Stories from a Customized Honeypot for the IoT

Javier Carrillo-Mondéjar[1], José Roldán-Gómez[2], Juan Manuel Castelo Gómez[3], Sergio Ruiz Villafranca[3], Guillermo Suarez-Tangil[4]*

[1] *Departament of Computer Science and Systems Engineering, University of Zaragoza, Spain*
[2] *Department of Computer Science, University of Oviedo, Spain*
[3] *Albacete Research Institute of Informatics, University of Castilla-La Mancha, Spain*
[4] *IMDEA Networks Institute, Spain*
jcarrillo@unizar.es, roldangjose@uniovi.es, {juanmanuel.castelo, sergio.rvillafranca}@uclm, guillermo.suarez-tangil@imdea.org

Abstract

Since the inception of the Internet of Things (IoT), the security measures implemented on its devices have been too weak to ensure the appropriate protection of the data that they handle. Appealed by this, cybercriminals continuously seek out for vulnerable units to control, leading to attacks spreading through networks and infecting a high number of devices. On top of that, while the IoT has evolved to provide a higher degree of security, the techniques used by attackers have done so as well, which has led to the need of continuously studying the way in which these attacks are performed to gather significant knowledge for the development of the pertinent security measures.

In view of this, we analyze the state of IoT attacks by developing a high-interaction honeypot for SSH and Telnet services that simulates a custom device with the ARM architecture. This study is carried out in two steps. Firstly, we analyze and classify the interaction between the attacker and the devices by clustering the commands that they sent in the compromised Telnet and SSH sessions. Secondly, we study the malware samples that are downloaded and executed in each session and classify them based on the sequence of system calls that they execute at runtime. In addition, apart from studying the active data generated by the attacker, we extract the information that is left behind after a connection with the honeypot by inspecting the metadata associated with it. In total, more than 1,578 malicious samples were collected after 9,926 unique IP addresses interacted with the system, with the most downloaded malware family being Hajime, with 70.5% of samples belonging to it, and also detecting others such as Mirai, Gafgyt, Dofloo and Xorddos.

Keywords: Honeypot, Malware, IoT, Data analytics, Expert systems

1 Introduction

Nowadays, there are innumerable devices connected to the Internet which interact with each other, providing services to users that, until a few years ago, seemed unthinkable. The result of integrating technology in new environments different from conventional ones, namely the cloud, desktop or mobile, is what is we know as the Internet of Things (IoT). IoT devices make people's lives easier by easing tasks that users perform on a daily basis. Using a mobile device for controlling household appliances or utilizing your own voice for turning on the light or the television are some examples.

This means that the IoT is heavily involved in the activities that a person carries out in many aspects of their life. As a consequence, due to the high number of interactions that are made between user and device, both actively and passively, the resulting volume of data that is managed in this environment is immense. In addition, since some of this data can have a high degree of sensitivity, cybercriminals find it very appealing to attack IoT units. Unfortunately, although these devices provide a great number of features that are attractive to users, the security measures implemented on them are not strong enough to stop these attacks, thus making them vulnerable due to such simple aspects such as using default and easily guessable user and password combinations, having weak default settings or running well-known-to-be outdated and vulnerable software.

The characteristics mentioned above, together with the lack of knowledge that many users have regarding the use of new technologies, have led cybercriminals to focus many of their efforts on attacking IoT devices and obtaining financial returns from them. According to a recent report [1], the number of attacks on these devices reached one hundred million in 2019, exceeding the number of attacks in 2018 by seven times.

Under these circumstances, it is important to understand what activities attackers perform in order to compromise IoT devices and what actions they carry out once they have gained access to a system. To study this, the use of honeypots, which are devices which simulate systems that present some kind of vulnerability, is a very effective approach to attract attackers, and a technique that has been successfully used in other experiments such as [2]. This allows for different types of attacks to be captured and, upon analysis, the extraction of knowledge regarding the multiple techniques and tools used by attackers for carrying out their criminal schemes.

In this work, we have deployed a high interaction

*Corresponding Author: Javier Carrillo-Mondéjar; E-mail: jcarrillo@unizar.es

DOI: 10.53106/160792642024012501010

honeypot [3] that simulates a device with the Advanced Reduced Instruction Set Computer Machine (ARM) architecture. After performing an analysis of the interaction between this infrastructure and the attackers, the main contributions that can be extracted from this work are the following:

- We present a statistical analysis of the connections that took place in our honeypot, showing the geographical information about the origin of the attacks as well as the most commonly usernames and passwords used in brute force attacks, and the different remote hosts to which the attackers tried to connect once they had obtained valid credentials.
- We perform an evaluation of the interaction of the attackers with our system, classifying the sessions established according to the sequence of commands introduced by the attackers.
- We present an analysis of the files downloaded by the attackers in our infrastructure, mainly consisting of binary files, but also bash-scripts and compressed files.
- We evaluate the similarity between each pair of collected files and classify them using N-grams together with the Jaccard index.

The remainder of this paper is organized as follows. Section 2 studies the proposals form the scientific community. Section 3 describes the methodology followed, and Section 4 presents the data analysis for the experiment that was carried out. Finally, Section 5 presents the main conclusions.

2 Related Work

The concept of a honeypot was first introduced more than two decades ago when the first worms started to spread through Windows and Linux systems [2]. Honeypots are monitored systems which are exposed to the Internet with the aim of obtaining information about attacks that are occurring in real time. These systems can be classified as low, medium and high interaction depending on the functionality that is available for attackers [3]. Low and medium interaction systems do not present a complete system to the attacker (i.e., lack of commands, static file system or fixed command outputs [4]), while high interaction systems do provide a complete system to attackers and, therefore, it makes it difficult to fingerprint the honeypot based on its interaction or the tools that are available [5].

