Measurement and Analysis of Hajime, a Peer-to-peer IoT Botnet

Stephen Herwig[1] Katura Harvey[1,2] George Hughey[1] Richard Roberts[1,2] Dave Levin[1]
[1]University of Maryland [2]Max Planck Institute for Software Systems (MPI-SWS)
{smherwig, katura}@cs.umd.edu, ghughey@terpmail.umd.edu, {ricro, dml}@cs.umd.edu

***Abstract*—The Internet of Things (IoT) introduces an unprecedented diversity and ubiquity to networked computing. It also introduces new attack surfaces that are a boon to attackers. The recent Mirai botnet showed the potential and power of a collection of compromised IoT devices. A new botnet, known as Hajime, targets many of the same devices as Mirai, but differs considerably in its design and operation. Hajime uses a public peer-to-peer system as its command and control infrastructure, and regularly introduces new exploits, thereby increasing its resilience.**

We show that Hajime's distributed design makes it a valuable tool for better understanding IoT botnets. For instance, Hajime cleanly separates its bots into different peer groups depending on their underlying hardware architecture. Through detailed measurement—active scanning of Hajime's peer-to-peer infrastructure and passive, longitudinal collection of root DNS backscatter traffic—we show that Hajime can be used as a lens into how IoT botnets operate, what kinds of devices they compromise, and what countries are more (or less) susceptible. Our results show that there are more compromised IoT devices than previously reported; that these devices use an assortment of CPU architectures, the popularity of which varies widely by country; that churn is high among IoT devices; and that new exploits can quickly and drastically increase the size and power of IoT botnets. Our code and data are available to assist future efforts to measure and mitigate the growing threat of IoT botnets.

I. Introduction

The proliferation of the Internet of Things (IoT) has resulted in a sudden increase in the number of network-connected devices in people's homes, spanning devices such as network-accessible web cams to network-controllable thermostats and light bulbs [37].

Unfortunately, these devices are often highly vulnerable to attack, and have been used to create large, powerful botnets. Most famously, the Mirai botnet [8] logged into open Telnet services on devices configured with default credentials. The combination of the ease of infection with the fast proliferation of IoT devices has led to large botnets and powerful attacks. In 2016, Mirai [8] was used to launch some of the largest recorded DDoS attacks in history, including a 623 Gbps attack [4] against `krebsonsecurity.com`, and a 1.2 Tbps attack [15] against Dyn, a large DNS provider.

Network and Distributed Systems Security (NDSS) Symposium 2019
24-27 February 2019, San Diego, CA, USA
ISBN 1-891562-55-X
https://dx.doi.org/10.14722/ndss.2019.23488
www.ndss-symposium.org

While there have been in-depth studies into the kinds of attacks that IoT botnets have launched [8], as well as the vulnerabilities that make IoT botnets able to flourish [6], [37], there remains much to understand about the underlying makeup of the vulnerable IoT devices themselves. For example: In what countries do more vulnerable devices reside? IoT devices are often embedded devices; what architectures are more vulnerable, and where are various architectures more popular? How large can a global IoT botnet grow today? Can an IoT botnet propagate software updates rapidly and thoroughly?

To answer these questions and more, we present in this paper an in-depth measurement and analysis of a recent IoT botnet called *Hajime*. Hajime is a contemporary of Mirai [16]; early versions of Hajime emulated many of Mirai's infections [6], but it is distinguished in three key ways. First, rather than use a centralized command-and-control (C&C), Hajime uses a popular peer-to-peer (P2P) distributed hash table (DHT) to disseminate software updates to its bots. Second, Hajime supports a far wider set of access methods than Mirai did—including updates from the recent Vault7 leak [3]. Third, Hajime uses a custom protocol for disseminating files, which includes exchanging a public key, thereby allowing us to measure the botnet by long-lived keys rather than ephemeral IP addresses.

What makes Hajime remarkable is not the attacks it has launched—in fact, Hajime has not launched *any* attacks, and has to date only been used to self-propagate. Rather, Hajime is worthy of study because its design *makes it possible for researchers to learn more about the ecosystem of vulnerable IoT devices*. For example, Hajime explicitly targets different CPU architectures (various versions of `arm` and `mips`), and makes different software updates available for different architectures. As a result, by measuring their C&C infrastructure, we are able to infer each bot's architecture type, and to track device types across different countries.

We present several novel datasets and measurement methodologies that have allowed us to measure the Hajime botnet since December 2016. These datasets include: (1) fully enumerating Hajime's use of the DHT every 16 minutes for four months, (2) actively handshaking all Hajime bots to obtain a total of 10,536,174 unique bot keys, (3) passively collecting backscatter data from a root DNS server for over 15 months, and (4) reverse engineering the Hajime binaries. Our datasets provide a more complete and more longitudinal view of Hajime than any prior study [45], [34].

We analyze these datasets to gain new insights into IoT botnets. Our findings include:

- The number of IoT devices that Hajime has compromised is larger than previously reported [18], [34], [29], [45]. After Hajime deployed a new exploit, we find upwards of ~95,000 active bots.
- There is a wide assortment of CPU architectures running vulnerable software, and the popularity of architectures can vary widely by country. For example, most bots in the US are `arm5`, while most in Brazil are big-endian `mips` devices. This has profound impact on the future design of honeypots for IoT botnets.
- There is considerably high churn among IoT devices, with a median bot lifetime of ~5 hours, indicating that devices are often rebooted and reinfected. This presents both challenges and opportunities for future defenses.
- New vulnerabilities risk drastically and quickly increasing the size and power of IoT botnets. In late March 2018, Hajime incorporated the new ChimayRed exploit [3], [1] and almost immediately doubled in number of active bots.

Although Hajime has not (yet) been used to launch an attack, the results from our measurement study offer *fundamental* insights about IoT botnets in general: *First*, by observing three distinct exploits over time, we show that new exploits can have profound and swift effects on the composition (size, hardware, and geography) of a botnet composed of heterogeneous devices. *Second*, by observing that different countries' IoT devices prefer different architecture types, we show that some IoT exploits may affect some countries more than others (and not necessarily in accordance with who has the most IoT devices overall). *Finally*, by analyzing Hajime's decentralized C&C, our work highlights the challenges inherent in containing and disrupting this latest generation of IoT botnets.

The rest of this paper is organized as follows. We review related work in Section II. In Section III, we provide relevant background on Hajime's operating logic In Section IV, we describe our data collection methodology and our datasets. In the subsequent sections, we analyze the Hajime botnet's size (§V), churn rates (§VI), geographic locations (§VII), and device types (§VIII). We measure how effective Hajime is at pushing out new software updates in Section IX. In Section X, we use a passive DNS dataset to analyze a specific vulnerability that Hajime exploited. Finally, we conclude in Section XI.

II. Related Work

IoT botnets are a relatively new and growing phenomenon. The Carna project [7] from 2012 is arguably the first of this ilk; this research-oriented botnet propagated by scanning for Telnet services with empty or default credentials, and ran Internet measurement studies on the some 420K of its infected devices. Following Carna, a series of IoT malware [21], [44], [2] similarly infected a broad range of embedded devices by either brute forcing Telnet credentials or exploiting unpatched vulnerabilities. The purpose of these botnets ranged from vigilante efforts in securing devices to cryptocurrency mining.

The most immediately related prior work are the studies of the Hajime botnet performed by Kaspersky Labs [45], Radware [34], and Symantec [18] in the wake of the Hajime discovery. These prior studies involved a combination of honeypots, short-term measurements of the P2P network, and reverse engineering of the botnet payloads. By comparison, we perform longer and broader studies of the network dynamics of Hajime, allowing us to observe changes in activity across botnet updates, and the impact of the deployment of new exploits.

Last year, Antonakakis et al. [8] performed a broad study of the Mirai botnet, including its size, its C&C infrastructure, the devices that it comprises, and the attacks that it was used to launch. Theirs was the first formal step towards studying and understanding IoT botnets; we extend it in several key ways.

First, Hajime represents a step in the evolution of IoT botnets in that it makes use of a much more sophisticated, P2P C&C infrastructure, and differentiates bot behavior based on the device's architecture. We present techniques for scanning the infrastructure and analyze the relationship of architecture to geographical dispersion. Second, Hajime has steadily incorporated new access vectors. Using several datasets, we analyze the impact of these new access vectors on the botnet size, location, and composition. Finally, thanks to Hajime's design, we are able to uniquely identify bots by public keys, alleviating us from the common concern of mis-counting IP addresses [25].

Another closely related botnet to Hajime is the widely studied Storm botnet [24], [22], [19], [38], [23]. Storm used an existing Kademlia-based DHT to distribute C&C information in a similar manner to Hajime, and similar active DHT measurements were conducted. Our paper demonstrates that even these resource-constrained IoT devices are using some of the same mechanisms.

More broadly, there have been many studies of live botnets and the spread of malware. Staniford et al. [41] measured the quick spread of Internet worms; we also observe dramatic changes in Hajime's size within just an hour of the deployment of a new exploit. Stone-Gross et al. [43] took control of a botnet's centralized C&C infrastructure for a short period of time, and used it to measure the number of bots and the various sensitive information that they report back to their bot master. Conversely, we study Hajime longitudinally, over the course of over a year; so doing allows us to see the spread of a botnet over time. Other studies have infiltrated other botnets such as MegaD [11] and Waledac (a Storm successor) [42] to gather information on its C&C architecture and its spam operations. Compared to such studies, we present a different methodology; rather than infiltrating or taking over the botnet, we leverage Hajime's P2P design to directly measure from outside.

Like many prior botnet studies [43], [8], [9], [48], we explore Hajime's size. These prior approaches caution against using weak identifiers like IP addresses in estimating botnet sizes [35], [25]. Padmanabhan et al. [31] performed an in-depth study of how IP addresses change—that is, how often ISPs assign new IP addresses to their customers. They found that ISPs can vary widely in how rapidly they perform IP address reassignment, and that there are some country-level trends. For example, they found that ISPs in Germany tend to

change addresses every 24 hours. We address these concerns by leveraging the fact that Hajime bots use public keys as part of their P2P communication, which may be used as long-lived identifiers.

