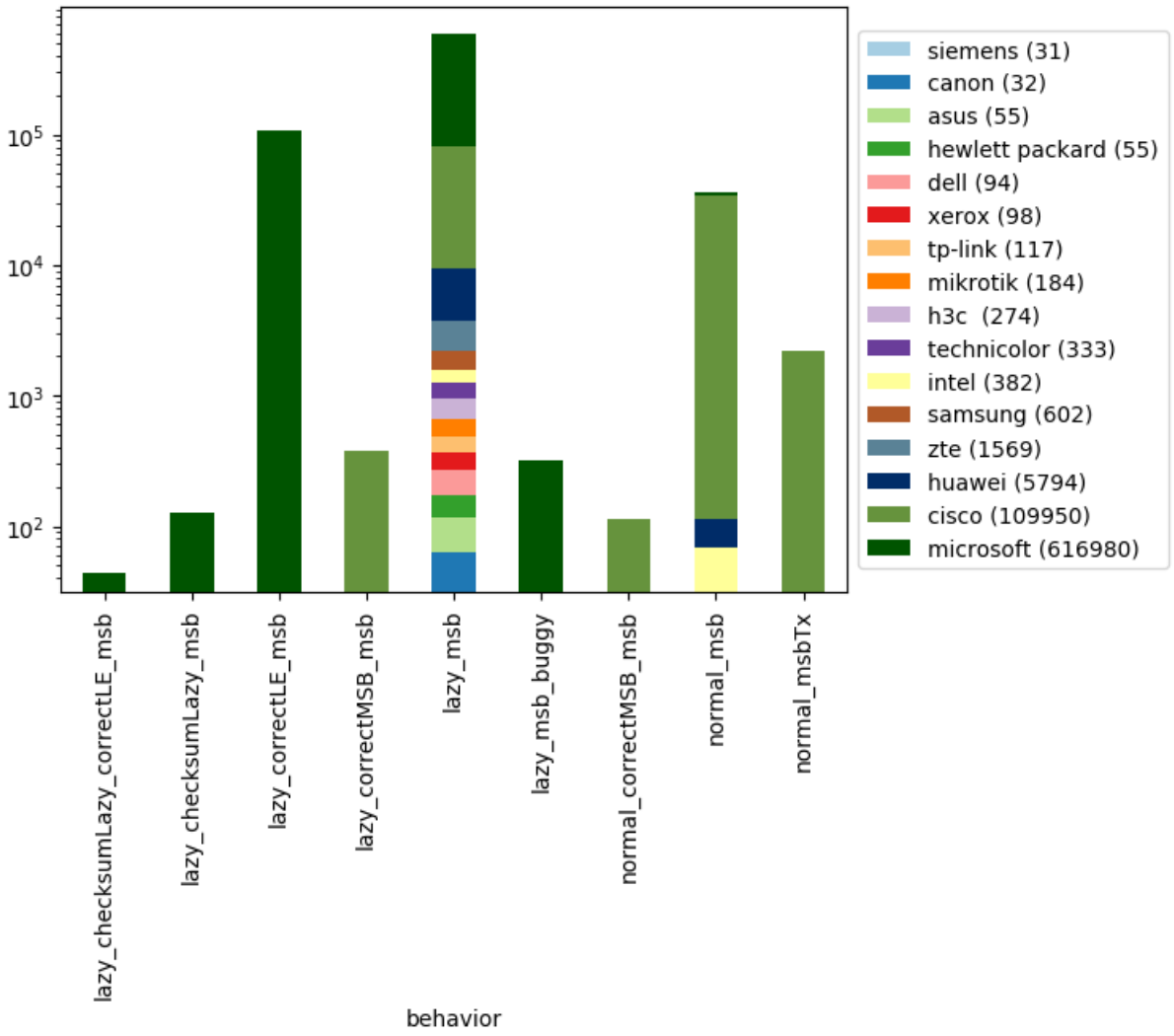


Figure G.9. Manufacturers by "MSB" Behavior

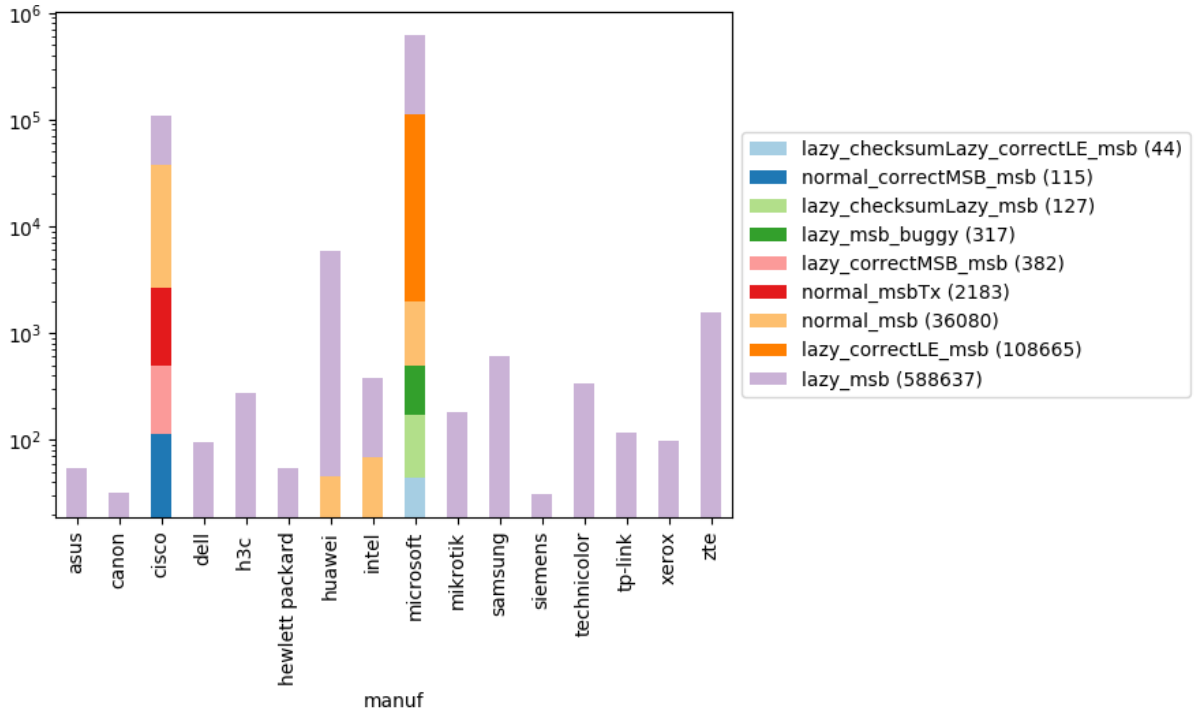