Nowadays, one of the main attack trends is targeting IoT devices, for the most part because a substantially number of these devices are more focused on providing new features to users rather than providing security or privacy measures [6] for protecting themselves and the data that they handle. In addition, since they are limited devices in terms of resources, it is highly unusual to find them using additional security measures such as AntiVirus (AV) or Intrusion Detection Systems (IDS) [7]. Due to this insecure nature, the research community, as well as the industry itself, use honeypots for detecting new threats and learning about the tactics, techniques and procedures used by attackers against these devices.

Pa et al. [8] designed a honeypot focused on Teletype Network (Telnet) attacks by combining a low-interaction honeypot with a sandboxing system. This way, when a command is unknown, it is sent to the sandbox in order to give a reliable response to the entered command, and it is stored for future requests. Another solution proposed by Šemić et al. [9] is a low-interaction honeypot for the Telnet protocol. The honeypot has two frontends, one dedicated to manual attacks, simulating some of the commands and components of a real system, and one specifically designed to respond to Mirai malware.

Other protocols used by IoT devices have been studied by Wang et al. [10], which proposed a high-interaction honeypot for Message Queuing Telemetry Transport (MQTT) and Extensible Messaging and Presence Protocol (XMPP) modules, while device emulation was performed via a Representational State Transfer (REST) Application Programming Interfaces (API). Luo et al. [11] designed a honeypot with intelligent interaction based on the responses received from other real IoT devices which it actively scans and sends requests logged from previous attacks, and combines it with the use of a reinforcement learning algorithm to give the best possible response to attackers.

Vetterl et al. [12] proposed a high-interaction honeypot for capturing attacks on Customer Premise Equipment (CPE) and IoT devices through emulation of their firmware. The honeypot is designed to obtain information on how the system is compromised and once the attack vector is known it is blocked.

Tambe et al. [13] introduced the idea of making use of a high-interaction honeypot using Virtual Private Network (VPN) tunnels so that a physical IoT device can be listening in different geographic locations, simulating multiple devices.

Cowrie [14] is an open-source honeypot that was created as a continuation and extension of Kippo [15]. It supports the Telnet and Secure SHell (SSH) protocols and, although it was initially designed as a medium-interaction honeypot, it can be used as a high-interaction honeypot allowing the Cowrie logging system to be used transparently with real or virtual devices. Fraunholz et al. [16] uses Cowrie as a medium-interaction honeypot to perform statistical and behavioral analysis on incoming attacks.

In summary, there are different studies made by the research community that use honeypots to analyze the different attacks on protocols or services used in the IoT. Unlike most related work, we used a high-interaction honeypot to collect attacks on Telnet and SSH services, where we classified the attackers' command sessions and downloaded malware samples to spread through these devices.

3 Methodology

In order to learn how SSH and Telnet services are exploited for gaining access to IoT devices and perform attacks through them, we deployed a high-interaction honeypot and monitored the actions and accesses carried out by exploiters. The methodology followed in this experiment, which is shown in Figure 1, is explained in this section.

Firstly, we describe the architecture of the deployed system and then we detail the configuration of the machines which simulated vulnerable devices. Finally, we present the analysis carried out in this study.

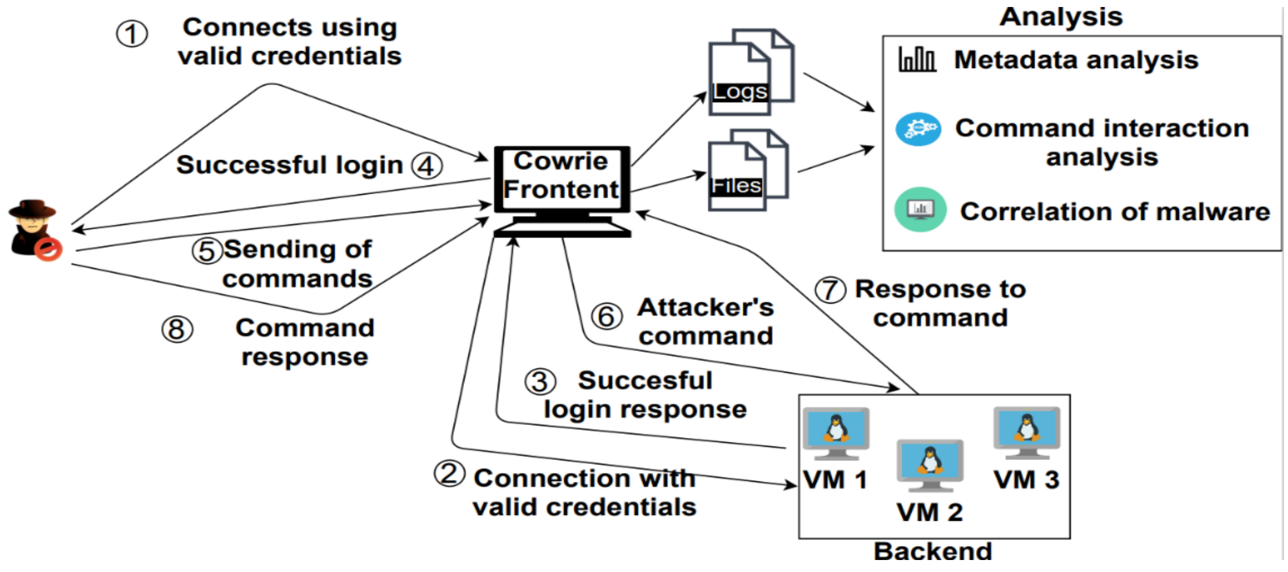

Figure 1. Outline of the methodology followed in this work

3.1 System Architecture

The architecture consisted of a private server that was deployed in Germany. We hardened the server in order not to allow unauthorized access to the services in this machine, and, after that, we deployed the honeypot on this machine and redirected the connections directed to the SSH and Telnet services to the internal ports used by Cowrie. We configured Cowrie to work in proxy mode, redirecting the traffic that reached these services to virtual machines compatible with QEMU [17]. This allows Cowrie to become a high-interaction honeypot because attackers are dealing with a complete system rather than an emulation of the file system and certain commands, thus making it more difficult for them to notice that they are not compromising a real system.