There is also a broad set of tangentially related work at the intersection of P2P and botnets, such as studying botnets that *construct* their own P2P network [19], [10], or constructing P2P networks that are resilient to attacks [12], [32], [47].

III. Hajime Background

In this section, we give a brief overview of Hajime's design and operation. Many of these details have been discussed in blog articles and security reports elsewhere [26], [16]; to provide background, we include the aspects that are most relevant to our study.

A. Binary files

Hajime consists of two Linux ELF executables: (1) the core *implant* (`.i`) performs all P2P tasks (described later), and (2) the *attack* module (`atk`) performs the scanning and propagation. These two modules run on all Hajime bots, and communicate with one another using IPC. Both modules have versions available for the `arm5`, `arm6`, `arm7`, `mipseb`, and `mipsel` CPU architectures. Bots download and host the binaries specific to their architecture.

1) Propagation logic: We have reverse engineered the Hajime binaries to verify that the `atk` module scans the IPv4 address space uniformly at random, with the exception of a few blacklisted IP blocks. Like Mirai, Hajime does not scan invalid, private, reserved, multicast, or IANA special use addresses, the US Postal Service, General Electric, Hewlett Packard, or the US Department of Defense. However, for unknown reasons, Hajime also excludes scanning 77.247.0.0/16, 85.159.0.0/16, and 109.201.0.0/16. These network blocks are owned by many different ISPs, mainly in Europe and the Middle East. There is however only *one* ISP in common with each of these address blocks: NFOrce Entertainment (ASN 43350), located solely in the Netherlands. This result, coupled with the fact that the first IP address we see launching the TR-064 attack (Section X-C) is in the Netherlands, leads us to speculate that that is where the botmaster's operations originate.

2) Exploits: One of the most interesting and useful features of Hajime is that it deploys a wide range of exploits across an even wider range of architecture types. We provide an overview in Table I.

The ports that the `atk` scans depend on the `atk`'s own architecture, as each architecture bundles a different set of access methods that the botmaster has added over time. Initially, scanning for all architectures was limited to port 23 (Telnet), with the `atk` using a small dictionary of common default credentials to attempt access. This method was later expanded to cover port 5358 (Telnet alternative), as well as to incorporate the ARRIS password of the day [39] into the credentials dictionary.

In 2017, the `mipseb` and `mipsel atk`s added an exploit targeting the TR-064 service on port 7547 (CVE-2016-10372). We describe this exploit in Section X-A, and analyze its deployment in Section X-C. That same year, the `arm7 atk` added scanning for HTTP services on port 81; specifically, the `atk` attempts access via a default password if it determines the webserver is GoAhead-Webs, and via a remote code execution (RCE) exploit for CCTV-DVRs running the Cross Web Server. The `arm5 atk` added scanning for MCTP services on port 9000, specifically targeting KGuard Digital Video Recorders using CVE-2015-4464.

Architecture	Port	Service	Method
`mipseb`	23, 5358	Telnet	credentials
	7547	TR-064	CVE-2016-10372
	many	HTTP	Chimay-Red
	80	HTTP	CVE-2018-10561,-10562
`mipsel`	23, 5358	Telnet	credentials
	7547	TR-064	CVE-2016-10372
`arm7`	23, 5358	Telnet	credentials
	81	HTTP	GoAhead-Webs credentials
	81	HTTP	Cross Web Server RCE
`arm6`	23,5358	Telnet	credentials
`arm5`	23, 5358	Telnet	credentials
	9000	MCTP	CVE-2015-4464

TABLE I: Hajime's architecture-specific access methods and the corresponding ports scanned

In March 2018, the `mipseb atk` started scanning ports 80, 81, 82, 8080, 8081, 8082, 8089, 8181, 8291, and 8880 for MikroTik routers, using the Chimay-Red exploit for access—an exploit taken from the Vault7 CIA leaks [3]. We describe the effect of the Chimay-Red exploit on Hajime's size (§V), churn (§VI), and geographical dispersion (§VII).

Most recently, on May 8, 2018, the `mipseb atk` started scanning port 80 for GPON (Gigabit-capable Passive Optical Network) routers manufactured by South Korea-based DASAN Zhone Solutions, using a pair of vulnerabilities disclosed by VPN Mentor just 10 days prior [46]. The vulnerabilities, detailed in CVE-2018-10561 and CVE-2018-10562, enable an attacker to bypass the device's HTTP server authentication by appending `?images` to a URL, and to carry out a shell injection via a request to a specific-URL that otherwise required authentication. We describe the effect of the exploit on Hajime's location (§VII) and device composition (§VIII).

As Table I demonstrates, the set of access methods used varies widely by architecture type. This has profound impact on future designs of honeypots for IoT networks: to capture all behavior, one may have to deploy a honeypot of myriad hardware architectures.

3) Attacks: Hajime has not launched any DDoS attacks to date, and there is speculation that the botnet may be the work of a whitehat hacker [16].

B. P2P Network

For C&C, Hajime uses BitTorrent's DHT overlay network, which is an implementation of the Kademlia DHT protocol [27]. This overlay network is the basis for BitTorrent's *trackerless* torrents [5].

In Kademlia, each node has a unique 20-byte node ID, and joins the DHT by bootstrapping with a known peer. To find a value in the DHT, a client performs a lookup of the value's key. In BitTorrent, the keys are hashes of the torrent's *info_hash* (a SHA-1 hash of the torrent's metadata) and the value is the list of peers serving (*seeding*) the corresponding torrent. Thus to start downloading (*leeching*) a torrent, a BitTorrent client looks up that torrent's info_hash in the Kademlia DHT. Once the client has begun downloading the torrent, the client then *announces* itself as a *seeder*, an operation which adds the client to the torrent's peer list.

1) Looking up files: Hajime joins the DHT using well-known peers as a bootstrap[1]. In contrast to info_hashes, Hajime uses as the key a hash of the payload filename concatenated with the current day's timestamp. This way, bots know what new info_hash to look up each day. The value stored under this key is a list of IP addresses and ports of peers that have announced themselves as seeders, signaling to other Hajime bots that the payload may be downloaded from them. Despite the semantic differences in the identifier key, it is important to note that this DHT is the same overlay network used by BitTorrent, and thus it is not solely populated by Hajime bots.

2) Transferring files: Hajime bots download files from other bot peers using a custom application protocol layered on top of the uTorrent Transport Protocol (uTP) [30]. Critical to our study is the protocol detail that, as part of an initial handshake, bots exchange their public keys. As we discuss in Section IV, we leverage this to obtain long-lived bot identifiers instead of having to use ephemeral IP addresses as identifiers.

C. Bot Lifecycle

Immediately upon infection, Hajime blocks ports 23, 5358, 5555, and 7547 using iptables, presumably to prevent reinfection. Upon execution, the `.i` joins BitTorrent's Kademlia DHT and starts seeding itself as well as the daily config file (serving a blank config file until an actual config is downloaded). The `.i` also generates a unique 256-bit public key for uTP communications, which is used to optionally RC4-encrypt a uTP connection.

At midnight, the `.i` looks up in the DHT the config file. The config file contains a signed list of the current version of the `atk` and `.i` for all architectures. Upon downloading a config file, a bot scans the file's list of binaries, then looks up, downloads, and executes any new `.i` or `atk` matching the bot's architecture.

When the `.i` executes a new `atk`, it kills the old `atk` (if there is currently one running); over the IPC channel, the new `atk` then requests that the `.i` also announce the `atk`. When the `.i` executes a new `.i`, the old version exits, and the new version rejoins the DHT and generates a new uTP key.

IV. Active Scanning Methodology and Datasets

Throughout this paper, we develop four novel datasets to measure and analyze Hajime. In this section, we describe two, both of which involve actively scanning.

[1]router.utorrent.com and router.bittorrent.com

A. Active BitTorrent DHT Measurement

Hajime bots join a public DHT and use predictable identifiers by hashing timestamps and filenames. This enables us to join the same DHT swarm as Hajime bots and to collect two datasets via active measurements: the set of bots stored in the DHT as hosting the file (the *seeders*), and the set of bots that search for seeders and request files via uTP (the *leechers*).

1) Seeder collection: To identify Hajime seeders, we compute the same identifier keys as the bots, and exhaustively look these keys up in the Kademlia DHT. Peers in the DHT maintain a list of peers who have *announced* themselves as seeders. If the peer list stored for a given key is too large to fit into a single lookup response, DHT nodes will respond to a lookup request by returning only a random subset of their list. Therefore with the goal of obtaining the *full* peer list, we continue to send requests until the following heuristic is met: when a node has sent us at least three times as many responses with no new peers as responses with new peers, we decide we have its complete peer list for the given key, and stop sending requests.

We anecdotally verified this heuristic with the coupon collector's problem, which provides a formula to determine, on average, how many randomly selected peers we should obtain, with replacement, before we have seen all peers. Contrary to our problem, the formula assumes the total number of peers is known, so we spot checked that, given the number of distinct peers we discovered, we had received at least as many total peers as the coupon collector's problem.

We repeatedly perform these scans, in order to map the behavior of Hajime over time. Hajime uses the KadNode implementation [28] of Kademlia. KadNode maintains announced IP addresses for 32 minutes; so as not to miss expired addresses, we perform exhaustive lookups at a shorter interval, namely, every 16 minutes.

Since the key is computed based on the current day's timestamp, and bots may have incorrectly synchronized clocks, we look up keys for a five day range (two days in the past through two days in the future).

