Before Toasters Rise Up: A View Into the Emerging IoT Threat Landscape

Pierre-Antoine Vervier and Yun Shen

Symantec Research Labs
{pierre-antoine_vervier,yun_shen}@symantec.com

Abstract. The insecurity of smart Internet-connected or so-called "IoT" devices has become more concerning than ever. The existence of botnets exploiting vulnerable, often poorly secured and configured Internet-facing devices has been known for many years. However, the outbreak of several high-profile DDoS attacks sourced by massive IoT botnets, such as Mirai, in late 2016 served as an indication of the potential devastating impact that these vulnerable devices represent. Since then, the volume and sophistication of attacks targeting IoT devices have grown steeply and new botnets now emerge every couple of months. Although a lot of research is being carried out to study new spurs of attacks and malware, we still lack a comprehensive overview of the current state of the IoT thread landscape. In this paper, we present the insights gained from operating low- and high-interaction IoT honeypots for a period of six months. Namely, we see that the diversity and sophistication of IoT botnets are both growing. While Mirai is still a dominating actor, it now has to coexist with other botnets such as Hajime and IoT Reaper. Cybercriminals also appear to be packing their botnets with more and more software vulnerability exploits targeting specific devices to increase their infection rate and win the battle against the other competing botnets. Finally, while the IoT malware ecosystem is currently not as sophisticated as the traditional one, it is rapidly catching up. We thus believe that the security community has the opportunity to learn from passed experience and act proactively upon this emerging threat.

1 Introduction

Over the last few years, security, or lack thereof, in the world of smart Internet-connected (or Internet of Things, IoT) devices has raised a lot of attention and concerns. In late 2016, several massive and high-profile DDoS attacks originated from a botnet of compromised devices, such as IP cameras and home routers, have taken part of the Internet down [7]. Although it was known for many years that there exists a lot of poorly configured Internet-facing IoT devices with default credentials or outdated firmware making them vulnerable to full device takeover, the high-profile attacks really revealed the destructive potential an army of such devices represent when used in a coordinated fashion.

There is a number of existing research work [7, 14, 18, 23, 30, 31] that looked into these increasing threats.The most notable work is probably Antonakakis *et*

al.'s forensic analysis of the Mirai botnet in 2017 [7]. Indeed, this study provides a very detailed description of the operations and evolution of the infamous botnet over a period of about one year. While this work provides an unprecedented understanding into this major IoT threat, we have seen that Mirai and its close variants only account for a limited set of IoT botnets. Additionally, Cozzi *et al.* [14] studied Linux malware but focused on their system-level behaviour. Others have proposed techniques to build honeypots to study IoT threats but these suffer from intrinsic limitations. For example, IoTCandyJar [23] requires active scanning of real IoT devices and replay parts of real attacks against these devices to build realistic models of real-device interactions. Siphon [18] relies on real devices to build high-interaction honeypots but lacks a proper instrumentation mechanism and suffer from scalability issues. Finally, IoTPot [30] by Pa *et al.* combine low-interaction honeypots with sandbox-based high-interaction honeypots but limit themselves to monitor *telnet*-based attacks. The analysis of IoT threats from these honeypot-based studies thus bears some limitations from their design. All this motivated us to carry out a global study of the IoT threat landscape. Our ultimate goal is to better answer the following questions. *What are the IoT threats we currently observe in the wild? What is the attackers' modus operandi to penetrate, infect and monetise IoT devices? How is the IoT threat landscape evolving?*

To help answer these questions, we designed an experimental environment specifically tailored for the study of the IoT threat landscape combining low- and high-interaction honeypots. We leverage embedded device firmware emulation techniques [11, 13] to build high-interaction honeypots and show that it enables us to overcome major limitations in previous deployments, such as the instrumentation of the honeypots, while assuring a highly accurate real device-like interaction with attackers. We then present the results of the analysis of six months of data collected from our honeypots. We take a look at the three main stages of IoT device compromise - intrusion, infection and monetisation - and present our findings. For example, we see that while the Mirai botnet and its variants appear to be dominating the IoT threat landscape, other botnets like IoT Reaper and Hajime are fighting to grow and compromise as many devices as possible. We also see that IoT botnets are very dynamic, with rapidly changing malware hosting infrastructure and malware polymorphism. They also evolve very quickly due to, for instance, the source code release of some botnets, which means that we have to continually monitor them to detect when their behaviour changes and adapt our mitigation strategy. Finally, we observe a worrying trend of more and more IoT botnets leveraging a myriad of software vulnerabilities in specific devices to compromise them.

It is important to mention that such a work is not meant to be an one-off study but should rather be repeated over time to closely monitor the evolution of the threat landscape, that is, track existing and new botnets so as to adapt our intrusion detection and infection mitigation strategies.

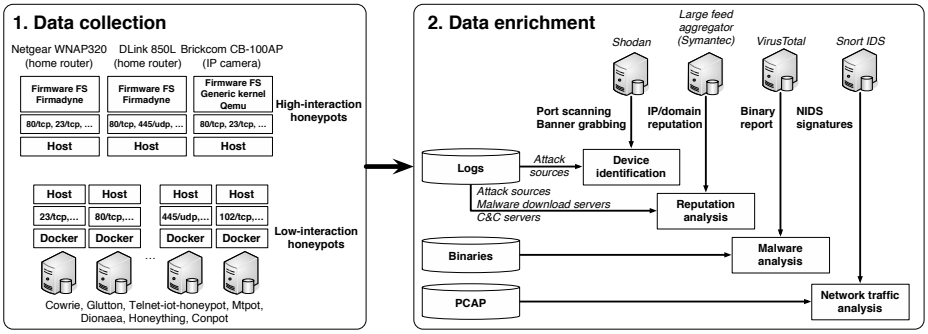


Fig. 1: Experimental environement for the study of the IoT threat landscape.

2 Data collection infrastructure

Studying the threat landscape in the world of smart Internet-connected (or IoT) devices is a very wide and complex task. IoT encompasses a lot of different types of devices, *e.g.*, smart televisions, surveillance systems, connected vehicles, water plant pumps, that can be deployed in a large variety of environments, *e.g.*, smart home and factories. Monitoring all these devices to detect potential compromise would be ideal but is of course infeasible. Indeed many of these devices, like industrial control systems, are deployed in very specialised environments and are also known to run on exotic and often proprietary hardware architectures and operating systems. These barriers thus makes it very hard to study the security of these devices. There is however a corpus of devices that run on commodity hardware and lightweight Linux-based operating systems. Such devices include, for instance, some home routers, IP cameras, smart televisions, DVRs and many more. These devices represent only a fraction of all the so-called "IoT" devices but, interestingly, they have been increasingly involved in cyber hazards over the last few of years due to flawed manufacturing and poor security configurations. Yet they are massively available in the consumer market.