For user authentication, we only considered the root user and the top 10 most common passwords used by malware targeting IoT devices [18] as valid ones. This way, when an attacker entered valid credentials, Cowrie performed a successful login in the virtual machine, returned the response from the server of the virtual machine and started acting as a proxy, redirecting the input and output of commands between the attacker and the virtual machine. The proxy mode can be configured to redirect traffic to virtual machines or devices outside Cowrie, or Libvirt compatible virtual machines can be included and Cowrie can take care of deploying and restarting the machine when needed through this toolkit.

3.2 Virtual Machine Configuration

For the creation of the virtual machine used to emulate a real system, we used Buildroot [19], which automates the process of creating a Linux environment for embedded systems. Using this tool, we were able to build a Linux system for the ARM architecture by cross-compiling, so that, once that an attacker has gained access to the system and performs a reconnaissance of the environment, they find that the architecture being emulated is the most used one for IoT devices.

For the Linux system compiled, we included compatibility for the old application interface (OABI), allowing the execution of binaries created for older ARM architectures. Finally, we included different tools such as BusyBox, Perl, and Python. as well as SSH and Telnet servers to allow Cowrie to connect in its proxy mode.

Once the machine was built with Buildroot, we obtained the kernel that would be emulated and a file system and utilities that resemble those that an attacker would find on a real device. The system generated was emulated using QEMU, and, in order for Cowrie to manage the virtual machine, it was necessary to generate an Extensible Markup Language (XML) file containing the configuration options that QEMU needed to emulate the machine. Since we are dealing with a high interaction honeypot, we add a series of preventive measures to reduce the exposure surface and prevent attackers from carrying out attacks through our system. We only allow inbound traffic to SSH and Telnet ports, denying connection attempts to these ports on the outside. We also allow connections to port 80 and 8080 commonly used by the HTTP protocol and by attackers to download malware samples. Emulated virtual machines are rebooted and restored after a 10-minute time interval. We believe that these measures are sufficient to reduce the exposure of the honeypot, and that they in turn allow the collection of information on the most prominent attacks.

3.3 Data Analysis

In order to evaluate the actions carried out by the attackers, the metadata that could be extracted from the stored logs was thoroughly studied. This analysis was divided into three different tasks: the inspection of the data associated with the connections, the examination of the interaction that the attackers made with the system, and the study the files that were downloaded in our honeypot.

Metadata analysis. In this phase, we analyzed the data associated with the connections made to our honeypot system. Firstly, we focused on the connection attempts made throughout the experiment, and then broke the data down into days of the week. Secondly, we evaluated all the login attempts that were made, also studying the username and password combinations most frequently used by the attackers. Thirdly, we studied the attacks that were attempted through the SSH feature known as port forwarding. This SSH feature allows the redirection of any Transmission Control Protocol (TCP) port and the sending of data via SSH, allowing, for example, accessing geolocation-restricted content, bypassing firewalls, etc. Therefore, the server would act similarly to a proxy and the connection data recorded on the target host would be from the SSH server. Finally, we analyzed the origin of the attacks by consulting the geolocation of the Internet Protocol (IP) addresses that interacted with our honeypot using public IP location services [20].

Interaction analysis. To analyze and classify the interaction, we extracted all the command sessions for each of the IP addresses. Then, we cleaned duplicated sessions from the same IP, i.e., sessions that are exactly the same and therefore have been created by bots that made another connection to the system and performed exactly the same tasks as in other connection. After this, we standardized the commands by eliminating specific semantics that can be variable and still be the same command. For this purpose, we replaced the following variables by constant values using pattern search and regular expressions:

- IP addresses and Uniform Resource Locators (URLs)
- Names of downloaded files or scripts
- Payload hardcoded in commands
- SSH keys
- Inserting users
- Replacing non-existing BusyBox commands

Once the commands entered had been standardized, we separated each one of them. In addition, those symbols that allow different commands to be chained together in the same order, such as the semicolon, were removed. After this cleaning and standardization stage, each session was composed of an ordered vector where each element represented a command.

For measuring the similarity between sequences and performing the classification, we use the cosine similarity between two vectors [21]. Given to sequences A and B, the first step is to transform each one of them into a vector, obtaining $\vec{A} = \{a_1, a_2, \ldots, a_n\}$ and $\vec{B} = \{b_1, b_2, \ldots, b_n\}$, where a_i and b_i , represent the number of times in which the command in position i appears in the sequence, and n is the total number of different commands in the whole set of sequences that are being compared.

Finally, given two vectors of command sessions, we compute the cosine similarity as follows:

$$similarity(A,B) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \times \|\vec{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}. \quad (1)$$

The obtained result is in the interval [0,1], with 1 meaning a perfect similarity and 0 a nonexistent one. In other contexts, this interval can be [-1,1], but in this experiment it is not possible to obtain negative values for the vectors.