2) Leecher collection: In addition to scanning the full list of seeders, we also passively collect leechers by announcing that we are seeding files. Concretely, we announce our own hosts for the same five days worth of identifiers, and record the IP addresses and ports of the bots that connect to us. We do not serve any files in return. Unlike with the seeder dataset, we do not expect to find a complete set of Hajime leechers in the DHT, as there are many potential seeding bots which a leecher may contact instead.

3) Per-architecture downloads: We actively download the config file from random samples of the bots, and then simultaneously seed and leech for any new modules listed in the config. Figure 1 shows the updates we recorded during the period of our active measurements. Hajime did not push updates from October 30, 2017 through March 17, 2018. From March 18 through May 31, we downloaded 50 config updates, comprising 47 new modules (34 `atk`s and 13 `.i`s). Twenty-nine configs update a single module, eight update multiple modules, and 13 do not update any of the modules. The `mipseb` architecture is by far the most updated, accounting

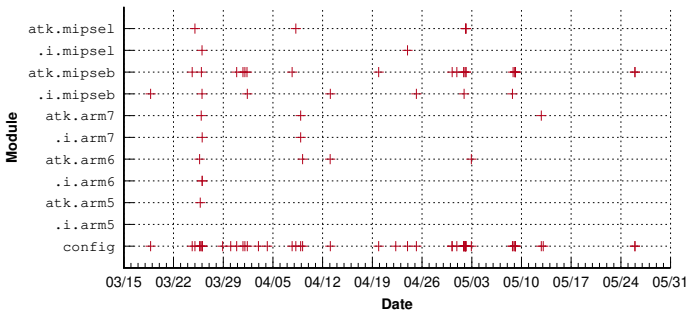


Fig. 1: Hajime updates. There were 50 config updates during the period 03/18/2018 – 05/31/2018.

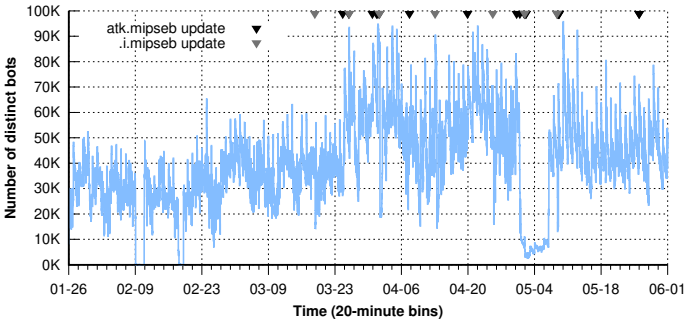


Fig. 2: Number of unique Hajime bots. (Active scans.)

for 28 of the new modules, followed by `mipsel` (7), `arm6` (6), `arm7` (5), and `arm5` (1). We analyze update propagation in Section IX.

We collect results on three different machines, and report measurements from 00:00 January 26, 2018 through 00:00 June 1, 2018 UTC. Through our active measurement datasets, we saw 5,404,045 unique IP addresses, 5,293,415 from the seeder dataset and 2,993,480 from the leecher. We present detailed analysis of these measurements in Sections V through VII.

B. Active uTP Scans

As we generate logs of the leechers and seeders of the Hajime files, we immediately initiate uTP connections with the logged bots, perform a key exchange, and then close the connection. We then annotate our leecher and seeder logs with the retrieved key.

We started exchanging keys with the set of current seeders and leechers on January 26, and again report results for four months. During this time period we saw 10,536,174 unique bot keys.

C. Ethical Considerations

To the best of our knowledge, our measurements do not disrupt the DHT, the bots, or the devices on which the bots execute. When possible, we notify ISPs of Hajime infections within their networks.

V. Botnet Size

We begin our analysis by investigating the size of Hajime using our active scanning datasets. To arrive at an accurate count of *bots*, and not a potentially misleading count of *IP addresses* [25], [35], we report the number of unique keys we obtained through active probing via uTP, equating a *distinct key* as a *distinct bot*. One ramification of this is that, when a machine changes its key (due to a `.i` upgrade or reboot and reinfection), it will appear to us as a new bot.

Figure 2 presents the number of distinct bots we see in each 20-minute interval over the course of four months (Jan. 26, 2018 – May 31, 2018). Updates to the `mipseb atk` and `.i` modules are marked along the top of the graph. Up until March 25, the overall shape remains relatively stable, hovering at about 40,000 bots.[2]

Antonakakis et al. [8] studied the number of active Mirai bots through most of 2016 and early 2017. They reported a steady-state of approximately 200,000–300,000 bots in 2016 before a steady decline to 100,000 in 2017.[3] For the sake of comparison, we note that on May 31, 2018 (the last day in our dataset), we reported an average of 42,042 bots per 20-minute bin, and an average number of 35,261 unique IP addresses per bin.

On March 25, the shape changes drastically, spiking up to nearly double the numbers seen previously, and then dropping off slightly with much higher fluctuation. We attribute this spike to a `mipseb atk` update released on March 24, 15:10 UTC that added the Chimay-Red exploit. This spike demonstrates IoT devices' ability to quickly propagate malware; there were an average of 42,674 bots per 20-minute bin during the last hour of March 24; this average jumps to 71,026 for the first hour of March 25, and achieves a local maximum of 93,467 active bots at the start of March 26. With the GPON exploit on May 8, we see a similar spike, and an overall max of 95,746 active bots just 31 hours after Hajime deployed the exploit.

This suggests that the hosts vulnerable to these attacks were quickly and thoroughly scanned and infected. We evaluate what kinds of devices these were in Section VIII; in Section X, we revisit Hajime's quick infection rate from the perspective of the TR-064 exploit.

Another notable feature of this plot is the high variability in number of bots we see. This may be explained in part because we conservatively only count a bot if we are able to actively connect to it and retrieve a uTP key. It is possible we see some keys in both a prior and a future bin, but either fail to see them in our DHT measurements or fail to directly connect. Although the bot is still *alive*, we omit it from bins in which we do not actively connect to the bot, so as not to overestimate the number of *active* bots at a time.

The stability of the shape before March 25 is notable, especially in contrast to the impact of the Chimay-Red `atk`

[2]The gaps in our data on February 9 and February 18, as well as the marked decrease in active bots from April 30 – May 6, are due to errors in our data collection on those days.

[3]Antonakakis et al. [8] use a network telescope to determine botnet size, and flag a bot as inactive after 20-minutes of inactivity; hence we use 20-minute bins to allow a more apples-to-apples comparison with that paper.

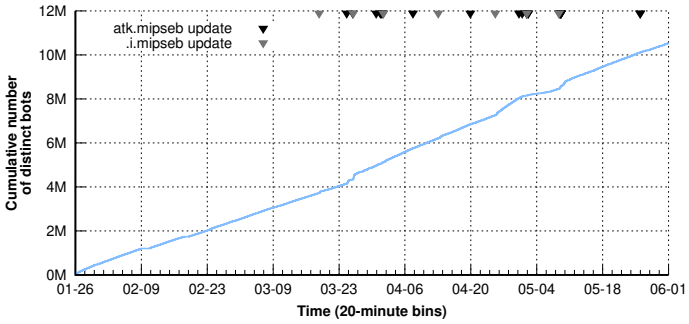


Fig. 3: Cumulative number of unique Hajime bots. (Active scans.)

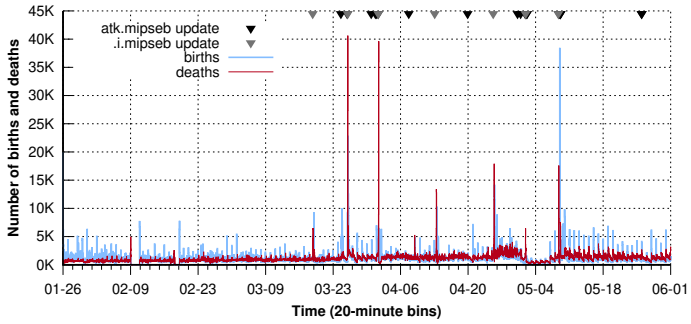


Fig. 4: Bot churn in Hajime. "Births" denote the first time we saw a given key, "deaths" denote the last time. (Active scans.)

update. One possible explanation for the general stability of the number of bots during this period would be if Hajime had "stabilized" around the same set of about 40,000 bots. To evaluate this hypothesis, we show in Figure 3 the cumulative number of distinct Hajime bot keys that we have observed over this four month interval. Were they a stable set of bots, this cumulative distribution would reach its peak on the first day. On the contrary, we see a surprisingly smooth, steady increase in the number of new bots over time.

As expected, we see a jump on March 25 corresponding to the spike in active bots. But the line quickly returns to nearly its former slope, reinforcing Hajime's rapid infection rate. However the slope after the jump remains slightly steeper than before, and we see another increase in slope around April 25, despite the total number of bots generally trending down over this period. As we show next, the smooth growth in the number of new hosts, as well as the steeper cumulative growth despite falling hourly numbers, can be largely attributed to high churn rates.

VI. Bot Churn

A distributed system's *churn* is the rate at which hosts enter and leave the system. To study churn in Hajime, we define the *lifetime* of a bot as the difference in time from when we last observe a uTP key (the *death* time) and when we first observe that key (the *birth* time). Keys that we only observe once therefore have a lifetime of 0.

Figure 4 shows the number of births and deaths observed in 20-minute intervals from our active DHT scans. We observe a consistent churn of around 2K bots during any given 20-minute interval through March 18.[4] On March 19, there is a spike in births and deaths corresponding to the release of a `mipseb .i` update. A `.i` update results in such a spike because a bot generates a new uTP key when executing the new `.i`. We observe this spike in both births and deaths for all `mipseb .i` updates.

We observe a spike in births without the accompanying spike in deaths on March 25, corresponding to the `mipseb atk` update containing Chimay-Red. With an `atk` update, bots do not change their keys, but begin executing a new scanning and infection module. We can once again see the strong impact adding a new attack vector had on the network.

[4]The spike in deaths and births near February 9 and February 18 are caused by measurement artifacts due to the gaps in our dataset.