Data collection Motivated by this, we thus decided to focus on the threats targeting these Linux-based IoT devices in this work. We deployed a set of honeypots mimicking various functionalities of some devices in order to observe three aspects of IoT attacks. That is, (i) the *reconnaissance* or *intrusion phase*, where attackers attempt to penetrate the defences of a device. (ii) The *infection phase*, where attackers usually take full control over the device and prepare it for whatever it is supposed to be used. (iii) Finally, the *monetisation phase* starts when the attackers use the compromised device for other nefarious purposes, such as infecting other devices, launching DDoS attacks, etc.

Figure 1 depicts the data collection and enrichment infrastructure we designed, deployed and have been operating since August 2017. The *data collection*

part consists of seven different open-source low-interaction honeypots[1]. Each honeypot is running inside a docker container to isolate it from the host and easily manage how network traffic flows between the host and the honeypot-emulated services. The main advantage of these low-interaction honeypots is that they are very straightforward to deploy thus allowing to collect IoT-related threat data very quickly. These honeypots aim at tricking attackers into infecting them by offering a very basic interaction for various services/applications, such as a *telnet* remote management interface, a FTP server or an embedded web interface. Since the interaction is purposely generic, *i.e.*, independent of a specific device, and completely hard-coded into the honeypot, attackers can take advantage of this to detect them by performing some specific checks. These honeypots are also unable to observe the *monetisation* phase as their functionalities do not enable them to get compromised.

To overcome the limitations of the low-interaction honeypots, we decided to explore the design and deployment of high-interaction IoT honeypots. As described in Section 4, different techniques have already been proposed to build a high-interaction IoT honeypot. Having considered the different previously proposed techniques, we realised that none of them provided the required amount of flexibility, scalability and ease of deployment. To this end, we leveraged two different techniques to build our emulated high-interaction honeypots. The first technique we used is an open-source firmware emulation framework called Firmadyne [11], which enables emulation of Linux-based systems by extracting the operating system from firmware images and running it with a generic kernel inside the `QEMU` virtualiser. It enables us to emulate the network-facing services provided by the devices, such as a *telnet* service, a web server, etc. However, Firmadyne requires the whole operating system (except the kernel) to be embedded in the firmware images for the emulation to work. Moreover, many device operating systems appear to be tightly bound to their hardware architecture, preventing the system from being successfully emulated when, for instance, the system seeks access to specific hard-coded memory addresses. This limitation also appears to affect certain types of devices, *e.g.*, IP cameras, more than others, *e.g.*, home routers. We thus decided to leverage another technique borrowed from [13], which consists of extracting the file system from firmware images and running the specific services we are interested in, such as a web server, inside a `chroot` environment on a QEMU-virtualised generic operating system of the same architecture as the real device.

We built one high-interaction honeypot for the Netgear WNAP320 (home router) and the DLink 850L (home router) using Firmadyne and one for the

[1] Glutton: https://github.com/mushorg/glutton
Cowrie: https://github.com/micheloosterhof/cowrie
Telnet-IoT-honeypot: https://github.com/Phype/telnet-iot-honeypot
MTPot: https://github.com/Cymmetria/MTPot
Honeything: https://github.com/omererdem/honeything
Dionaea: https://github.com/DinoTools/dionaea
Conpot: https://github.com/mushorg/conpot

Brickcom CB-100AP (IP camera) using the "chroot" technique. Since these honeypots run on emulated firmware images, we instrument them by reseting them to their original, clean state every hour.

We thus operate a total of 10 different honeypots - seven low-interaction ones and three high-interaction ones - offering a total of 15 different services on 26 ports. We also collect all incoming and outgoing traffic (the outgoing traffic from high-interaction honeypots is blocked on ports known to be used for scanning and rate-limited otherwise so that they do not involuntarily attack or scan other real devices on the Internet). Each honeypot is deployed on two different network infrastructures, namely a large cloud infrastructure that publishes its cloud-reserved IP address ranges and a tier-3 ISP cloud and hosting infrastructure. Finally, the honeypots are deployment over a set of 76 IP addresses located in six different countries and spanning two continents.

Data enrichment As depicted in Figure 1 the data enrichment part of our framework essentially consists of two tasks: (i) enrich the logs generated by and the files dropped on the honeypots, and (ii) process the network traffic captured at the honeypots to extract additional attack logs and files generated by attackers. More specifically, we extract information about devices our honeypots interact with from Shodan [1] and an IP and domain reputation feed. Furthermore, we retrieve binary reports about files dropped on the honeypots from VirusTotal [2] and run the Snort IDS with the subscription rules to help us label the collected network traffic.

3 Insights into the IoT threat landscape

In this section we present the results obtained by analysing the data collected from our honeypot deployment over a period of six months between August 2017 and February 2018. This data consists of (i) enriched logs produced by the different honeypots, (ii) raw network traffic and (iii) files dropped by attackers.

3.1 IoT device reconnaissance and intrusion

First, let us look at how attackers penetrate IoT devices in order to further compromise and monetise them. We have recorded a total of 37,360,767 connections to our honeypots from 1,586,530 unique IP addresses over the six month period. Additionally, our honeypots record peaks of up to 500K connections per day. It is noteworthy to mention that, in an effort to exclude as much as possible the basic port scanning traffic (*i.e.*, check if port is open/closed/filtered), we consider only fully-established TCP connections or at least two-packet long UDP connections. Comparably, previous IoT honeypot deployments reported about 70K *telnet* connections for IoTPot [30], 18M requests by IoTCandyJar [23] and 80K connections by the *telnet* honeypot used by Antonakakis *et al.* in their study of the Mirai botnet [7].

Looking at attack sources First, we take a look at countries that originated attacks against the honeypots. Surprisingly, more than one third of the attacks originated from Brazil. Note that none of the honeypots are deployed in this country and in the South American continent. Looking into more details at this phenomenon, it turns out that no less than 25% of attacks come from one of the biggest ISP in Brazil: Telefonica. Japan's third place is also surprising and, for the big part, attributable to the largest Japanese branch of the ISP NTT. Interestingly, Antonakakis *et al.* [7] observed a similar distribution in their study of Mirai with most bots concentrated in South America and East Asia. This suggests that the issues affecting these regions have yet to be resolved. Finally, China, Russia and the United States together account for about 20% of attacks.

Now looking at the distribution of device types attacking our honeypots, we see that networking devices, such as routers, DSL/cable modems, come first supposedly due to the fact that they are widespread and typically directly reachable from the Internet. After networking devices our honeypots were heavily hit by IP cameras, digital video recorders (DVRs) and alarm systems.

Finally, we extracted the IP-based reputation of attack sources from a large feed aggregator at the time these IP addresses connected to the honeypots. This reputation feed aggregates tens of blacklists describing different malicious activity, such as bot infections, spam, C&C server hosting, web-based attacks, etc. More than two thirds of attack sources were not known to any blacklist when we observed them for the first time. Additionally, about 15% of attacking IP addresses have been flagged as compromised and already part of a botnet. This last observation is consistent with the worm-like behaviour of IoT botnets where compromised devices are trying to self-replicate themselves.