Analysis of downloaded samples. For the analysis and clustering of the downloaded files in the honeypot, we run the samples in an IoT sandbox environment [22], which automatizes the analysis and feature extraction process of pieces of malware from various architectures. For the clustering and classification tasks, we follow a dynamic approach, due to the fact these might be samples that are packed, so features based on static analysis are less robust to obfuscation and therefore more prone to false positives. We run each sample and extract its sequences of system calls (syscalls) of size N, also known as N-grams. For example, for the following trace of syscalls: [execve, time, getpid, getppid], the set of n-grams of size 2 that will be obtained is: (execve, time), (time, getpid) and (getpid, getppid). Once that the N-grams from the different samples were extracted, we calculated the similarity using the Jaccard index [23], which allowed us to determine the similarity between two sets in the following way:

$$Jaccard\_index(s_1, s_2) = \frac{|s_1| \cap |s_2|}{|s_1| \cup |s_2|}, \quad (2)$$

where the numerator represents the subsets (N-grams) found in both samples and the denominator indicates all unique subsets between samples. The result is a value between 0 and 1 representing the similarity between two sets.

4 Data Analysis

Our honeypot system ran for a period of 35 days. During this time, the actions carried out on the system were monitored and the corresponding data was collected. This section presents the results than can be extracted from this experiment.

4.1 Metadata Analysis

In this section we analyze the information associated with the connections or connection attempts that were logged at the honeypot. Firstly, we perform a study based on the number of attempts and the time stamp at which they occurred. Secondly, we analyze the login attempts. Thirdly, we analyze the connection attempts via SSH tunnels to other hosts, and finally we analyze the origin of the IP addresses that interacted with the honeypot.

Analysis of connections. We analyze the timestamp of the connection attempts recorded by the honeypot using Spanish local time (Greenwich Mean Time +1). In the thirty-five days of the experiment, the system captured a total of 830,053 connection attempts. Figure 2 represents the number of attempts per protocol. It can be seen that most of the connections were through the SSH protocol, reaching a total of 781,339 connection attempts. Which is more noticeable from this data is that the number ofconnections captured via the Telnet protocol (48,714) is so low, as for years it was the preferred protocol for brute-force attacks on IoT devices.

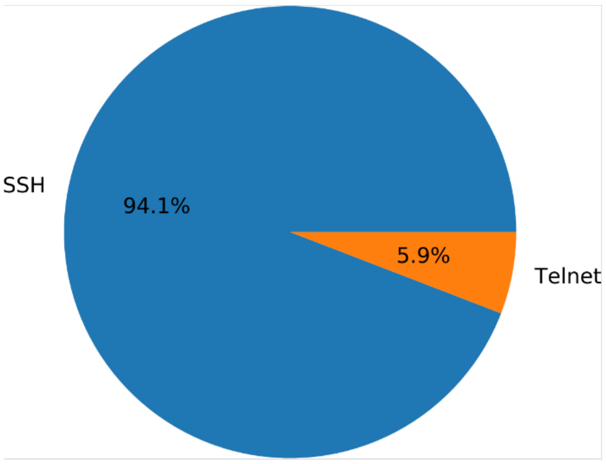

Figure 2. Percentage of connections by protocol

As for the timeline of the connection attempts captured, Figure 3 shows the attempts that occurred on each of the days of the experiment. Red represents those attempts that occurred between 00:00 and 08:00, yellow represents those that occurred between 08:00 and 16:00, and green represents those that occurred between 16:00 and 00:00. Looking at the results, it can be seen that, on average, there were 20,000 attempts per day, with this figure even exceeding 30,000 on some days. In general, in terms of time periods, the number of attacks remains uniform and does not show any tendency towards a specific one. This indicates that most of the attacks were automated and not focused on out-of-office hours such as the 00:00 to 08:00 period.

Figure 4 shows the connection attempts that were made on each day of the week. It can be seen that the number of attempts is very similar for each day and, although the peak is reached on Sunday, it does not seem to follow any trend.

Analysis of login attempts. A total of 769,685 honeypot login attempts were captured, of which 539,369 (70.07%) succeeded in logging into the system and 230,316 (29.93%) were unsuccessful. The successful logins belonged to 1,534 unique IP addresses, i.e., only 15.45% of the unique attackers managed to access the system.

Of the login attempts that occurred on the system, 70.91% and 71.19% of the login pairs were generated from the list of users and passwords used by Mirai and from a specific dictionary of users and passwords of IoT devices, respectively. Table 1 shows the top 10 most used both users and passwords, as well as the top 10 most used combinations for logging into the system. In the table it can be seen how the attackers try to use usernames and passwords that are clearly commands (e.g., iptables, ping, sh, shell, etc.). This is due to scripts that are not able to capture the fact that they are facing a system that requires authentication and the script continues its execution, evidencing the lack of sophistication of some attacks.

It can be seen that brute-force attacks on SSH and Telnet credentials are still one of the main methods of finding vulnerable systems and that the trend has continued since the release of Mirai, such methods allowing attackers to compromise systems quickly without investing too much effort or money in searching for vulnerabilities or zero-days.

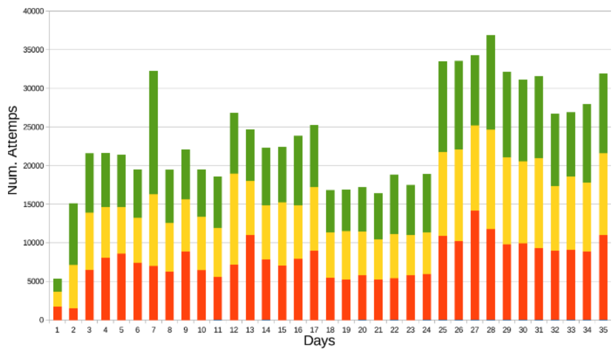

Figure 3. Number of connection attempts captured

(Red represents attempts made between 00:00 and 08:00, yellow between 08:00 and 16:00 and green between 16:00 and 00:00.)