Figure G.10. "MSB" Behaviors by Manufacturer

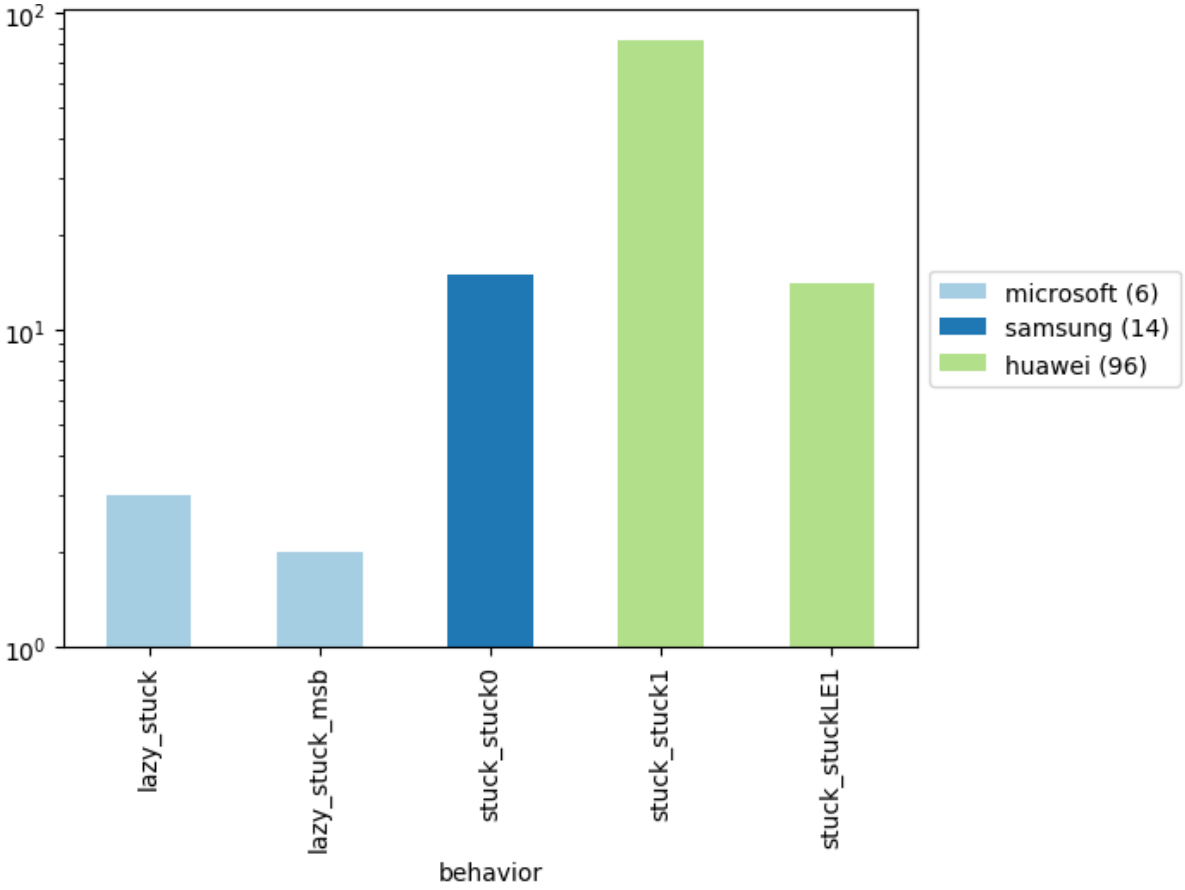


Figure G.11. Manufacturers by "Stuck" Behavior