Scanned and attacked services Looking at the distribution of connections per service given in Table 1, we can see that *telnet* dominates with more than 65% of connections, followed by *http* accounting for about 22% of connections. The remaining 13 decoy services represent a total of about 10% of connections. This distribution is of course skewed towards some services, such as *telnet* or *http*, which are provided by multiple of our honeypots while others, such as `modbus`, `bacnet` or `mqtt`, are emulated by only one honeypot. We thus provide the average number of connections per service, per day and per honeypot as a metric of the popularity (or attractiveness) of a service to attackers/scanners. With such a metric we can see that *http* ranks first, closely followed by *telnet* with an average of $5,712.97$ and $4,733.02$ daily connections respectively.

Since *http* and *telnet* are by far the most "attractive" and hit services at our honeypots and that these services are often provided by Internet-connected devices for remote administration, we decided to focus our investigation of IoT device intrusion mechanisms with these services.

Telnet access As documented in [7, 30] as well as in various blog posts [21], the intrusion mechanism of a large number of IoT botnets rely heavily on the exploitation of the *telnet*-based remote management interface often provided

Rank	Service	No. of connections	Avg. hit rate per day ↓	Rank	Service	No. of connections	Avg. hit rate per day ↓
1	http	8,469,122	5712.97	9	s7comm	8,623	7.10
2	telnet	25,334,377	4733.02	10	snmp	4,620	4.98
3	ssh	1,061,343	1019.26	11	mqtt	698	4.78
4	upnp	208,635	761.44	12	cwmp	10,011	4.51
5	smb	1,824,945	356.37	13	pptp	512	3.51
6	https	384,863	131.57	14	bacnet	1,193	1.04
7	modbus	14,408	15.54	15	ipmi	16	0.01
8	ftp	37,401	12.38				

Table 1: Breakdown of the number of connections to and average daily hit rate of the different decoy services offered by the honeypots.

by IoT devices. Given the usual lack of proper security management and poor manufacturing of devices, default or hardcoded [6] *telnet* login credentials can provide an easy, dictionary-based brute-force attack vector that usually leads attackers to take full control over the devices.

We have seen a total of 11,791,128 *telnet* connections (46.5% of the total 25M) where attackers successfully logged into the box. Furthermore, attackers needed on average three attempts to guess the correct username and password associated with the different honeypots. Note that our honeypots are all configured with default or easy to guess passwords as our goal is to capture as many attacks as possible. Finally, we have seen that attackers have tried to log in with a total of 4,095 unique usernames and passwords.

Vulnerability exploitation Lately, anecdotal evidence suggested that IoT botnets started leveraging not only *telnet* credentials brute-forcing but also exploiting very specific software vulnerabilities in IoT device firmware [8, 9]. To investigate this phenomenon, we leveraged our three high-interaction honeypots to determine how attackers have attempted to exploit them. Table 2 summarises the various vulnerabilities affecting these devices and the number of times these vulnerabilities were seen exploited by attackers[2].

We can see that both the DLink router and the Brickcom IP camera are affected by a lot of vulnerabilities, and than many of them - seven for the router and five for the camera - are being exploited in the wild. We can also see that the most exploited vulnerability for both the DLink router and the Brickcom IP camera leads to credentials disclosure, which appears to be what attackers are looking for the most. The other exploited vulnerabilities on the DLink router lead to remote command execution or full system takeover. As far as the Brickcom IP camera is concerned, apart from the XSS vulnerability, all other vulnerabilities are related to credentials/device information disclosure and all of

[2] We retained only vulnerabilities that can be exploited from by a remote attacker and that were related to services exposed by our honeypots.

Device	Vulnerability	Discl. date	No. of exploitations
DLink 850L (home router)	Stealing login and password [17]	Sep. 2017	258
	Remote Buffer Overflow in Cookie Header [25]	Jun. 2014	49
	Full Superuser access (RCE to Root) to the device [17]	Sep. 2017	13
	Remote Command Execution via WAN and LAN [29]	Aug. 2017	6
	Buffer overflows in authentication and HNAP functionalities [26]	Nov. 2015	3
	Remote code execution (CVE-2016-5681) [5]	Jun. 2016	2
	UPnP SOAP TelnetD Command Execution [24]	Sep. 2013	1
	Updating firmware in Recovery mode [17]	Sep. 2017	0
	XSS (CVE-2017-{14413,14414,14415,14416}) [19]	Sep. 2017	0
	Retrieving admin password (CVE-2017-{14417,14418}) [19]	Sep. 2017	0
	Nonce brute-forcing for DNS configuration - CVE-2017-14423 [19]	Sep. 2017	0
	Pre-Auth RCEs as root (L2) - CVE-2017-14429 [19]	Sep. 2017	0
Netgear WNAP320 (home router)	Arbitrary command execution (CVE-2016-1555) [4]	Jan. 2016	0
Brickcom CB-100AP-3456 (IP camera)	Remote Credentials and Settings Disclosure [28]	Jul. 2017	50
	Cross-site Request Forgery [27]	Jun. 2016	11
	Hard-coded Credentials [27]	Jun. 2016	6
	Cross-site Scripting [27]	Jun. 2016	6
	Insecure Direct Object Reference/Authentication Bypass [27]	Jun. 2016	6

Table 2: Software vulnerabilities affecting the high-interaction honeypot devices.

them are being exploited. Surprisingly, the only vulnerability affecting our Netgear router honeypot, which enables remote code execution and eventually a full device takeover, was never found to be exploited. In total 411 vulnerability exploitations have been observed across the three high-interaction honeypots over a period of four and a half month. While this number is still low compared to the number of *telnet* credentials cracking attempts, the fact that cybercriminals use so many and diverse vulnerability exploits (sometimes very recent) shows that they are putting a lot more care and sophistication into the building of their botnets. It also shows a real evolution from the first IoT botnets that were relying solely on *telnet* credentials brute-forcing. To the best of our knowledge this is the first time such a behaviour is reported with an assessment of actual vulnerability exploitations against IoT devices in the wild. It is also noteworthy that the disclosure date of the various exploited vulnerabilities vary a lot, from 2013 to the end of 2017. Moreover, the most exploited vulnerability for the DLink router and the Brickcom IP camera were both disclosed in the second half of 2017, only a few weeks before we started seeing them used against out honeypots.

To sum up, most of the time the goal of attackers is to get some privileged access to the device in order to proceed with the infection and later the monetisation. On the one hand, exploiting a software vulnerability on a specific device can facilitate the intrusion when devices are not properly patched and reduce the noise produced by the brute-forcing. However, it also requires more work and research from the botnet creator to find IoT device exploits.