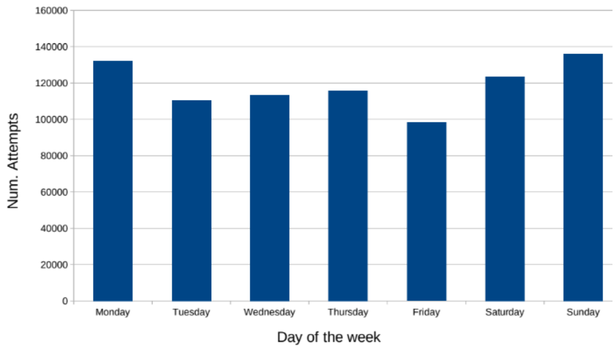

Figure 4. Number of connection attempts per day of the week

Table 1. Top 10 users, passwords and combinations of both most used by attackers

Users	Total	Passwords	Total	Pairs	
root	640,170	admin	411,542	root/admin	410,365
admin	16,006	root	126,693	root/root	124,801
enable	4,295	123456	9,781	enable/system	4,293
sh	4,200	123	4,401	sh/shell	4,185
test	2,927	system	4,394	ping ; sh//bin/bu..	2,002
user	2,596	shell	4,198	root/54321	1,888
ping ; sh	2,002	/bin/busy…	3,549	admin/888888	1,882
ubuntu	1,921	888888	2,771	root/8888	1,825
postgres	1,836	5555	2,590	root/5555	1,824
Iptables -F	1,547	password	2,095	root/111111111	1,821

Direct TCP-IP connection analysis. We analyzed the connection attempts that allowed attackers to create TCP sessions in an SSH tunnel. In total, there were 1,278,006 connection attempts using this SSH feature, of which 33,804 were unique targeted hosts. It is important to note that all of these connection attempts were made by only 96 attackers or bots. Figure 5 graphically shows the top 30 hosts to which the most connection attempts were made. The red nodes represent the IPs of the attackers, while the blue nodes represent the hosts to which they tried to connect to. The edges represent whether there were any connection attempts between the attacker IP and the targeted node, and the thickness of the edges varies according to the number of times the attacker tried to connect to the targeted host through our SSH server. It should be noted that no connection was made from our honeypot to the outside. Also, those targeted hosts to which they tried to connect directly through their IP address were anonymised. As can be seen in the image, some well-known domains appear, such as Google, Amazon, Evernote or the Russian search engine 'ya.ru', to which 30.5% of the total connection attempts belong to. Making requests to known domains could be a way to check whether the redirection is working correctly and whether they are in a honeypot.

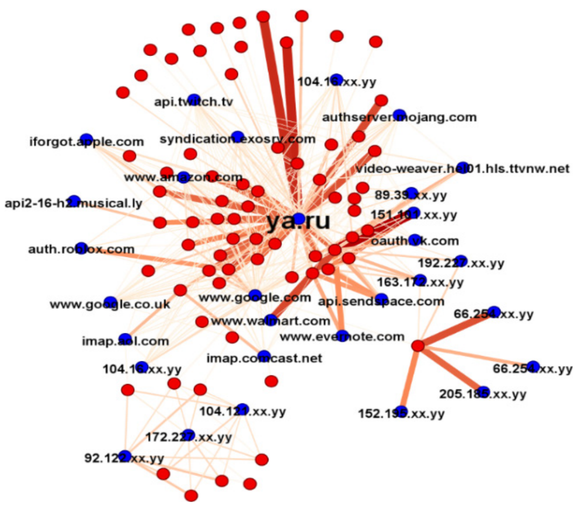

Figure 5. Connection attempts via ssh direct TCP-IP

(The blue nodes represent the hosts or addresses they attempted to connect to and the red nodes represent the attacker's ip address. The edge weight indicates the number of connections between both nodes.)

In addition, we analysed the destination ports to which the requests were directed. Table 2 shows the top 10 ports and indicates that most of the requests were directed to connections via the Hypertext Transfer Protocol (HTTP) and the Hypertext Transfer Protocol Secure (HTTPS). The rest of the connections in the top 10 are mainly related to sending and receiving email and the different protocols used for this. Therefore, they were trying to connect to Simple Mail Transfer Protocol (SMTP) servers to send emails anonymously, something commonly used in spam and phishing campaigns.

Table 2. Top 10 ports to which petitions were addressed

Port	Total	Protocol information
443	526,670	https
80	508,109	http
25	176,972	smtp
993	24,855	imaps
587	22,627	smtp over tls
465	5,183	smtp over tls
43594	4,802	runescape servers
143	3,409	imap
26	1,988	smtp
2525	1,952	smtp

Finally, an unusual port can be observed. After applying Open Source INTelligence (OSINT) [24] techniques, we observed that this port is used by the servers of RuneScape [25], an online role-playing game that is developed by Jagex. This could be used to perform a Distributed Denial of Service (DDoS) attack on the game server or to evade bans based on IP addresses in the game.