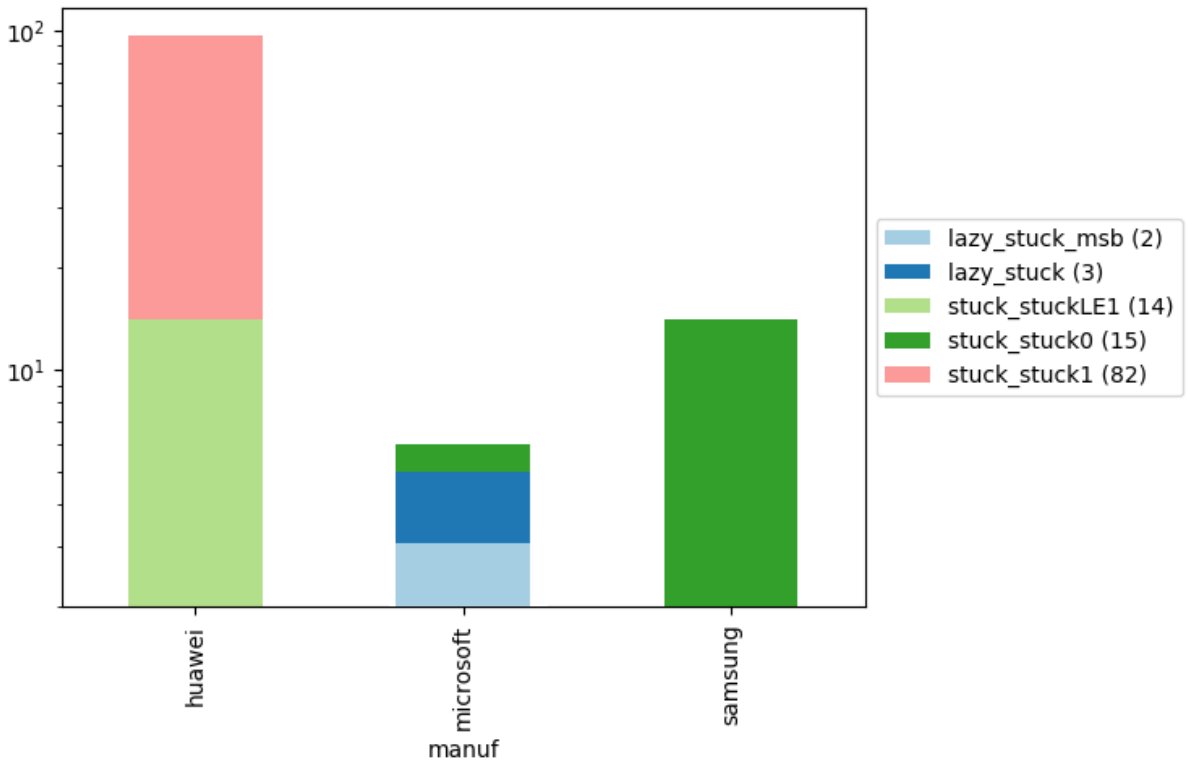


Figure G.12. "Stuck" Behaviors by Manufacturer

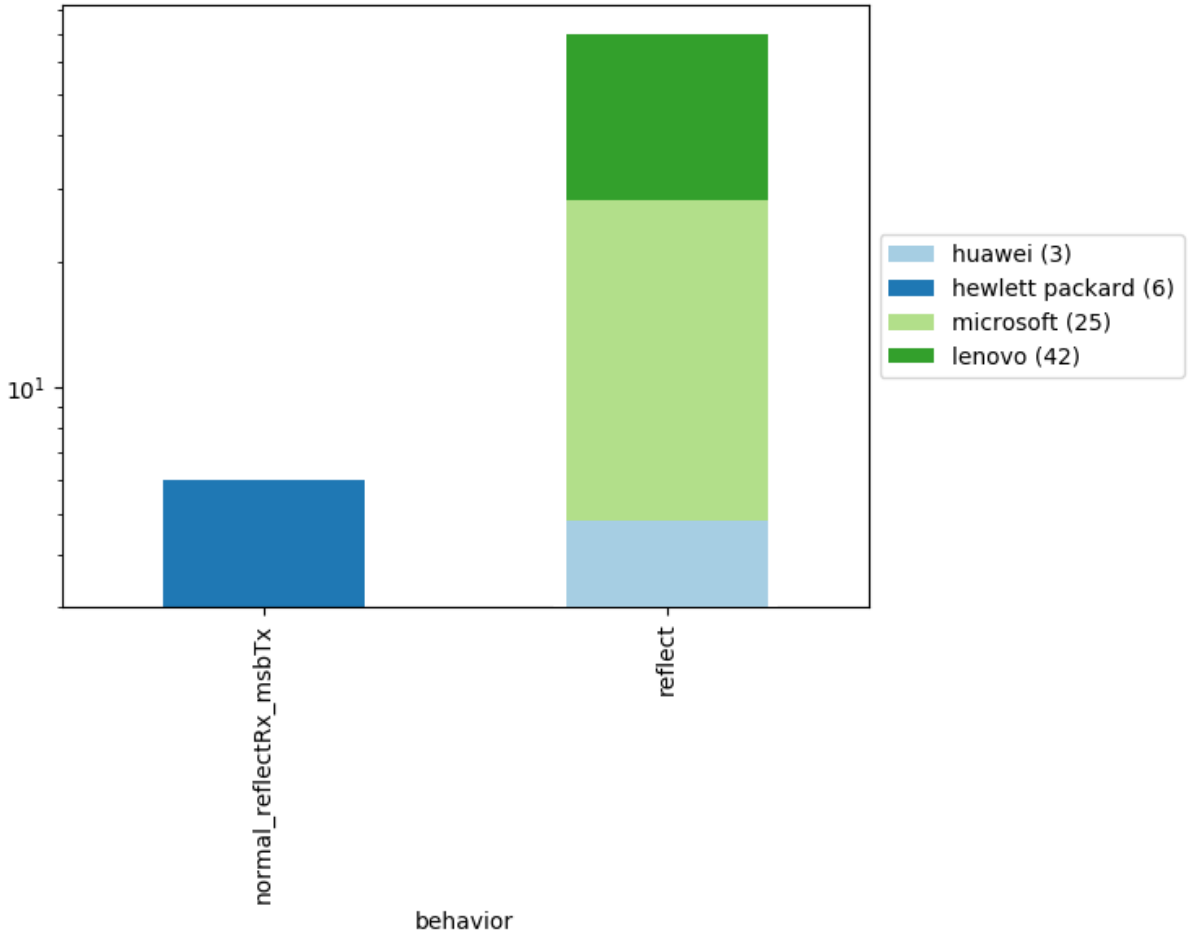