A sneak peek at IoT Reaper One particular botnet appears to be heavily relying on software vulnerability exploitation to spread: *IoT Reaper* [10]. The

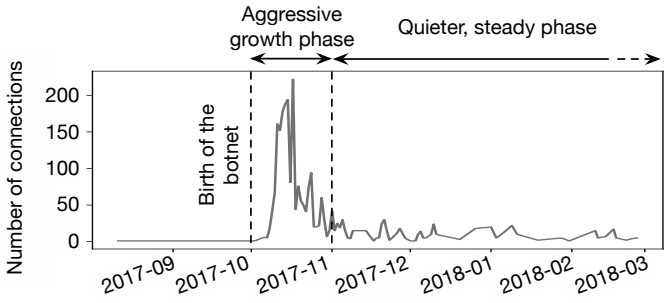


Fig. 2: Intrusion attempts from the IoT Reaper botnet.

botnet emerged in late 2017. Figure 2 depicts the number of intrusion attempts attributable to the IoT Reaper botnet against our honeypots. The figure shows that the botnet exhibited an aggressive peak of intrusions at the beginning of October 2017, when the botnet was born. After one month, it initiated a quieter phase, which could be due to (i) the botmaster(s) purposely slowing down the growth of the botnet after it reached a steady size or (ii) an attempt to remain under the radar after raising a lot of attention in its first, very active month. Interestingly, the motivations behind IoT Reaper's operators is still unknown.

Browsing attackers Telnet credentials brute-forcing and vulnerability exploitation are not the only behaviours we observe from our honeypots. In fact, given that our high-interaction honeypots mimics almost all functionalities of the real devices they emulate, we have witnessed some attackers "browsing" through the web interfaces of the two routers and the IP camera. Table 3 shows a snippet of some URLs requested from the different devices and the action triggered or information disclosed. Furthermore, we next attempt to determine if such "browsing" behaviour is generated by individuals actually visiting the pages or if it is generated by automated scanning tools. First, we look at the time elapsed between *http* requests from each client IP address and notice that 30% of clients issue requests with an average time gap of less than one second, which means these queries are thus likely generated by automated scanning tools. On the other hand, about 10% of clients issue requests with an average time gap of several 10's of seconds, which is more compatible with a real human "browsing" behaviour. Finally, one could argue that such a behaviour, when performed in an automated way, is likely to be part of some reconnaissance or device identification phase. However, in most cases, access to specific admin pages of the remote management web interfaces requires authentication, which assumes that attackers have already gotten access to valid credentials and presumably already know what device they are interacting with. Moreover, we typically observe that attackers accessing more than one page of a given web interface never request inexistent pages, showing that they are either browsing through the web interface or know exactly what pages are provided by the given device.

Device	URL	Action	No. of requests
DLink 850L (home router)	`GET /diagnostic.php`	Display previous diagnostic reports	99
	`GET /bsc_wlan.php`	Wireless network settings	14
	`GET /adv_wps.php`	Access WiFi protection setup	14
	`GET /tools.php`	Access administrator settings	4
	`GET /advanced.php`	Access advanced setup	3
	`GET /setup.php`	Access internet connection setup	3
	`GET /status.php`	Get device information	2
	`GET /st_wlan.php`	Get connected wireless client list	1
	`GET /st_routing.php`	Get device routing table	1
	`POST /routing_stat.php`	Issue routing-related command	1
Netgear WNAP320 (home router)	`GET /config.php?json=true`	Dump router configuration	127
	`GET /downloadFile.php?file=config`	Download config. file containing credentials	1
Brickcom CB-100AP-3456 (IP camera)	`GET /snapshot.jpg`	Get snapshot from IP camera video feed	85
	`POST /cgi-bin/camera.cgi`	Set camera settings	9
	`GET /cgi-bin/motiondetection.cgi?action=getMD&index=1`	Get motion detector settings	2
	`POST /cgi-bin/audiometer.cgi`	Set microphone sensitivity	1

Table 3: Snippet of URLs requested by "browsing" attackers.

3.2 IoT device infection

In the previous section, we described some of the IoT device reconnaissance and intrusion mechanisms we observed are used by cybercriminals to access and take control over IoT devices. This is usually the first step to a multi-stage attack eventually leading to compromised devices being used to perform other nefarious activities. In this section, we will discuss the second stage of an IoT device takeover where attackers prepare the device for its monetisation, usually by running some malicious code that (i) further tries to spreads itself by exploiting other devices and (ii) joins the C&C channel of an existing botnet.

In order to study the infection mechanisms against our IoT honeypots and given that, from our observations, *telnet* is by far the most prominent intrusion mechanism used by attackers, we extracted all commands issued by attackers from each *telnet* connection to our two *telnet*-enabled high-interaction honeypots, namely the Netgear router and the Brickcom IP camera, stripping away command arguments and credentials entered at the beginning of the sessions. Filtering out empty connections as well as connections where attackers didn't manage to successfully log into the box left us with a total of 169,804 out of 611,429 (27.77%) connections. Next, we removed short connections where the client issued a sequence of less than two commands, which is unlikely to implement a real compromise. This step left us with 93,099 (15.23% of the total) connections. Finally, we transformed each sequence of commands into an unordered set of commands, leading to a total of 8,167 *unique telnet* sessions. We then clustered these 8K sessions with the DBSCAN clustering algorithm using the Jaccard index to compute the similarity between each pair of *telnet* sessions. We obtained a total of 70 clusters. The clustering results are summarised in Table 4. Note that only *telnet* commands were used in the clustering and the malware families were added afterwards to illustrate the clusters.

Cluster ID	Size		Malware families
	No. of connections	No. of sessions	
A	45,599 (7.46%)	5624	Linux.Downloader, Linux.Mirai, Linux.Aidra, Linux.Kaiten, Linux.Gafgyt
B	10,259 (1.68%)	5	LinuxMirai, Linux.Masuta
C	7,146 (1.17%)	8	Linux.Mirai
D	6,820 (1.12%)	6	Linux.Mirai, Linux.Gafgyt
E	4,205 (0.69%)	3	Linux.Hajime
F	3,121 (0.51%)	7	Linux.Mirai
G	2,620 (0.43%)	21	Linux.Mirai, Linux.Gafgyt
H	2,212 (0.36%)	5	Linux.Mirai, Linux.Aidra
I	1,444 (0.24%)	4	Linux.Mirai, Linux.Aidra, Linux.Gafgyt
J	1,402 (0.23%)	5	Linux.Mirai

Table 4: Telnet session clustering results: top 10 clusters by size.

First of all we can see that the first cluster (A) is by far the biggest one with more than 45K `telnet` connections. It contains a lot of variety, with more than 5K unique sessions (*i.e.*, unique sets of commands). Cluster A can be linked to malware samples belonging to multiple families, namely Mirai, Aidra, Kaiten and Gafgyt, based on AV detections extracted from running the binaries dropped during these *telnet* sessions to VirusTotal. This observation, plus the low compactness of the cluster can be explained by the fact that the commands founds in the cluster are quite generic and common to a lot of malware families.