Origin of the attacks. In total, 9,926 unique IP addresses interacted with the system. We obtained the origin of the addresses by using location services and plotted them on a map. Figure 6 shows a heat map and the representation of the existing IP addresses in that geographical area. It can be seen that the origin of most attacks is in Asia, Europe and the United States, with China being the most common location with 28.03% of the IP addresses collected, followed by the United States (10.89%), France (5.67%), Brazil (4.49%) and South Korea (4.07%). It is worth noting that 51.42% of the IP addresses collected come from Asia, which is twice the number of IPs from Europe (24.20%). It is also relevant that the origin of the actors behind the attacks is not necessarily that location specifically, as they could be using some kind of proxy, virtual private networks (VPNs), Tor or systems that have been previously compromised through malware and are acting as bots looking for other vulnerable systems to spread malware.

4.2 Interaction Classification

This section presents an analysis of the interaction of the different attacker sessions via the SSH and Telnet protocols. As discussed in Section 4.1, only 1,534 unique IP addresses successfully logged in. Out of that entire pool of IP addresses, at least 1,402, or 91.4%, executed at least one command. In total, 4,217 sessions were established, i.e., some of these IP addresses connected to the honeypot several times.

To analyze the interaction, automated techniques were applied to classify the sessions and extract knowledge. Firstly, as described in Section 3.3, the commands entered were standardized and the most commonly used commands were extracted. Table 3 shows the top 10 commands most used by attackers. In the table, we can see some that are quite common, such as system, shell, and enable, and which are normally used on some devices such as routers to obtain shell or more privileged commands. Also, the command "echo CODE >> .file", which dumps binary code to a file, can be observed. The most striking command is the one that was repeated most often as it is an invalid command. Attackers use the command /bin/busybox followed by non-existent "command" names such as CORONA, TSUNAMI, FBOT, etc., in order to find out if the previous commands have been executed correctly [26].

Table 3. Top 10 most used commands

Command	Total
/bin/busybox BUSYBOX	6,065
sh	4,483
shell	4,197
system	4,171
enable	4,171
linuxshell	3,397
/bin/busybox cat /bin/busybox	2,629
>.file	2,099
>file	1,930
echo CODE >>.file	1,272

Secondly, the similarity between each pair of sessions was calculated using the cosine similarity. In order to consider two similar sessions, different thresholds were tested, and finally we selected one with the value of 0.9, i.e., two sessions are similar if the cosine similarity is greater than 0.9. Figure 7 shows the results of clustering sessions based on the commands entered during the session. The nodes represent the sessions, and the edges connect two sessions if their cosine similarity exceeds the set threshold. It can be seen that most of the sessions are similar to each other and that they are mostly grouped in 7-8 clusters, indicating that most of the connections were made by bots searching for vulnerable systems to download and install malware using similar tactics, techniques and procedures (TTP).

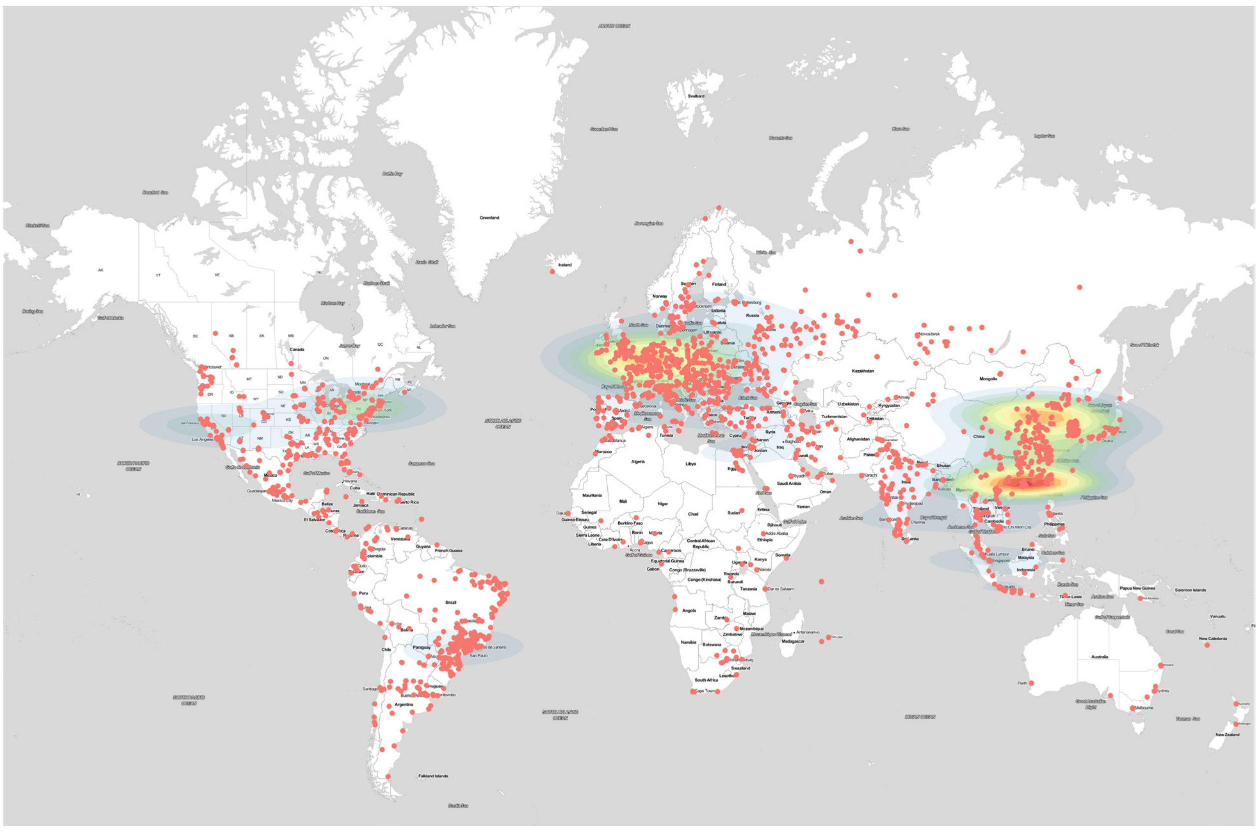

Figure 6. Origin of the attacks received in the honeypot

4.3 Downloaded Malware Analysis

This section presents the results of the analysis of the samples collected by the honeypot. In total, 1,578 samples were collected, of which 710 were unique. The unique samples included 590 Linux executable binary files, 35 gzipped files, 82 bash script files and 3 perl scripts.