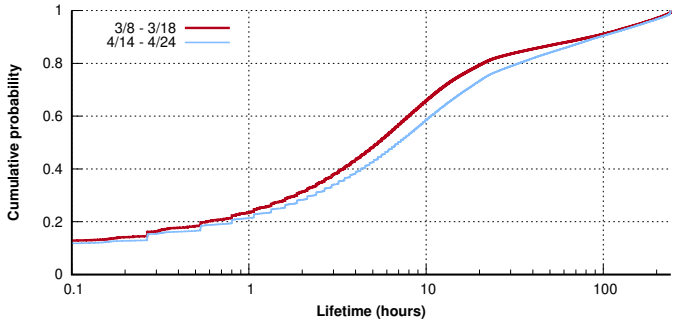


Fig. 5: CDF of a Hajime bot's lifetime during two "stable" periods with no `.i` updates. (Active scans.)

Between March 25 and April 2, we see rhythmic birth and death rates, following a diurnal pattern. On March 26, the botmaster released an update to the `mipseb .i` that added a persistence mechanism, enabling Hajime to run upon device reboot. We speculate that the diurnal pattern is the botmaster testing this mechanism. On April 2, the botmaster published a new config with the message *"Give me a place to stand and with a lever I will move the whole world"*. It is also interesting to note that the corresponding `.i` update on April 2 seems to stabilize the birth and death rate.

We also observe that the more stabilized birth and death rate after April 2 does not increase compared to before the March 24 update, which would occur if the new set of bots targeted by the latest attack vector were born and died at similar rates to the previous set of bots. Instead we see an initial spike in births, followed by a return to close to the previous rate, indicating the new bots remain in the network for longer.

To better understand these rates and what impact updates had on bot lifetimes, Figure 5 shows the cumulative fraction of bot lifetimes during two 10-day "stable" periods with no `.i` updates. We observe that the median lifetime increases from 5.6 to 6.9 hours, and that, whereas 18.3% of bots have lifetimes of longer than 1 day before the deployment of Chimay-Red, 23.7% of bots have lifetime longer than 1 day after deployment. This reinforces the observation from Figure 4 that the new population of bots have longer lifetimes.

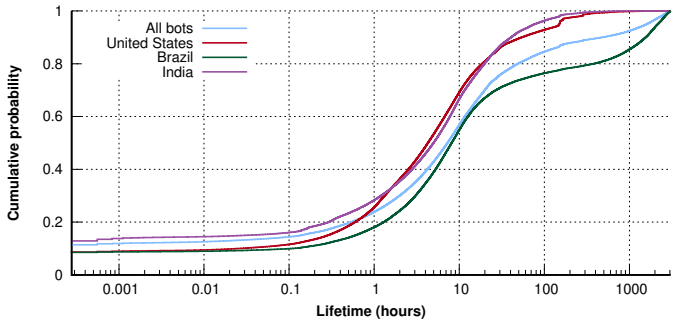


Fig. 6: CDF of a Hajime bot's lifetime. We show the overall lifetime of all bots, and break it down into three of the top countries. (Active scans, 01/26/2018 – 05/31/2018.)

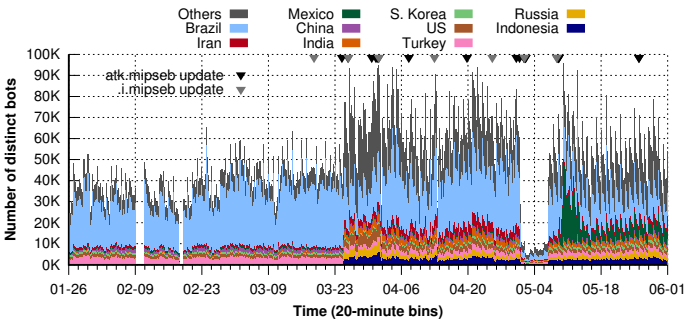


Fig. 7: Number of unique Hajime bots for the 10 most prevalent countries. (Active scans.)

In general, churn may be due to device reboot and infection. Grover et al. [20] found that reboots of home routers are somewhat common in developing countries (∼1 reboot per day), but rare in developed countries (∼1 per month). The authors of the Carna research botnet also claimed that approximately 85% of all their bots were available at any time [7]. Unfortunately, we cannot empirically verify that Hajime faces the same behavior: whereas Grover et al. and Carna measured from within the network, we would have to detect reboots remotely. Nevertheless, as an initial step in testing the reboot and reinfection hypothesis against the previously cited disparities in device uptime among countries, we present in Figure 6 a CDF of lifetimes for a developed country (United States) against that of two developing countries (India and Brazil), as well as against all bots in general. Unfortunately, Figure 6 does not demonstrate such uptime disparity between develop and developing countries, as bots in the United States generally have a shorter lifetime than in India or Brazil, thereby suggesting that other phenomena may be influencing churn.

VII. Bot Location

To understand where Hajime bots are located, we apply the MaxMind IP geolocation database[5] to each of the bot IP addresses we observe. Table II shows the top 10 countries by the number of unique bot keys we have observed. Overall, these results show a globally distributed set of bots, with a particularly heavy concentration in a small number of countries.

We also include in Table II the number of unique IP addresses we observed during our four months of active scans, as well as across two periods during which there were no `.i` updates. Since bots change their keys on `.i` updates, comparing IP addresses to keys over a period with `.i` updates leads to an overestimation of the key to IP ratio, as can be seen in the ratio computed over the full four months. It is well-known that IP addresses can be error-prone when used as long-term identifiers; some ISPs reassign IP addresses in surprisingly small intervals [31], which would result in an overestimate in the number of bots. Moreover, some countries run many hosts behind so-called "carrier-grade" NATs [36], which would result in an *underestimate* of the number of bots. We see both dynamics at play in Table II; Brazil appears to have extensive IP address reassignment, with each key appearing at 1.33 and 1.36 IP addresses on average during the first and second stable intervals, respectively. China's CGNAT deployments are also evident, with each IP address shared among 3.68 and 3.58 bots during the two stable windows.

It is also interesting to note how the key to IP ratios changed between the two time intervals. This may be due in part to a change in network-level behavior within a country. It may also be caused by a shift in the composition of devices within that country following module updates.

Our results in Table II differ from previous honeypot-based studies [45], [34], both in terms of raw number (we see a few tens of thousands more bots than these studies) and in terms of relative ranking of countries. In our active scanning datasets, Brazil [6] is by far the most prevalent attacker, constituting 52.5% of all unique bots we see, whereas Radware's honeypot study found only 10% of their hosts from Brazil. Conversely, Vietnam has appeared in the top three of other studies' most prevalent countries, yet in our data it constitutes only 0.42%, ranking 24th. The differences are likely due to methodology and time frame. In particular, we scrape information about as many active bots as possible, whereas prior studies log scanning at fixed honeypots. Moreover, these studies are from early 2017. To that end, Antonakakis et al. [8] also observed Vietnam to have the third-highest number of *Mirai* bots over their course of study (Sept. 2016 – Feb. 2017), but also observed a 90.2% decrease in bots from a particular Vietnamese ISP over that time period.

Figure 7 shows how the country composition of Hajime changed over our four-month measurement, annotated with `.i` and `atk` updates. We make several observations.

First, different infection vectors affect different countries in different ways. As Hajime's behavior is to scan the IPv4 address space randomly, the change in country composition is a reflection of the composition of vulnerabilities across different countries. The United States and Russia, in particular, have much higher numbers of bots following the Chimay-Red exploit update, reflecting a higher proportion of devices vulnerable to that attack vector. South Korea, Turkey, and Mexico, in contrast, see practically no change. On the other

[5]https://www.maxmind.com/en/geoip2-databases

[6]Nearly 40% of all unique keys we collect are from bots in Telefônica Brasil S.A (ASN 27699, 26599, and 19182).

Country	01/26 - 05/31 Keys	IPs	Keys/IP	03/8 - 03/18 Keys	IPs	Keys/IP	04/14 - 04/24 Keys	IPs	Keys/IP
Brazil	4,899,248	3,412,070	1.44	765,729	1,024,643	0.75	651,236	893,960	0.73
Iran	586,594	319,006	1.84	26,142	25,937	1.01	102,455	82,587	1.24
Mexico	533,925	197,155	2.71	20,621	3,131	6.59	12,840	3,608	3.56
China	515,061	128,970	3.99	55,845	15,166	3.68	45,154	12,603	3.58
India	431,565	88,643	4.87	3,722	5,587	0.67	67,560	25,443	2.66
South Korea	361,946	11,538	31.37	39,368	1,948	20.21	29,237	2,231	13.10
United States	310,905	31,579	9.85	25,015	3,743	6.68	24,655	8,045	3.06
Turkey	293,289	207,009	1.42	32,096	26,857	1.20	32,433	24,944	1.30
Russia	246,780	102,084	2.42	9,170	4,279	2.14	27,033	15,745	1.72
Indonesia	236,861	71,203	3.33	3,688	2,223	1.66	41,128	19,514	2.11

TABLE II: Top 10 countries with the most Hajime bots over four months, and over two stable windows with no `.i` updates. We use keys obtained through uTP scans to uniquely identify bots; IP address counts can lead to both over- and under-estimates (Active scans, 01/26/2018 – 05/31/2018.)