Cluster B contains about 10K *telnet* connections attributable to Mirai and Masuta. Masuta is a very recent variant of the Mirai botnet that emerged in late 2017. When digging further, we notice that the *telnet* command sequences leading to a Mirai sample and to a Masuta sample are almost identical, highlighting their common roots. In this case, the difference between the two threats resides in the dropped binaries.

Interestingly, cluster E appears to be related to the so-called "vigilante" (a.k.a. white hat) botnet *Hajime* [16]. Hajime is known to be a sophisticated, P2P-operated botnet that infects vulnerable IoT devices by brute-forcing their credentials. So far, it has not been linked to any specific type of attack, such as DDoS attacks. Interestingly, Edwards *et al.* described the Hajime infection process that would drop the malicious binary by issuing a series of `echo -ne "<hex-string>" >> <file>` commands over *telnet* in order to rebuild the binary and then execute it. This contrasts with most of IoT botnets, which drop binaries by downloading them from a remote host. However, from cluster E, it appears that Hajime has added the "download" functionality to its self-replication module. It now appears to first check whether it can download the binary and, upon failure, "echo load's" a custom dropper that downloads the main bot via HTTP.

We further looked at the number of commands issued by attackers visiting our low- and high-interaction honeypots. It is interesting to note that attackers, when getting into low-interaction honeypots are inclined to execute more unique

Rank	Malware family	No. of files ↓	Rank	Malware family	No. of files ↓
1	Linux.Mirai	1,609	8	Linux.Generic	7
2	Linux.Hajime	792	9	Linux.Remaiten	9
3	Linux.Gafgyt	464	10	Linux.Amnesia	5
4	Linux.Aidra	297	11	Linux.BitcoinMiner	1
5	Linux.Kaiten	154	12	Others	4
6	Linux.Download	30	13	Undetected	3
7	Linux.Masuta	10		**Total**	**3,385**

Table 5: Normalised AV detections of dropped binaries as given by VirusTotal.

commands during *telnet* sessions. We speculate that this phenomenon is due to the fact that low-interaction honeypots provide some default *telnet* session policies that return an empty result to the attackers. This default behaviour triggers the attackers to execute several other branches of their scripts to identify the architecture of the honeypot, alternative ways (e.g., `tftp`) to deliver binaries when `wget` failed, etc.

Looking at the *telnet* commands issued during the infection phase thus appears to provide a way to fingerprint attackers and attribute them to specific threats (or botnets). We believe that such a fine-grained profiling of attackers can greatly assist with the detection and investigation of IoT threats, for instance when writing IoCs.

Dropped files analysis Over the six months of operations our honeypots have collected 3,385 files that were dropped by attackers. For the sake of comparison, previous work on the study of IoT malware analysed 43 binaries in IoTPot [30] and 434 in the Mirai botnet study [7]. Attackers use various techniques to drop files to compromise devices. (i) The most common technique consists in downloading the binary, usually via HTTP or FTP from a remote host. (ii) The other technique we have witnessed is the "echo load" where attackers rebuild the binary in the *telnet* session by "echoing" hexadecimal strings into a file. So far, we have witnessed all malware families use method (i) and only a couple of them, namely Hajime and Gafgyt use (ii) in combination to (i).

We further obtained malware families of the more than 3K binaries we collected by querying the VirusTotal binary reports and normalising AV detection labels, as presented in Table 5. Note that 2,887 out of 3,385 (85.2%) files were not known to VirusTotal before we submitted them. From the perspective of our honeypots, Mirai represents the biggest set (47.5%) of binaries we see. Hajime and Gafgyt follows, with 24.4% and 13.7% respectively. We also observed a mix of old and new botnets, *e.g.*, Masuta emerged in late 2017, Mirai, Hajime and Remaiten appeared in 2016 and the first evidence of Gafgyt dates back to 2014. Interestingly, we can see that there appears to be one instance of a cryptocurrency mining malware that infected one of our honeypots.

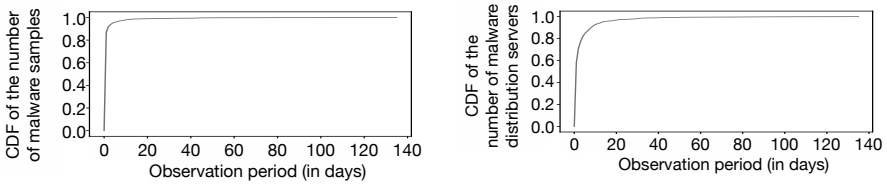
Finally, we looked at the observation window of individual malware sample hashes, which is plotted in Figure 3 (a). The short-lived trend here is very strong, with almost 90% of unique malware hashes seen during only one day. Additionally, four malware families - Mirai, Gafgyt, Kaiten and Hajime - have samples that are being observed for weeks and even months (with a maximum of four months and 13 days for Mirai). In case of Hajime, we witnessed only two binaries that have an observation window of several weeks. According to Edwards *et al.*'s analysis of the botnet [16], the two binaries appear to be Hajime's *stage2s* module, which corresponds to the final piece of the bot being run to fully compromise the device. Unlike the rest of Hajime's binaries we collected, these two binaries are also very likely packed, based on their Shannon entropy above 7.98 (out of 8). We speculate that malware authors decided to put more care into designing and obfuscating the *stage2s* binary, which is then observed for longer periods of time than the other first-stage binaries.

Malicious files download A total 2,837 binaries were downloaded from 832 different IP addresses hosted in 146 different ASes. Figure 3 (b) plots the observation window as seen from our honeypots. We can see that at least 90% of malware download servers appear to be short-lived, with a witnessed lifetime of less than five days. This is also corroborated by the fact that 60% of malware distributing IP addresses were never blacklisted throughout the six month data collection period. Such IP addresses thus appear to be used for a very short period of time to distribute IoT malware and then disposed of to move on to other IP addresses, so it is hard to rely on techniques like IP blacklisting to block them. It is also noteworthy that 40% of IP addresses are located in only five different ASes associated with large national ISPs providing hosting and cloud services. Note that 99.7% of URLs used by attackers to download files use raw IP addresses rather than domain names. Another interesting thing we observe is that attackers seem to use very limited number of IP adresses to host malicious binaries. We observe, on average, 12 malicious files being hosted at a single IP address and it is worth noting that one IP address was seen hosting up to 468 malicious binaries. We also noticed that the Brickcom IP Camera (avg. 863 downloads/day) is more active than the Netgear router honeypot (avg. 163 downloads/day) in terms of malware downloads. We speculate that this is due to the widely publicised article disclosing the camera's vulnerabilities with PoCs.

3.3 IoT device monetisation

The final stage of an IoT device compromise usually consists, for the attacker, in leveraging its full control of the device to perform other nefarious activity, such as infecting other devices to expand its botnet, launching DDoS attacks, etc.