Binary files. These were executable and Linkable Format (ELF) binary files, mainly from the 32-bit ARM architecture (87.46%). The rest of the samples corresponded to other architectures such as Intel 80386, Microprocessor without Interlocked Pipeline Stages (MIPS), etc. For the classification of the samples, the syscall sequences were extracted as discussed in Section 3.3, and N-grams were extracted for each syscall sequence using four as the N-gram size. We use a dynamic approach and the extraction of syscalls since it allows analyzing and relating samples from other architectures (i.e., ARM, MIPS, Intel 80386). Figure 8 shows the results of the clustering of the collected samples. The nodes represent each of the samples and the edges join nodes that have a similarity greater than 80%. It can be seen that there is one cluster that stands out from the rest, and then there are small clusters or sets of connected samples.

We applied reverse engineering techniques in different samples from each cluster to confirm that the samples were clustered correctly. The samples that are clustered with any other sample based on the established similarity threshold are the following:

- **Hajime:** To this category belong 70.5% of the samples found, and it is this grouping that stands out from the rest in Figure 8. This sample is Hajime's downloader, a malware that first appeared in 2016 and spreads mainly via Telnet and vulnerabilities whose exploits it has been incorporating into its scanning and propagation module [27]. It is a botnet that communicates with its C&C through a decentralized network and its real purpose is so far unknown as it has not launched any denial-of-service (DoS) attacks [28-29]. The binaries found all have the same size and are responsible for downloading the sample from the next phase [30]. Mainly, it is the same binary with only changes in the address and download port of the next stage of the malware.
- **Mirai:** This is the most popular family of malware that attacks IoT devices. It became famous because it generated the largest DDoS attack using IoT devices [31]. Its source code was leaked in 2016 and since then multiple Mirai variants have appeared. It accounts for 9.83% of the total binary files.
- **Gafgyt:** This accounts for 6.44% of the executable files collected. It is malware whose source code was leaked in 2015 and, like with Mirai, there are many variants of this malware family. Among its main features is the ability to perform various DDoS attacks [32].
- **Dofloo:** This is malware that allows DDoS attacks and the loading of cryptocurrency miners [33-34], and accounts for 1.52% of the samples.
- **Xorddos:** This is malware that affects Linux-type devices and allows different types of DDoS attacks [35]. This family accounts for 2.54% of the samples.
- **Others:** The remaining samples, which were not related to any other sample on the basis of the similarity index, belong to this category.

Bash-script. These were script files downloaded in the sessions and designed to download malware for different architectures and execute it. All scripts work in a similar way: 1) they move to a directory where the user has permissions; 2) they download the malware for different architectures via wget, tftp or curl; 3) they give the downloaded file execution permissions; and 4) they execute the file with or without arguments. All files perform the same task and differ only in the IP addresses they try to connect to, the filename and the architectures supported by the malware.

Perl-script. The downloaded scripts were Internet Relay Chat (IRC) bots written in Perl with command execution capabilities. All three scripts were based on source code that is publicly available on the Internet, with two of them even having the same comments as the versions they were based on [36-37].

Compressed files. These were files in "tar.gz" format that had a hidden folder with the name *rsync*. The sum of all the unzipped files was 979, 106 of which were unique. There were 34 shared libraries, 31 executable files and 40 script files. In general, they were different versions of malware designed to mine cryptocurrencies. Such botnets provide cybercriminals with a network for distributed mining and financial gain, either by saving electricity bills or by obtaining cryptocurrencies [38]. Each compressed file contains several scripts that are responsible for finding and stopping the execution of any other mining malware and initiating the execution of its own malware. They also carry different versions of the executable used to perform the mining as well as the necessary libraries for ARM, x86 and x86-64 architectures.

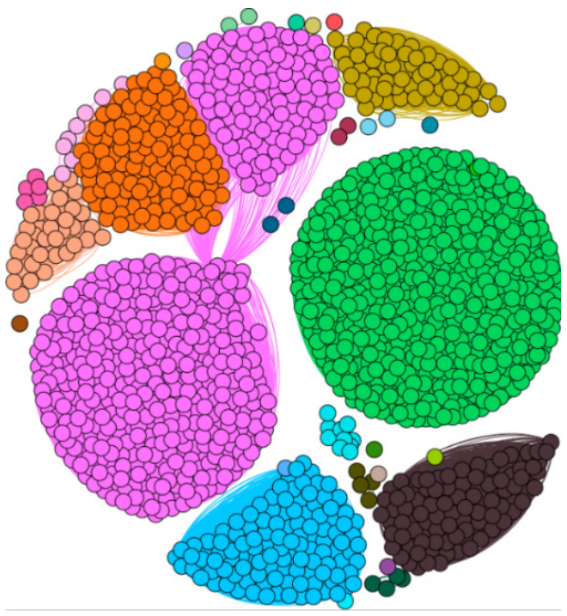

Figure 7. Clustering of sessions according to the commands entered

(Each node represents a session and edges between two nodes indicate that the similarity is above the set threshold of 0.9.)