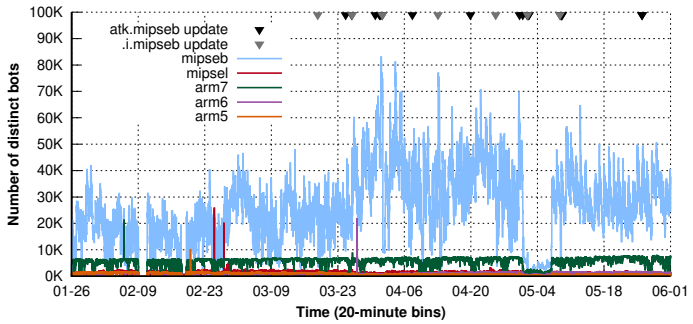
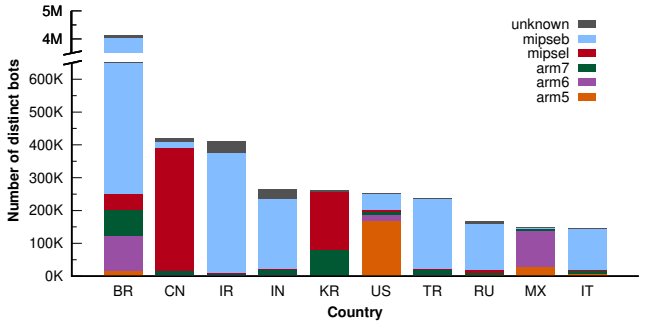
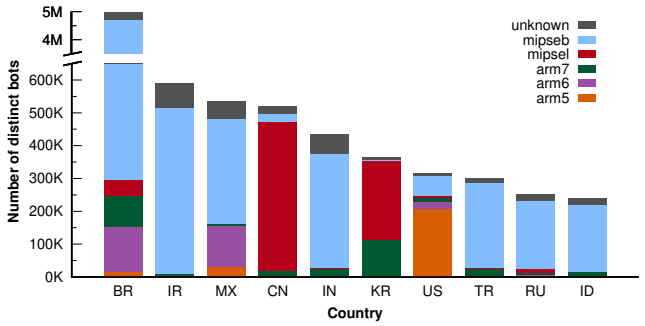


Fig. 8: Number of Hajime bots, broken down by architecture. (Active scans.)



(a) Before GPON (01/26/2018 – 04/28/2018).



(b) Before and after GPON (01/26/2018 – 05/31/2018).

Fig. 9: The architecture of bots within the top 10 overall infected countries (a) from the start of our data collection to before the GPON exploit on May 8, and (b) for the entire collection period, including the GPON exploit deployment. Note the broken y-axis for Brazil, which has an order of magnitude more bots than any other country, predominantly of `mipseb` architecture.

hand, Mexico acquires the majority of new bots following the GPON exploit deployment on May 8.

Second, these dynamics can be drastic. This can be seen clearly in the case of Russia, which had a low infection rate of roughly 500 bots each hour until the release of the Chimay-Red update. Within five hours of the release, that number had increased sevenfold to 3,699. Five hours later, during the first hour of the next day when most bots saw and fetched the update, that number rose another 60% to reach 5,963, making Russia the third most-infected country.

VIII. Device Composition

Next, we turn our study towards understanding what kinds of devices make up Hajime. Unlike the vast majority of botnet studies (which comprise traditional desktop and laptop computers [24], [43]), Hajime encompasses a diverse set of devices. This section extends recent prior work on fingerprinting IoT botnet devices [8] by offering a unique perspective: the inherent separation of Hajime by CPU architecture. We also briefly describe Hajime composition using characteristics from remote fingerprinting services.

A. Architecture type

As noted, Hajime targets a wide range of architectures. Figure 8 presents the same data as in Figure 2, but broken down by architecture. This result shows a wide disparity in architecture types across bots. The `mipseb` architecture is by far the most infected, constituting 74.2% of all of the devices we measured in our four month active DHT scans. Collectively, the `arm` bots comprise 14.4% of all Hajime bots.

Figure 9a shows the composition of architecture types for each of the top 10 countries by overall number of bots before Hajime's deployment of the GPON exploit on May 8, and Figure 9b shows the composition including the GPON

arm5		arm6		arm7		mipseb		mipsel		No arch.	
Huawei HG658d	2.45%	MikroTik	0.12%	Huawei HG658d	1.03%	MikroTik	66.62%	MikroTik	5.60%	MikroTik	56.62%
Technicolor Modem	0.57%	—		Dahua Technology	0.92%	Intelbras 5000	0.17%	Hikivision	0.11%	Intelbras 5000	0.44%
Motorola Modem	0.51%	—		T+W Gateway	0.37%	D-Link Router	0.10%	DreamBox	0.11%	D-Link Router	0.21%
ZyXEL	0.38%	—		MikroTik	0.26%	Huawei	0.10%	D-Link Router	0.06%	Ubiquiti Networks	0.19%
Huawei Gateway	0.34%	—		Huawei Gateway	0.21%	Fiberhome	0.08%	HP Officejet 7610	0.03%	Huawei	0.19%
other	2.20%	—		other	1.19%	other	0.40%	other	0.17%	other	0.51%
unknown	93.56%	unknown	99.88%	unknown	96.02%	unknown	32.54%	unknown	93.91%	unknown	41.84%

TABLE III: Censys [14] device fingerprinting, broken down by architecture. (Active scans, 04/22/2018 – 04/28/2018.)

exploit and through the end of our data collection period. We find strong correlations between architecture type and country. Brazil, the most infected country by far, is made up predominantly of `mipseb` (note the broken y-axis). Yet, interestingly, it is not the case that all of the most-infected countries are predominantly `mipseb`; China is made up almost exclusively of `mipsel`, and the US has a majority of `arm5`.

The effects of the GPON exploit on architecture composition are also interesting. In particular, before the exploit, Mexico was historically the 9th most infected country, and was comprised nearly entirely of `arm` devices, whereas after the exploit's deployment, Mexico jumps to the third overall most infected country, and changes to a `mipseb` majority. The exploit also pushes Indonesia into the top 10, displacing Italy.

Note that if we only see a bot seeding or leeching a config file, not an architecture-specific payload, we are unable to determine the bot's architecture. Thus there is a small portion of bots for which the architecture is unknown.

These results have a number of profound implications. *First*, recall that different vulnerabilities target different architecture types (see Table I). Combined with the results in Figure 9, this indicates that a new vulnerability could result in an influx of bots from specific regions, due to global biases in manufacturer and device market shares. This represents a stark difference between IoT botnets and traditional botnets, which have more global homogeneity in device type.

Second, the variability in architecture type also has implications for the design of future honeypots. To capture a representative number of IoT infection attempts across a representative set of countries, one must construct a honeypot comprised of multiple architecture types (at least in emulation).

In summary, our results show a diverse set of architectures, distributed geographically in uneven ways, indicating a diverse array of vulnerabilities around the world.

B. Device Fingerprinting

Given the heterogeneous nature of IoT, we compare the DHT and uTP datasets with fingerprinting scans from Censys [14] to estimate the device composition of Hajime. Censys scans the entire Internet daily to provide a snapshot of running services over time. We collected every IP address that we could associate with a uTP key over a 7-day period from April 22-28, 2018. This resulted in 2,208,419 (IP, day) pairs. We then queried each individual day's Censys scan results with the list of IP addresses for that day. The daily results were combined, then partitioned by the architecture that each IP address seeded or leeched on the DHT each day[7].

Censys successfully scanned 468,073 IP addresses on the day each address was active on the DHT, and the day we were able to obtain a uTP key from the address. We therefore have high confidence that Censys scanned these devices while they were active under the influence of Hajime. Table III shows the device description information provided by Censys for these bots, separated by architecture. Percentages are shown relative to the total number of bots per architecture. The devices that we were able to fingerprint demonstrate that device composition differs between architectures. Some bots were only associated with seeding or leeching a config file on a given day. We do not know the underlying architecture for these devices, but report the Censys results under the "No arch." column in the table.

Table III shows that device makeup can vary significantly across different architecture types. That said, there are two common device types: home routers and MicroTik devices. As we discussed in Section V, the MicroTik devices can be explained by Hajime's inclusion of the Chimay-Red exploit.

We speculate that home routers are over-represented in Table III due to a bias in how Censys scans device types. Hajime does in fact run on many home routers (we discuss a vulnerability Hajime uses that specifically targets home routers in Section X). But many of the devices it targets are IoT devices that would likely be *in a NAT behind* a home router, and therefore share an IP address with the router. This, coupled with the many "unknowns" that Censys returned, show that accurate IoT device fingerprinting at scale remains an open, challenging problem.

To determine how Hajime's device composition changed as a result of the addition of Chimay-Red, we fingerprinted devices one day before (March 23, 2018) and one day after (March 25, 2018) the exploit's inclusion in Hajime. Again we took all IP addresses for which we obtained a uTP key, and ran those addresses through Censys. Chimay-Red was designed to target MikroTik devices. Before the exploit's release, Censys tagged 0.79% of devices across all architectures as MikroTik. The day after, that number rose to 80.29%. This stark change in the makeup of the Hajime botnet happened *overnight*. Chimay-Red's release further emphasizes the magnitude and speed with which a single event can alter the composition of an IoT botnet.

IX. Payload Updates

Updates to Hajime payloads significantly affect the botnet's size and composition. Here, we study the updates' dynamics.

[7] 34,637 (1.57%) of IP addresses were paired with multiple architectures in one day; we consider these addresses anomalous for those days and discard them.

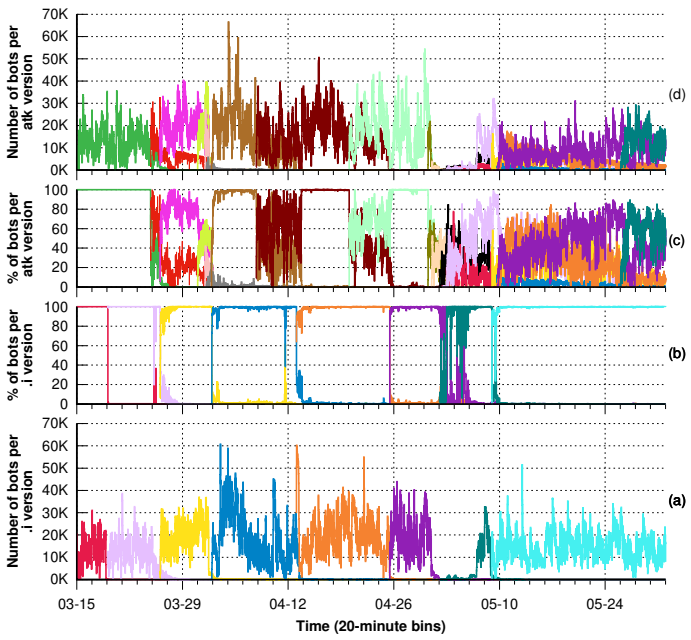


Fig. 10: Propagation of Hajime updates for `mipseb` architecture payloads. All four plots use the same x-axis time scale. (Active scans.)