We define the post-infection traffic as the traffic received and generated by a honeypot excluding the intrusion and infection. The post-infection traffic thus contains all potential C&C communications as well as other attacks or malicious activity performed from the compromised honeypots. It is important to note



(a) Malware samples

(b) Malware distribution servers

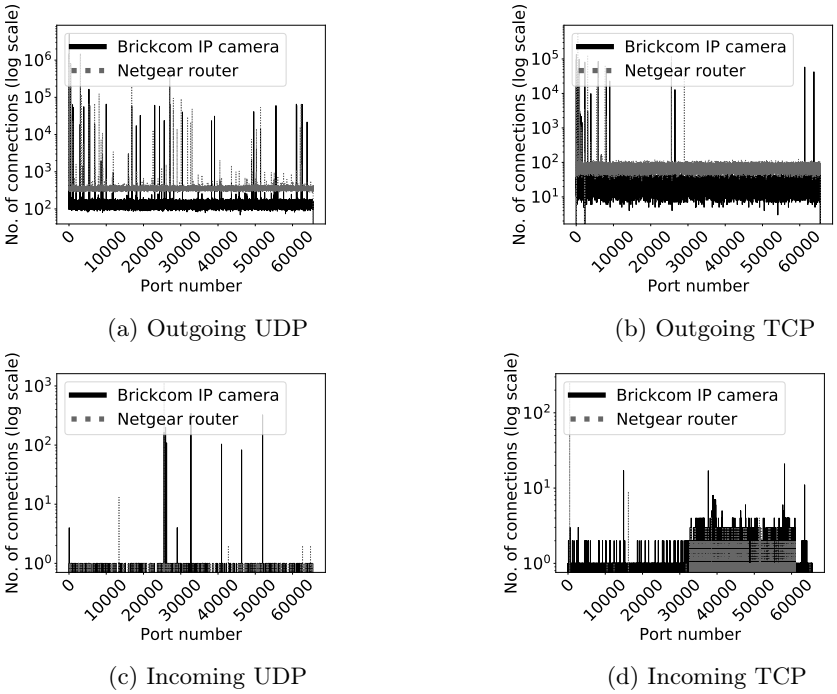
Fig. 3: CDF of the observation window of (a) malware samples and (b) malware distribution servers.

	Netgear router		Brickcom IP camera	
Rank	**Port**	**No. of connections**	**Port**	**No. of connections**
1	80/udp	2,604,683	80/udp	4,866,276
2	3074/udp	1,337,377	3074/udp	1,461,617
3	53/udp	1,004,940	53/udp	1,384,311
4	443/udp	764,040	22/udp	810,670
5	22/udp	630,907	443/udp	805,234
6	443/tcp	519,864	27015/udp	618,537
7	27015/udp	489,547	5355/udp	164,900
8	16837/udp	195,545	777/tcp	98,130
9	3074/tcp	129,435	34/tcp	96,703
10	8080/udp	123,446	53/tcp	95,490

Table 6: Top 10 ports in volume of post-infection traffic.

that the following post-infection traffic analysis is different from the previous research efforts due to the fact that we allow the malicious binaries in the real environment and only block or rate-limit outgoing traffic (see Section 2 for the detailed design information). Note in the following analysis we focus on the Netgear router and Brickcom IP camera high-interaction honeypots.

High-level overview We first carry out a high-level analysis of post-infection traffic by dividing it into TCP and UDP and incoming and outgoing. The goal is to identify if the high-interaction honeypots would experience different traffic volumes due to the fact that both devices, in the real world, have different computational capabilities. Our analytical results are shown in Figure 4. It is straightforward to notice that the router honeypot experienced more outgoing UDP and TCP traffic (see Figure 4 (a) and (b)) than the IP camera honeypot. Our interpretation of this behaviour is that the Netgear router has superior computational capabilities compared to an IP camera, hence the attackers do not need to "throttle" the performance of the payloads. It is also interesting to note that we observe outgoing traffic (on both TCP and UDP) on 65,335 different ports (see Figure 4), which contrasts with the incoming traffic observed on only



(a) Outgoing UDP

(b) Outgoing TCP

(c) Incoming UDP

(d) Incoming TCP

Fig. 4: Incoming and outgoing post-infection traffic per destination port (x-axis) recorded at the Netgear router and Brickcom IP camera honeypots.

14,560 and 5,320 ports for the Brickcom and Netgear honeypots respectively. The top 10 ports for outgoing post-infection traffic from both high-interaction honeypots are shown in Table 6. The first thing we observe is that the top 10 ports for outgoing traffic contributes to 30% of each high-interaction honeypot's total traffic. The second thing we observe is that the top three ports for outgoing traffic cover port scanning (`80/udp`), possible C&C communications (`3074/udp`) and DDoS attacks (`53/udp`). This traffic contributes to about 20% of each high-interaction honeypot's total traffic. We observed that even though there is a difference in the volume of traffic between the two high-interaction honeypots, however, in general, they show similar traffic patterns, *e.g.*, 80% of connections on ports `80/udp` and `53/udp` have less 10 packets and 80% of connections on port `3074/udp` have less than 20 packets.

Spiked traffic analysis After a high-level overview of the post-infection traffic, we turn our attention to the temporal analysis. As shown in Figure 5, there are four peaks in the traffic: peak ❶ is caused by a dramatic increase in TCP traffic while peak ❷, ❸ and ❹ are triggered by elevated UDP traffic. In order to identify the root causes of these peaks, we correlate the file downloads observed during the infection phase (see Section 3) to these traffic peaks in order to identify

Peak	Malware sample	Time	Top three ports per volume of traffic (port number / no. of connections)
❶ `tcp (attack)`	binary 1, 2, 3 and 4	2018-02-11 16:00 - 17:00	`3324/tcp` (117,267), `53/udp` (160), `6881/udp` (75)
❷ `udp (attack)`	binary 5 and 6	2018-02-20 02:00 - 03:00	`80/udp` (172,493), `3074/udp` (167,535), `100/udp` (54,772)
❸ `udp (scan)`	binary 7 and 8	2018-02-25 17:00 - 18:00	`21351/udp` (79), `64031/udp` (78), `3937/udp` (77)
❹ `udp (scan)`	binary 9, 10 and 11	2018-02-27 19:00 - 20:00	`6881/udp` (108), `35159/udp` (100), `54680/udp` (100)

Table 7: Analysis of binaries that triggered peaked post-infection traffic.