Figure G.13. Manufacturers by "Reflect" Behavior

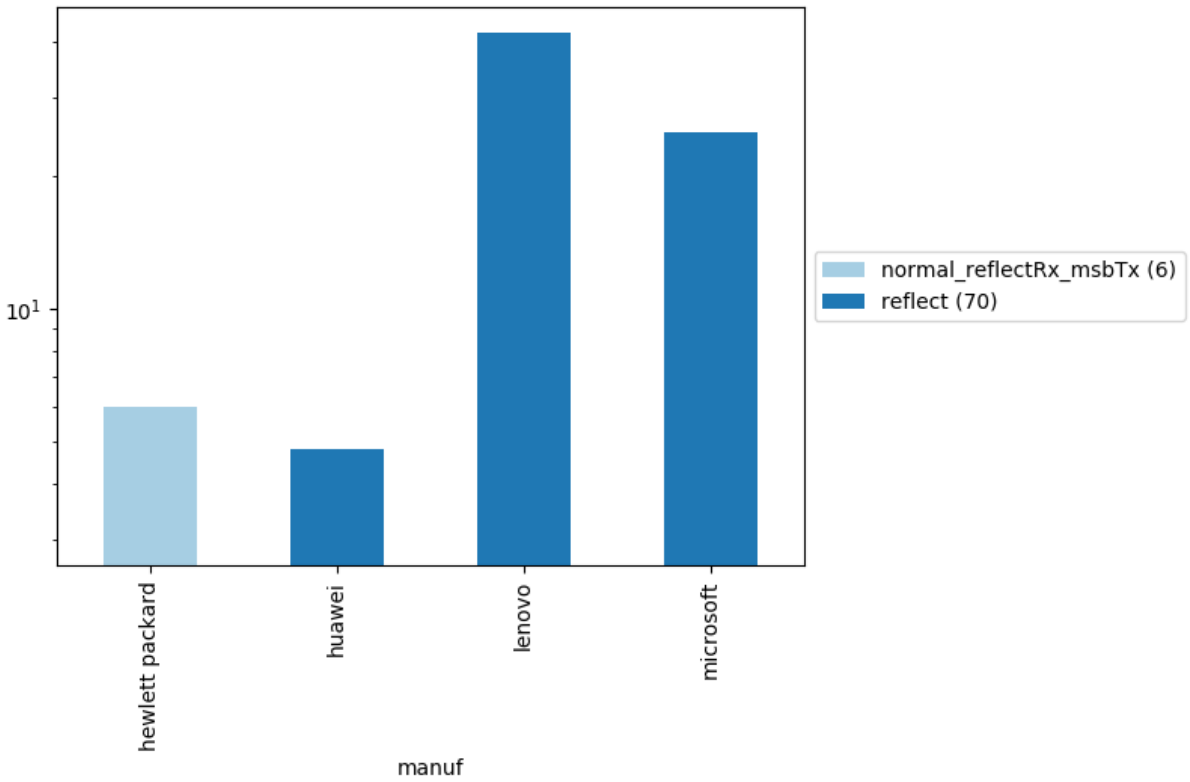


Figure G.14. "Reflect" Behaviors by Manufacturer

List of References

[1] G. F. Lyon, *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Sunnyvale, CA: Insecure.Com, 2009.