A Hajime payload filename consists of the payload type (`.i` or `atk`), the target architecture (e.g., `mipseb`), and a Unix epoch timestamp. We speculate that the file's timestamp is the file's creation time and thus a lower bound on when any bot could have received the update. Since the Hajime algorithm for the payload's DHT key identifier involves a hash of the payload filename (including the timestamp), we can monitor the different payload versions that bots seed and leech.

We seek to understand how long it takes Hajime's P2P infrastructure to propagate an update after the botmaster has published that update. Figure 10 shows how `mipseb` updates spread through the botnet. Figure 10a shows the number of distinct bots seeding and leeching each version of the `mipseb .i`, in 20-minute intervals. Each colored line represents a different version. Note that the date interval begins at March 15, since we saw no updates until March 18. For the same versions, Figure 10b shows the number of distinct bots seeding and leeching that version of the `mipseb .i` divided by the total number of distinct bots for all versions of the `mipseb .i` during that hour. Figures 10d and 10c show the same data for `mipseb atk` versions, respectively. Note that the percentages within an interval may sum to more than 100% if we see the same bot for multiple versions within that same interval. We only discuss `mipseb` because it is the most common architecture, and the other architectures exhibit similar behavior.

The propagation of the `.i` updates is close to ideal: when a new version is released, most bots switch over to the new version very quickly, illustrated by each version quickly rising to 100% and remaining there until the next version is released. The reason that bots switch over to the new version over such a small time window is likely that bots download the config file once per day at 00:00 UTC. [8] This means that most bots will see a config file updated with a new module, and subsequently download that updated module at 00:00 the following day. This effectively synchronizes the payload updates.

The propagation of the `atk` updates looks markedly different. There is generally a sharp drop in the proportion of bots for the last version when a new version is released, but instead of falling to zero, a significant portion of bots continue to seed and leech the older version(s). Interestingly, when a new .i propagates, the stale `atk` versions are quickly cleaned up and removed almost entirely from the system. At this point the latest `atk` version finally achieves near 100% coverage, meaning the botmaster can quickly update the `atk` if they so wish.

Recall that the release of a `.i` also had the effect of stabilizing the highly variable birth and death rate as seen on April 2 in Figure 4. The difference between the spread of the `.i` module and the `atk` module is due to the differing nature of the two. When a new `.i` module is found, the running `.i` downloads it and begins its execution. It then immediately exits after starting the new `.i`. This effectively cleans up the old DHT state, and rejoins the DHT with a new key and port. However, this is not the case with the `atk`, as it will not restart the `.i`'s execution, flushing the old state.

Although there have not yet been any recorded attacks by Hajime, and the author claims (in the config file message) to be a white hat hacker securing devices, the possibility for a future attack remains. Given how quickly `.i` updates propagate to nearly all bots, Hajime demonstrates the potential to carry out a large-scale, highly coordinated attack.

Note the introduction of Chimay-Red on March 25, 2018, and the sharp increase in bots on that day. Although our reverse engineering efforts found that only `mipseb` was updated with the Chimay-Red exploit, every module except `.i.arm5` (refer to Figure 1) was also updated, suggesting that the botmaster is updating the other binaries with timestamps. Indeed, we found that many updates are minor additions to the logic or modifications to the timestamps necessary to check for new modules.

X. Tracking a Vulnerability over Time

In this section, we turn to a new dataset to study one of Hajime's infection strategies in particular: the use of the TR-064 vulnerability. We study this vulnerability for two reasons. First, it is a vulnerability that Hajime and Mirai have in common; our dataset captures both of them, and thus we are able to provide a direct comparison to their numbers and activity. Second, we are interested in understanding how (in)complete a view of the global Hajime network one obtains by observing just one vulnerability. Recall that Hajime uses a wide range of exploits across a wide range of architectures (Table I); if one were to run a honeypot with only one of these, it would risk biasing one's results. We compare our findings from the TR-064 vulnerability to our other results that involve exhaustively scanning *all* bots.

[8] We observe a small percentage of bots that seem to have incorrectly configured device clocks, which download at other times.

A. TR-064 vulnerability

In January 2017, the `mipseb` and `mipsel` `atk`s added an exploit for the TR-064 service as an additional propagation vector to that of brute forcing default Telnet credentials. TR-064 [13] is a deprecated technical report published by the Broadband Forum that specifies a method for configuring a home router from within the LAN.

The exploit leverages a shell injection vulnerability in the processing of a TR-064 request to set the device's NTP server. When processing the request, a vulnerable device will form an NTP client command-line using the attacker's unsanitized NTP server argument; this command is eventually interpreted by the shell. The shell injection takes on one of the following two forms:

```
`cd /tmp;tftp -lX -rX -g ADDRESS;chmod 777 X;./X`
```

```
`cd /tmp;wget http://ADDRESS/X;chmod 777 X;./X`
```

Each command effectively downloads and executes the file **X**, where **X** is the `.i`.

Consider what happens when this exploit is launched against a *non-vulnerable* device. Immune to the shell injection, the device treats the shell commands as an NTP server hostname, and thus performs a DNS lookup for the IP address. This involves sending a DNS query to the device's local resolver (or to a publicly available open resolver), which in turn recursively issues queries on the client device's behalf. Since the injection string lacks a valid top-level domain (TLD), local resolvers must go to a *root DNS server* to get an authoritative answer.

In other words, whenever an attacker targets a non-vulnerable device, the attacker triggers a DNS query to a root DNS server for the shell injection string, *which, in the case of Hajime, includes the attacker's IP address*. Because the DNS query ends in an invalid TLD (`./X‘`), it will be forwarded to (and authoritatively answered by) a root DNS server. This resulting "backscatter" effect [17] makes it possible to track unsuccessful TR-064 injection attempts by monitoring query data at a root DNS server. We describe such a dataset next.

B. DNS backscatter dataset

We collect samples of all queries to the D-root DNS root server. D-root comprises over 100 replicas in different geographic locations. We collect 20% of all queries to these replicas, corresponding to approximately 30,000 queries per second; pcaps are available for analysis within 36 hours. Due to sampling, our traces include roughly one fifth of one thirteenth of the queries that reach root name servers in general [9].

There are several challenges that we needed to address to analyze this dataset. Due to the non-vulnerable devices' processing of the injection string, 99.2% of the backscatter contains the injection string truncated to 35 characters. This causes 76.9% of the injections that `wget` to an IP address to have four complete octets, and the remainder to have only three. Since the injection string with `tftp` is longer, only 2.0% of these addresses have four complete octets, 81.9% have three, and the remainder have two.

[9]The precise ratio may vary due to imbalance between the thirteen DNS roots.

Validating that Hajime originated the backscatter is difficult not only due to this truncation, but also the historical nature of the data. Anecdotally, during our initial analysis in April 2017, we issued the `wget` command observed in the backscatter having complete IP addresses and port numbers, and, when successful, downloaded only `mipseb` and `mipsel` `.i` samples. We repeated this experiment in April 20, 2018 for a sample of roughly 42K backscatter queries form the prior day, opting to brute force a port number if it contained at least three digits. For this sample, there were only 90 candidate IP-port pairs, of which just three returned a response, all of which were `.i`s for the two `mips` architectures.

We must also determine how to classify the injection by actor, as other botnets have used this exploit; namely, Mirai deployed this same exploit roughly a month before Hajime. To that end, we reverse engineered the `mips` `atk`s and noted a consistent injection template of a `wget` or `tftp` to the infecting Hajime bot's IP address and ephemeral port (the `atk`s implement both a simple HTTP and TFTP server, and randomly select which service to use in the injection). Note that this port is different from the port used for uTP communication. In contrast, as Mirai uses a centralized C&C server, Mirai's injection traditionally uses the hostname of the C&C server. Thus, we classify an injection as originating from Hajime if the command string matches the `atk`'s template prefix and includes at least a partial IP address, and from Mirai (or one of its variants), if it uses a domain name. The exception to this rule is that we classify as "Mirai" those injections with a complete IP address and implicit default port (e.g., `wget http://1.2.3.4/`).

It is possible that only certain non-vulnerable devices generate backscatter, and that other phenomena may be at play, such as a non-vulnerable device generating multiple "echo" backscatter queries for a single injection attempt. Creating a census of all non-vulnerable devices and reverse engineering their behavior with regard to the NTP server configuration is outside the scope of this work.

Even despite these limitations, the data provides remarkable insight into the rollout and deployment of the exploit. With the TR-064 backscatter data, we explore the rate of infection attempts, estimates of botnet size, and breakdowns of both of these characteristics by country. As we do not have uTP keys for this dataset, *for this analysis, we equate a Hajime bot with a unique IP address*. When displaying counts of injections, we consider all injections regardless of DNS query truncation; when displaying statistics on IP addresses, we consider only those that have three or four octets, as country-level geolocation for those IP addresses with three octets is not influenced by the unknown last octet.