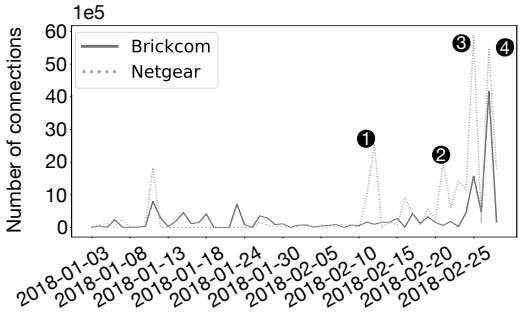


Fig. 5: Post-infection traffic analysis: traffic bursts.

the binaries that have contributed to such spikes. Note that due to the design philosophy of our high-interaction honeypots, we allow multiple binaries to run at the same in the same environment and it is hard to nail down traffic to a single binary. Taking this point into consideration, we slide an one-hour window along each day of post-infection traffic (*i.e.*, we obtain 24 traffic measurements per day) and rank them based upon the traffic volume. Once we have these ranked measurements in place, we identify the binaries that were downloaded during the top most active windows as the root cause of this peaked traffic. The results, shown in Table 7, are interesting. We noticed that the volume of traffic per port for the top three ports of peak ❸ and ❹ is way smaller than that of peak ❶ and ❷. For example, we observed 3,936,628 UDP connections between 19:00 - 20:00 in peak ❹, yet the traffic on the top three ports only cover around 0.01% of the total traffic observed during that hour. More interestingly, during the same hour, we observed outgoing traffic on 54,721 different ports. In contrast, in peak ❶, we only observed traffic on 628 different ports, yet one port covered 96% of the total traffic of that window. This observation motivates us to carry out a detail behavioural analysis.

Behavioural analysis In order to obtain a better understanding of the underlying communications behind the post-infection traffic, we attempted to classify it into three categories: (i) scanning traffic, (ii) attacking traffic and (iii) C&C traffic. First of all, we focus on days where we observe more than 10K connections, for TCP and UDP separately. On each day, we identify the largest hourly

Honeypot	Attack (No./avg. ports/avg. binaries)		Scan (No./avg. ports/avg. binaries)		Mixed (No./avg. ports/avg. binaries)	
	udp	tcp	udp	tcp	udp	tcp
Netgear router	**16 / 1,910 / 3**	10 / 308 / 54	**7 / 60,020 / 3**	6 / 53,573 / 2	**2 / 29,307 / 3**	0 / 0 / 0
Brickcom IP camera	**12 / 1,371 / 4**	12 / 11 / 8	**8 / 58,424 / 5**	7 / 39,820 / 5	**3 / 37,393 / 6**	0 / 0 / 0

Table 8: Attack, scan and mixed post-infection traffic analysis.

TCP and UDP traffic and obtain the top three ports, which are measured and ranked by the number of connections on each port. Then, if the volume of traffic on the top three ports is higher than 70% of the total hourly traffic, we classify the period as an *attack*; if the volume is lower than 20%, we classify the period as a *scan*, otherwise we consider it a *mixed* period. We can see from Table 8 that both the Netgear and Brickcom honeypots show similar volume of TCP/UDP traffic. It is interesting to observe that the UDP traffic covers a wider range of ports in the *attack* scenarios then the TCP traffic, yet in the *scan* scenario, the number of ports for TCP and UDP are comparable. This leads to our preliminary conclusion that binaries that launched TCP attacks focused on a more limited number of ports than those using UDP. However, it requires further proof from binary analysis which is out of the scope of this paper. To our best knowledge, IoTPoT [30] is the only research effort that discussed how attackers monetised the compromised devices. However, it provided limited insights into binary clusters and traffic patterns. In this paper, we were able to quantify the relationship between binaries and the traffic observed with fine granularity.

Distributed Denial of Service attacks IoT botnets are well known to be primarily used to launch DDoS attacks. Leveraging the classification of traffic bursts into scan traffic and attack traffic, we next attempted to estimate the total number of DDoS attacks launched from our honeypots. We observed a total of 41 high-volume traffic peaks we attributed to DDoS attacks our honeypots have taken part of. On average each attack generated 141,962 packets and lasted 143 seconds, which gives an average of 993 packets per second. From the 41 attacks, 37 were carried out over UDP and 4 over TCP. Moreover, 38 attacks were targeted towards a single IP address. We recorded five noticeably massive attacks that lasted for several minutes: four DNS (`53/udp`) attacks at 6K packets per second with from 25 to 54Mb/s of traffic and one TCP SYN (`25565/tcp`) attack at 3K packets per second with 1Mb/s of traffic. These five attacks were targeted towards five different IP addresses, one IP address each. Interestingly, two of the IP addresses appeared to be hosting online gaming servers: one Steam and one Minecraft server. In fact, gaming servers seem to be regular targets of DDoS attacks as outlined by Brian Krebs' article [20] and Antonakakis *et al.*'s Mirai study [7]. The three other IP addresses belong to two hosting providers and one American university network.

4 Related Work

IoT honeypots ScriptGen by Leita *et al.* [22] is one of the earliest efforts in building high-interaction honeypots. It analyses the sequences of message exchanged between attackers and real servers and automatically derives a state machine that represents the observed interaction. In the same spirit, IoTCandyJar [23] proposed a technique that captures attackers requests, then scans the internet for real IoT devices that can respond to these requests and use machine learning to build a model to be used in future interactions with the attackers. However, the ethics of this approach with respect to routing traffic to real IoT devices remains debatable. Guarnizo *et al.* [18] proposed Siphon, an architecture to build a scalable high-interaction honeypot infrastructure backed by real IoT devices. Given that Siphon relies on real devices, its scalability is thus intrinsically limited. Authors also fail to explain how they actually perform the IoT device instrumentation and reset to clean state. Pa *et al.* [30] proposed IoTPoT, a *telnet*-based IoT honeypot. Its core design philosophy is similar to ScriptGen. When an incoming command is unknown, it forwards its to a set of sandbox environments running an embedded Linux OS for different CPU architectures. The interaction between the attacker and the backend is modeled so that the system can later handle the same request without interacting with the backend. Following this design philosophy, Wang *et al.* [31] proposed ThingPot, a proof-of-concept honeypot for Philips HUE light bulbs.

Embedded device security and IoT botnets The insecurity of Internet-connected embedded devices have been studied for many years. In 2010, Cui *et al.* [15] already reported on more than 500K devices with weak or default credentials. In 2012, the Internet census powered by the IoT device-backed Carna botnet confirmed this trend [3]. In parallel, researchers have also been studying the security of embedded device firmware. Costin *et al.* performed extensive vulnerability assessment of IoT device firmware in [12, 13]. Zaddach *et al.* also proposed Avatar [32] and Chen *et al.* proposed Firmadyne [11], both of which provide an emulation environment for embedded devices that further enable vulnerability assessment of firmware images. All these studies highlight the myriad of software vulnerabilities crippling IoT products.

The most notable work on the IoT threat landscape is the recent forensic study of the Mirai botnet by Antonakakis *et al.* in [7]. By combining historical and heterogeneous data sources they were able to reconstruct the whole history of the infamous botnet and track its various evolutions following the release of its source code. Pa *et al.* also leverages their IoTPot [30] infrastructure to analyse *telnet*-based intrusion mechanisms and the behaviour of the few malware samples they collected. Recently, Cozzi *et al.* [14] published a study of Linux (and Linux-based IoT) malware describing some of the trends in their behaviour and level of sophistication.