[2] E. Rye and R. Beverly, "Sundials in the shade: An internet-wide perspective on ICMP timestamps," in *Proceedings of Passive and Active Measurement Conference*, Mar. 2019, pp. 82–98. Available: https://rbeverly.net/research/papers/sundial-pam19.pdf

[3] J. Postel, "Internet control message protocol," RFC 792, Sep. 1981. Available: https://tools.ietf.org/html/rfc792

[4] *traceroute(1) - Traceroute For Linux*, 2nd ed., Oct. 2006. Available: http://manpages.ubuntu.com/manpages/trusty/man1/traceroute.db.1.html

[5] J. Heidemann, Y. Pradkin, R. Govindan, C. Papadopoulos, G. Bartlett, and J. Bannister, "Census and survey of the visible internet," in *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, 2008, pp. 169–182.

[6] Carnegie Mellon University CERT, "CA-1998-01: Smurf IP denial-of-service attacks," Jan. 1998. Available: https://resources.sei.cmu.edu/asset_files/WhitePaper/1998_019_001_496180.pdf

[7] D. Stødle, "Ping tunnel: For those times when everything else is blocked," Sep. 2011. Available: http://www.cs.uit.no/~daniels/PingTunnel/

[8] D. Mills, "DCNET internet clock service," RFC 778, Apr. 1981. Available: https://tools.ietf.org/html/rfc778

[9] D. Mills, J. Martin, J. Burbank, and W. Kasch, "Network time protocol version 4: Protocol and algorithms specification," RFC 5905, June 2010. Available: https://tools.ietf.org/html/rfc5905

[10] "CVE-1999-0524," Available from MITRE, CVE-ID CVE-1999-0524, June 1999. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0524

[11] R. Braden, D. Borman, and C. Partridge, "Computing the internet checksum," RFC 1071, Sep. 1988. Available: https://tools.ietf.org/html/rfc1071

[12] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, pp. 93– 108, May 2005.

[13] M. Cristea and B. Groza, "Fingerprinting smartphones remotely via ICMP timestamps," *IEEE Communications Letters*, vol. 17, pp. 1081–1083, June 2013.

[14] K. Anagnostakis, M. Greenwald, and R. Ryger, "cing: Measuring network-internal delays using only existing infrastructure," in *22nd Annual Joint Conference of the IEEE Computer and Communications*, 2003, vol. 3, pp. 2112–2121.

[15] F. Buchholz and B. Tjaden, "A brief study of time," in *Proceedings of the 7th Digital Forensics Research Workshop*, 2007.

[16] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman, "A search engine backed by internet-wide scanning," in *22nd ACM Conference on Computer and Communications Security*, Oct. 2015. Available: https://censys.io/data

[17] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast internet-wide scanning and its security applications," in *Proceedings of the 22nd USENIX Security Symposium*, 2013.

[18] E. C. Rye, "Sundial ICMP timestamp inference tool," 2019. Available: https://www.cmand.org/sundial

[19] E. Aben, R. Beverly, F. Bustamante, B. Donnet, T. Friedman, M. Fomenkov, P. Haga, M. Luckie, and Y. Shavitt, "CAIDA macroscopic internet topology data kit," Mar. 2018. Available: https://www.caida.org/data/internet-topology-data-kit/

[20] Y. Xie, F. Yu, K. Achan, E. Gillum, M. Goldszmidt, and T. Wobber, "How dynamic are IP addresses," in *Proceedings of SIGCOMM '07*, 2007.

[21] "IEEE registration authority assignments," 2019. Available: https://regauth.standards.ieee.org/standards-ra-web/pub/view.html

[22] J. Martin, E. Rye, and R. Beverly, "Decomposition of MAC address structure for granular device inference," in *Proceedings of the 30th Annual Computer Security Applications Conference* (ACSAC '16), Dec. 2016.

[23] "Sanitized IEEE OUI data," Sanitized oui.txt, June 2019. Available: https://linuxnet.ca/ieee/oui/

[24] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear, "Address allocation for private internets," RFC 1918, Feb. 1996. Available: https://tools.ietf.org/html/rfc1918

[25] J. Weil, V. Kuarsingh, C. Donley, C. Liljenstolpe, and M. Azinger, "IANA-reserved IPv4 prefix for shared address space," RFC 6598, Apr. 2012. Available: https://tools.ietf.org/html/rfc6598

[26] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "Hypertext transfer protocol – HTTP/1.1," RFC 2616, June 1999. Available: https://tools.ietf.org/html/rfc2616

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California