C. Injection attempts over time

We first compare Hajime and Mirai's use of the exploit. Figure 11 shows the number of injections from Hajime and Mirai, binned in 20-minute intervals, and annotated with the timeline of Hajime updates as recorded by a Rapidity researcher [33]. In total, we observed 125,210,696 injections: 96.6% from Hajime, 3.3% from Mirai, and 0.1% lacking enough information to classify. We also observe that Mirai deployed the exploit on November 26, 2016; the C&C server

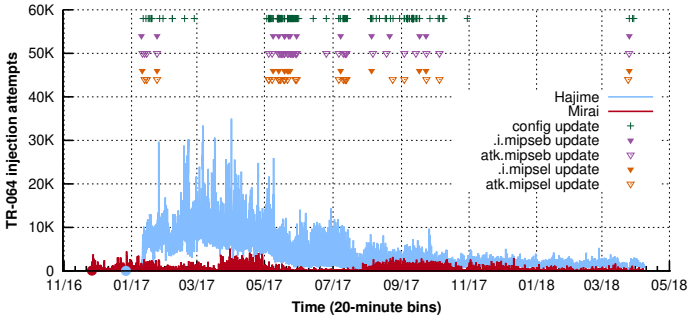


Fig. 11: TR-064 injection attempts for Hajime and Mirai. (DNS backscatter, 11/26/2016 – 04/08/2018.)

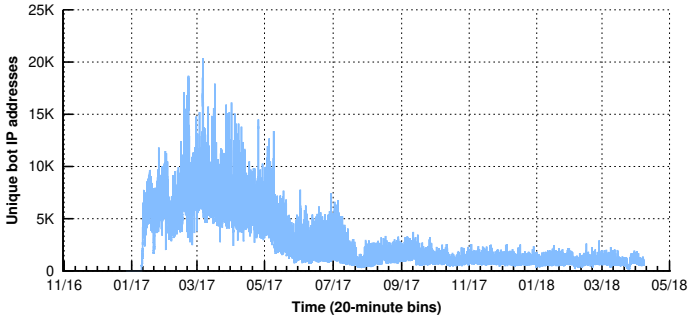


Fig. 12: Unique Hajime bot IP addresses. (DNS backscatter, 12/27/2016 – 04/08/2018.)

for these initial injections is a complete IP address and geolocated to Russia. In contrast Hajime deployed the exploit a month later, on December 27, with an address geolocated to the Netherlands. Coupled with the fact that Hajime blacklists address blocks from the Netherlands (Section III-A1) leads us to speculate that this is where Hajime's operations originate.

In Figure 12, we show the count of unique Hajime bot IP addresses in each 20-minute interval. Overall, we observe 4,604,802 unique Hajime bot IP addresses. For the first two weeks, both the number of Hajime bots and the number of Hajime injection attempts in a given 20-minute bin stay within the single digits, suggestive that the botmaster was testing the exploit. On January 11, 04:45 UTC, the botmaster updated the `mipseb atk`. In the 2-hours prior to the update there were an average of 2,275 unique addresses per 20-minute bin; in the 2-hours after, this number jumps to 4,031. This doubling of number of bots immediately after the exploit's deployment mimics the dynamics for Chimay-Red observed in Figure 2.

Using the number of unique IP addresses as an estimate of botnet size, we see from Figure 12 that Hajime generally stays in a band of 5-15K bots, before tapering in May. The difference in the size estimates based on backscatter compared to those based on active DHT measurements in Figure 2 can be attributed to two factors. First, only the `mipseb` and `mipsel` `atk`s deployed the TR-064 exploit, and thus the backscatter is always an underestimate of botnet size. Second, in the time span between the TR-064 and Chimay-Red deployments, Hajime pushed additional access vectors to the `arm` platforms (see Table I), which would recruit additional bots.

The brief spikes in number of bots in late February and again in March do not coincide with the update history, and are likely either a network effect, or a misclassification of the backscatter as Hajime for those intervals. From May through July there are a series of updates which appear to disable, at least in part, the TR-064 scanning. Security analysts [29] note that Hajime's scanning for this vulnerability stopped in July; we also ran a simple honeypot on port 7547 from late January 2018 to present for the TR-064 vulnerability and have not observed any Hajime injection attempts. Thus, the injections after July are potentially the result of some other phenomenon.

Finally, the shape of the number of Hajime IP addresses in Figure 12 mimics that of injections in Figure 11, giving additional assurance to our classification method; in particular, if IP addresses in the backscatter were C&C servers rather than Hajime bots, Figure 12 would have a smaller range and appear compressed in comparison.

D. Bot Location

We conclude this section by exploring the distribution of bot IP addresses and TR-064 injection attempts based on geolocation of the country for each bot IP address.

Table IV shows the top 10 countries based on the number of unique bot IP addresses geolocated to that country during the TR-064 exploit deployment. Iran dominates with 29% of all bot IP addresses, and the top 10 countries account for 81.5% of all addresses. To understand whether the relative ranking of these distributions changes over time, we show in Figure 13 the count of bot IP addresses in these countries, in 20-minute intervals. We observe that the ranking remains consistent over the course of the deployment.

We also compare these top 10 countries to the architectural breakdown of the top 10 countries based on the number of unique uTP keys in Figure 9a (before the GPON exploit). We note that seven of the countries from these two sets overlap, all of which have bot majorities for the `mips` architectures. Two of the three countries that appear in Figure 9a but not Table IV — United States and Mexico — have bot populations predominately composed of `arm`. This further validates that the botmaster only deployed the TR-064 exploit to the `mipsel` and `mipseb atk` modules. It is interesting that Brazil, which is overwhelmingly infected by `mipseb` bots, dominates bot counts based on uTP keys, but is only seventh when ranked according to the backscatter IP addresses. This may be a product of the different time frames for the DHT and DNS studies.

Finally, we determine the top 10 countries based on the number of injection attempts originating from bot IP addresses in that country, indicating the top ten countries overall in Table V and the number of injections from these countries over time in Figure 14. Interestingly, eight of the countries overlap between Table IV and Table V, with Brazil and Thailand swapping out for South Korea and Slovakia. This is likely a product of the different country behaviors with respect to IP address reassignment. For instance, if we refer back to Table II, we see that South Korea has the largest fraction of uTP keys to IP addresses (indicating multiple bots NATted behind a single IP), corroborating why this country appears in Table V but not Table IV.

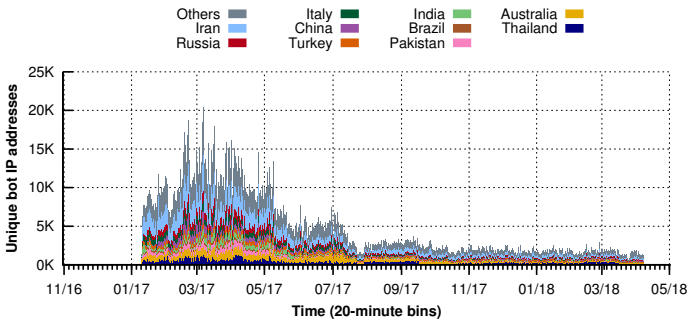


Fig. 13: Unique bot IP addresses for the top-10 countries with the most overall unique bot IP addresses (DNS backscatter, 12/27/2016 – 04/08/2018.)

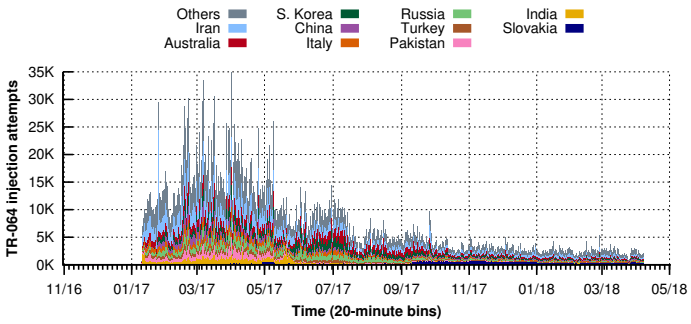


Fig. 14: Injection attempts for the top-10 countries with the most overall Hajime injection attempts (DNS backscatter, 12/27/2016 – 04/08/2018.)

Country	Unique Bot IP addresses	% of Total
Iran	1,336,385	29.0%
Russia	443,428	9.6%
Italy	427,203	9.3%
China	284,561	6.2%
Turkey	258,232	5.6%
India	232,277	5.0%
Brazil	228,503	5.0%
Pakistan	209,741	4.6%
Australia	177,894	3.9%
Thailand	155,726	3.4%
Others	850,852	18.5%
Total	4,604,802	100%

TABLE IV: Top 10 countries, sorted by the number of unique Hajime bot IP addresses in that country. (DNS backscatter, 12/27/2016 – 04/08/2018)

Country	Injection Attempts	% of Total
Iran	22,235,366	18.4%
Australia	11,461,797	9.5%
South Korea	8,844,582	7.3%
China	6,929,353	5.7%
Italy	6,434,260	5.3%
Russia	6,233,428	5.2%
Turkey	6,053,726	5.0%
Pakistan	4,882,090	4.0%
India	4,845,977	4.0%
Slovakia	4,762,013	3.9%
Others	38,309,187	31.7%
Total	120,991,779	100%

TABLE V: Top 10 countries, sorted by the number of Hajime injection attempts from that country (DNS backscatter, 12/27/2018 – 04/08/2018)

XI. Conclusion

Hajime is a new, large IoT botnet that is distinguished by its use of an existing DHT for command-and-control. In this paper, we have analyzed the spread and operation of Hajime using both passive and active measurements. These techniques enable us to gain insight into its macroscopic behavior at a scale that is not easily replicable with existing methods such as honeypots. Moreover, our active scans allow us to disambiguate IP addresses from bots.

Our results throughout this paper have demonstrated that the Hajime botnet has grown to be incredibly large, geographically dispersed, and resilient. Our measurement techniques lend insight not only into the Hajime botnet in particular, but into the makeup of this new, early wave of IoT botnets. For instance, Hajime makes it possible for us to measure the architecture types of all active bots. The diversity of architecture type by country, and the diversity in the classes of vulnerabilities that architectures are susceptible to, are what ultimately lead to the large discrepancies in the number of bots on a per-country basis.