Our work serves as a follow up to these previous studies. First, we go beyond the scope of the Mirai botnet and aim at providing a global picture of the current

IoT threat landscape that includes a myriad of other malware and botnets. We leverage some of the techniques used in the study of embedded device firmware security to build high-interaction IoT honeypots. Finally, unlike IoTPot and ThingPot, we do not restrict ourselves to attacks targeting a particular service or device but instead try to provide as much diversity as possible to get an accurate view into the threat landscape.

5 Conclusion

We used six months of data collected from our honeypots and enriched with various reputation feeds and binary analyses to report on the current threats targeting IoT devices. For instance, we have seen that while attackers still heavily rely on brute-forcing attacks against remote management interface of devices, a worrying and increasing number of botnets are getting equipped with a suite of exploits targeting a wide range of software vulnerabilities inside IoT device firmware, sometimes disclosed a couple of weeks before we start seeing them in the wild. Additionally, botnets powered by the Mirai malware appear to be dominating the IoT thread landscape but other new and old players, such as Hajime and IoT Reaper are aggressively claiming their share of the vulnerable IoT products. Finally, while the core business of IoT botnets is still DDoS attacks, some emerging botnets like IoT Reaper are yet to unveil their real purpose. In summary, we are now witnessing an abrupt change in the sophistication of the IoT threat ecosystem. We believe that, given the experience gained from studying the traditional malware ecosystem, we now have an opportunity to better anticipate and take proactive actions upon the evolutions of the IoT threat landscape.

References

1. Shodan. https://www.shodan.io/
2. VirusTotal. https://www.virustotal.com/
3. Internet Census. http://census2012.sourceforge.net/paper.html (2012)
4. CVE-2016-1555. https://nvd.nist.gov/vuln/detail/CVE-2016-1555 (2016)
5. CVE-2016-5681. https://nvd.nist.gov/vuln/detail/CVE-2016-5681 (2017)
6. CVE-2017-17107. https://nvd.nist.gov/vuln/detail/CVE-2017-17107 (2017)
7. Antonakakis, M., April, T., Bailey, M., Bernhard, M., Bursztein, E., Cochran, J., Durumeric, Z., Halderman, J.A., Invernizzi, L., Kallitsis, M., Kumar, D., Lever, C., Ma, Z., Mason, J., Menscher, D., Seaman, C., Sullivan, N., Thomas, K., Zhou, Y.: Understanding the Mirai Botnet. In: USENIX Security Symposium (2017)
8. Anubhav, A.: Agile QBot Variant Adds NbotLoader Netgear Bug in Its New Update. https://blog.newskysecurity.com/agile-122bf2f4e2f3 (July 2017)
9. Anubhav, A.: Masuta : Satori Creators' Second Botnet Weaponizes A New Router Exploit. https://blog.newskysecurity.com/masuta-satori-creators-second-botnet-weaponizes-a-new-router-exploit-2ddc51cc52a7 (January 2018)
10. Checkpoint: IoTroop Botnet: The Full Investigation. https://research.checkpoint.com/iotroop-botnet-full-investigation/ (October 2017)

11. Chen, D.D., Woo, M., Brumley, D., Egele, M.: Towards automated dynamic analysis for linux-based embedded firmware. In: NDSS (February 2016)
12. Costin, A., Zaddach, J., Francillon, A., Balzarotti, D.: A Large Scale Analysis of the Security of Embedded Firmwares. In: USENIX Security Symposium (2014)
13. Costin, A., Zarras, A., Francillon, A.: Automated Dynamic Firmware Analysis at Scale: A Case Study on Embedded Web Interfaces. In: ASIACCS (May 2016)
14. Cozzi, E., Graziano, M., Fratantonio, Y., Balzarotti, D.: Understanding Linux Malware. In: IEEE Symposium on Security and Privacy (May 2018)
15. Cui, A., Stolfo, S.J.: A Quantitative Analysis of the Insecurity of Embedded Network Devices: Results of a Wide-area Scan. In: ACSAC (December 2010)
16. Edwards, S., Profetis, I.: Hajime: Analysis of a decentralized internet worm for IoT devices. Rapidity Networks (October 2016)
17. Embedi: Enlarge your botnet with: top D-Link routers. https://embedi.com/blog/enlarge-your-botnet-top-d-link-routers-dir8xx-d-link-routers-cruisin-bruisin/ (September 2017)
18. Guarnizo, J.D., Tambe, A., Bhunia, S.S., Ochoa, M., Tippenhauer, N.O., Shabtai, A., Elovici, Y.: Siphon: Towards scalable high-interaction physical honeypots. In: CPSS (April 2017)
19. Kim, P.: Pwning the Dlink 850L routers and abusing the MyDlink Cloud protocol. https://pierrekim.github.io/blog/2017-09-08-dlink-850l-mydlink-cloud-0days-vulnerabilities.html (Sep 2017)
20. Krebs, B.: Who is Anna-Senpai, the Mirai Worm Author? https://krebsonsecurity.com/2017/01/who-is-anna-senpai-the-mirai-worm-author/ (January 2017)
21. Kumar, M.: Advanced Malware targeting Internet of the Things and Routers. https://thehackernews.com/2016/03/internet-of-thing-malware.html (March 2016)
22. Leita, C., Mermoud, K., Dacier, M.: Scriptgen: an automated script generation tool for honeyd. In: ACSAC (December 2005)
23. Luo, T., Xu, Z., Jin, X., Jia, Y., Ouyang, X.: IoTCandyJar: Towards an Intelligent-Interaction Honeypot for IoT Devices. In: Blackhat USA (July 2017)
24. Offensive Security: D-Link Devices - UPnP SOAP TelnetD Command Execution (Metasploit). https://www.exploit-db.com/exploits/28333/ (September 2013)
25. Offensive Security: Remote Buffer Overflow in Cookie Header. https://www.exploit-db.com/exploits/33863/ (June 2014)
26. Offensive Security: D-Link DIR-890L/R - Multiple Buffer Overflow Vulnerabilities. https://www.exploit-db.com/exploits/38716/ (November 2015)
27. Offensive Security: Brickcom Corporation Network Cameras - Multiple Vulnerabilities. https://www.exploit-db.com/exploits/39696/ (April 2016)
28. Offensive Security: Brickcom IP Camera - Credentials Disclosure. https://www.exploit-db.com/exploits/42588/ (July 2017)
29. Offensive Security: SSD Advisory – D-Link 850L Multiple Vulnerabilities. https://blogs.securiteam.com/index.php/archives/3364 (August 2017)
30. Pa, Y.M.P., Suzuki, S., Yoshioka, K., Matsumoto, T., Kasama, T., Rossow, C.: IoTPOT: Analysing the Rise of IoT Compromises. In: WOOT (August 2015)
31. Wang, M., Santillan, J., Kuipers, F.: ThingPot: an interactive IoT honeypot (2017)
32. Zaddach, J., Bruno, L., Francillon, A., Balzarotti, D.: Avatar: A Framework to Support Dynamic Security Analysis of Embedded Systems' Firmwares. In: NDSS (February 2014)