Guided by these findings, we conclude with a thought experiment: what would it take to bring down Hajime's C&C infrastructure? Ideally, we would be able to patch all vulnerable IoT devices, but given the diversity of equipment, architecture, and attacks, this is extremely challenging. Another approach would be to focus defensive efforts on the BitTorrent DHT itself. It is straightforward to calculate the DHT keys Hajime bots use; one approach would be to patch BitTorrent DHT peers to blacklist those keys. Unfortunately, this would make it trivial for Hajime to DoS key values in the BitTorrent DHT simply by instructing its peers to use it. Alternatively, Kademlia allows peers to choose their own IDs: one could run many virtual peers to consume all of the IDs corresponding to Hajime keys, effectively launching an eclipse attack [40] against all Hajime bots. Another approach would be to use measurement techniques like ours to collect the set of IP addresses where bots are located, and to block those; unfortunately, this may block benign users behind a NAT, as well.

Unfortunately, none of these approaches satisfactorily halts Hajime's C&C without harming the quality of BitTorrent's DHT. This leads us to conclude that C&C for the current generation of IoT botnets is highly resilient. We hope that these findings serve as a call-to-arms to improve patching of IoT devices and to filter botnet traffic closer to the bots. To assist in such efforts, we have made our code and data publicly available:

`https://iot.cs.umd.edu`

Acknowledgments

We thank Bobby Bhattacharjee, Neil Spring, and Ramakrishna Padmanabhan for their help and feedback on early

versions of this work. We also thank Ioannis Profetis for clarifying details of Hajime's uTP protocol. Finally, we thank the anonymous reviewers for their helpful comments. This work was supported in part by NSF grants CNS-1409249, CNS-1564143, and CNS-1816802.

REFERENCES

[1] "Chimay-Red source code," Online: https://github.com/BigNerd95/Chimay-Red.

[2] "Linux.wifatch," Online: https://github.com/tbodt/linux.wifatch.

[3] "Vault 7: CIA Hacking Tools Revealed," Online: https://wikileaks.org/ciav7p1/.

[4] Akamai, "Akamai's State of the Internet / Security, Q3 2016 Report," Online: https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/q3-2016-state-of-the-internet-security-report.pdf, 2016.

[5] A. N. Andew Loewenstern, *DHT Protocol*, Online: http://www.bittorrent.org/beps/bep_0005.html, 2008.

[6] Anna-senpai, "Mirai source code," Online: https://github.com/jgamblin/Mirai-Source-Code/.

[7] Anonymous, "Internet Census 2012," Online: http://census2012.sourceforge.net/paper.html.

[8] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the Mirai botnet," in *USENIX Security Symposium*, 2017.

[9] M. Bailey, E. Cooke, F. Jahanium, Y. Xu, and M. Karir, "A survey of botnet technology and defenses," in *Cybersecurity Applications Technology (CATCH)*, 2009.

[10] D. Brown, "Resilient botnet command and control," in *DEF CON*, 2010.

[11] C. Y. Cho, J. Caballero, C. Grier, V. Paxson, and D. Song, "Insights from the inside: A view of botnet management from infiltration," in *USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET)*, 2008.

[12] C. Dixon, T. Anderson, and A. Krishnamurthy, "Phalanx: Withstanding multimillion-node botnets," in *Symposium on Networked Systems Design and Implementation (NSDI)*, 2008.

[13] DSLHome-Technical Working Group, "DSL Forum TR-064: LAN-Side DSL CPE Configuration," Online: https://www.broadband-forum.org/technical/download/TR-064.pdf, 2004.

[14] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman, "A search engine backed by Internet-wide scanning," in *ACM Conference on Computer and Communications Security (CCS)*, 2015.

[15] Dyn, "Dyn analysis summary of Friday October 21 attack," Online: https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/, 2016.

[16] S. Edwards and I. Profetis, "Hajime: Analysis of a decentralized internet worm for IoT devices," Online: https://security.rapiditynetworks.com/publications/2016-10-16/hajime.pdf, 2016.

[17] K. Fukuda and J. Heidemann, "Detecting malicious activity with DNS backscatter," in *ACM Internet Measurement Conference (IMC)*, 2015.

[18] W. Grange, "Hajime worm battles Mirai for control of the Internet of Things," Online: https://www.symantec.com/connect/blogs/hajime-worm-battles-mirai-control-internet-things, 2016.

[19] J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, and D. Dagon, "Peer-to-peer botnets: Overview and case study," in *Workshop on Hot Topics in Understanding Bots (HotBots)*, 2007.

[20] S. Grover, M. S. Park, S. Sundaresan, S. Burnett, H. Kim, B. Ravi, and N. Feamster, "Peeking behind the NAT: An empirical study of home networks," in *ACM Internet Measurement Conference (IMC)*, 2013.

[21] K. Hayashi, Online: https://www.symantec.com/connect/blogs/iot-worm-used-mine-cryptocurrency, 2014.

[22] T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. Freiling, "Measurements and mitigation of peer-to-peer-based botnets: A case study on storm worm," in *USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET)*, 2008.

[23] B. B. Kang, E. Chan-Tin, C. P. Lee, J. Tyra, H. J. Kang, C. Nunnery, Z. Wadler, G. Sinclair, N. Hopper, D. Dagon, and Y. Kim, "Towards complete node enumeration in a peer-to-peer botnet," in *International Symposium on Information, Computer, and Communications Security (ASIACCS)*, 2009.

[24] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage, "Spamalytics: An empirical analysis of spam marketing conversion," in *ACM Conference on Computer and Communications Security (CCS)*, 2008.

[25] C. Kanich, K. Levchenko, B. Enright, G. M. Voelker, and S. Savage, "The Heisenbot uncertainty problem: Challenges in separating bots from chaff," in *USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET)*, 2008.

[26] MalwareMustDie, "MMD-0055-2016-Linux/PnScan; ELF Worm That Still Circles Around," Online: http://blog.malwaremustdie.org/2016/08/mmd-0056-2016-linuxmirai-just.html, 2016.

[27] P. Maymounkov and D. Mazières, "Kademlia: A peer-to-peer information system based on the xor metric," in *Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.

[28] J. C. Moritz Wanring, "Kadnode source code," Online: https://github.com/mwarning/KadNode.

[29] M. Negishi, "Observation of hajime botnet," Online: https://sect.iij.ad.jp/d/2017/09/293589.html, 2017.

[30] A. Norberg, *uTorrent tranposrt Protocol*, Online: http://www.bittorrent.org/beps/bep_0029.html, 2009.

[31] R. Padmanabhan, A. Dhamdhere, E. Aben, kc claffy, and N. Spring, "Reasons dynamic addresses change," in *ACM Internet Measurement Conference (IMC)*, 2016.

[32] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu, "Portcullis: Protecting connection setup from denial-of-capability attacks," in *ACM SIGCOMM*, 2007.

[33] I. Profetis, "hajime_hashes repo," Online: https://github.com/Psychotropos/hajime_hashes, 2017.

[34] Radware ERT Threat Advisory, "Hajime – friend or foe?" Online: https://security.radware.com/ddos-threats-attacks/hajime-iot-botnet/, 2017.

[35] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "My botnet is bigger than yours (maybe, better than yours): Why size estimates remain challenging," in *Workshop on Hot Topics in Understanding Bots (HotBots)*, 2007.

[36] P. Richter, F. Wohlfart, N. Vallina-Rodriguez, M. Allman, R. Bush, A. Feldmann, C. Kreibich, N. Weaver, and V. Paxson, "A multi-perspective analysis of carrier-grade NAT deployment," in *ACM Internet Measurement Conference (IMC)*, 2016.

[37] E. Ronen, A. Shamir, A.-O. Weingarten, and C. O'Flynn, "IoT goes nuclear: Creating a ZigBee chain reaction," in *IEEE Symposium on Security and Privacy*, 2017.

[38] C. Rossow, D. Andriesse, T. Werner, B. Stone-Gross, D. Plohman, C. J. Dietrich, and H. Bos, "SoK: P2PWNED - Modeling and evaluating the resilience of peer-to-peer botnets," in *IEEE Symposium on Security and Privacy*, 2013.

[39] R. P. F. Santos, "Arris passwod of the day generate," Online: https://www.borfast.com/projects/arris-password-of-the-day-generator/.

[40] A. Singh, M. Castro, P. Druschel, and A. Rowstron, "Defending against eclipse attacks on overlay networks," in *Proc. of ACM SIGOPS European Workshop*, 2004.

[41] S. Staniford, V. Paxson, and N. Weaver, "How to 0wn the Internet in your spare time," in *USENIX Security Symposium*, 2002.

[42] B. Stock, J. Göbel, M. Engelberth, F. C. Freiling, and T. Holz, "Walowdac – Analysis of a peer-to-peer botnet," in *European Conference on Computer Network Defense (EC2ND)*, 2009.

[43] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your botnet is my botnet:

Analysis of a botnet takeover," in *ACM Conference on Computer and Communications Security (CCS)*, 2009.

[44] A. Tellez, "Bashlite," Online: https://github.com/anthonygtellez/BASHLITE.

[45] J. van der Wiel, V. Diaz, Y. Namestnikov, and K. Zykov, "Hajime, the mysterious evolving botnet," Online: https://securelist.com/hajime-the-mysterious-evolving-botnet/78160/, 2017.

[46] VPNMentor, "Critical rce vulnerability found in over a million gpon home routers," Online: https://www.vpnmentor.com/blog/critical-vulnerability-gpon-router/.

[47] X. Yang, D. Wetherall, and T. Anderson, "TVA: A DoS-limiting network architecture," in *ACM SIGCOMM*, 2008.

[48] L. Zhuang, J. Dunagan, D. Simon, H. J. Wang, I. Osipkov, G. Hulten, and J. Tygar, "Characterizing botnets from email spam records," in *USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET)*, 2008.