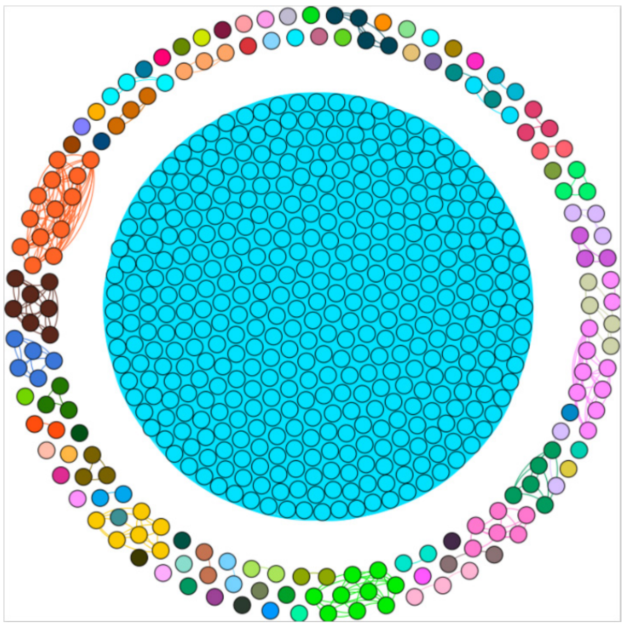

Figure 8. Clustering of the samples captured in the honeypot

(Each node is a sample and an edge connects two nodes if they have a similarity greater than 80%.)

5 Limitations and Future Work

This study is focused on Telnet and SSH services and the actions that attackers take once they take control. In view of the results obtained, it can be seen that brute force attacks on these services continue to be one of the preferred methods to take control of devices and infect them. However, in addition to brute force, attackers also exploit known vulnerabilities in other services to gain access to devices (e.g., HTTP, HNAP, or UPnP).

Not only conventional protocols are targeted by cybercriminals, IoT-centered ones such as MQTT are as well. The study of these types of protocols opens an interesting line of research that, although is out of the scope of our work, can lead to gaining additional knowledge on how cybercriminals behave when operating in the IoT.

Additionally, the use of search engines in the early phases of the fingerprinting technique is another interesting aspect in which to delve into. This is of added interest when working with honeypots, as making them accessible by these types of tools can lead to cybercriminals suspecting that the IoT system that is behind the service is not a real one.

6 Conclusions

In this study, a high-interaction honeypot has been deployed for a period of one month, mimicking the behaviour of an IoT device. The data captured have been analysed yielding valuable insight of the actions carried out by the attackers once the Telnet and SSH services had been compromised. Firstly, a statistical analysis was performed based on connection attempts, authentication attempts, IP addresses and attack attempts using SSH port forwarding. Then, we analysed the interaction of the attackers with the honeypot and classified the sessions established. Finally, we analysed the different files that were downloaded on our honeypot system by classifying the ELF format binary files.

During this experiment, a total of 830,053 connection attempts were made to the system, the majority of them, namely 781,339, through the SSH protocol, with the rest of them made via Telnet. 769,685 times these connections were translated into login attempts, with 15.45% of them succeeding. In these sessions, 1,578 malicious samples were downloaded, and, after clustering them, it was determined that 70.5% of them belonged to the Hajime downloader malware family. However, others such as Mirai, Gafgyt, Dofloo and Xorddos were detected. With respect to the origin of the attacks, 9,926 unique IP addresses interacted with the system, with the top three attacking countries being China, the United Stated and France, with 28.03%, 10.89%, and 5.67% of connections respectively.

The results show that most of the sessions established were conducted in an automated manner by bots searching for brute-force vulnerable servers on which to install and execute their malware. With respect to the variants found, it can be concluded that there are old malware families that are still operating and actively look for new systems to add to their botnet, as well as others that aim to use these devices for cryptocurrency mining.

With this work we have been able to gain a better understanding of the actions that attackers take when targeting vulnerable services, thus providing the research community with valuable knowledge on the behaviour of cybercriminals in the IoT, which is one of the main issues currently under study due to their importance for users and their data.

Acknowledgments

This work has been supported by the Spanish Ministry of Science and Innovation under the project PID2021-123627OB-C52, by the Regional Government of Castilla-La Mancha under the SBPLY/21/180501/000195, by the Spanish Education, Culture and Sports Ministry under grant FPU 17/03105 and by MCIN/AEI/10.13039/501100011033 and European Union NextGenerationEU/PRTR under grant TED2021-131115A-I00. Guillermo Suarez-Tangil was funded by the "Ramon y Cajal" Fellowship RYC-2020-029401-I, and supported by TED2021-132900A-I00 from the Spanish Ministry of Science and Innovation, both grants funded by MCIN/AEI/10.13039/501100011033 and ESF "The European Social Fund invests in your future".

References

[1] D. Demeter, M. Preuss, Y. Shmelev, IoT: a malware story - Securelist, October, 2019, https://securelist.com/iot-a-malware-story/94451/.

[2] M. Nawrocki, M. Wählisch, T. C. Schmidt, C. Keil, J. Schönfelder, A survey on honeypot software and data analysis, August, 2016, https://arxiv.org/abs/1608.06249.

[3] W. Fan, Z. Du, D. Fernández, V. A. Villagrá, Enabling an Anatomic View to Investigate Honeypot Systems: A Survey, *IEEE Systems Journal*, Vol. 12, No. 4, pp. 3906–3919, December, 2018.

[4] W. Z. Cabral, C. Valli, L. F. Sikos, S. G. Wakeling, Advanced cowrie configuration to increase honeypot deceptiveness, *IFIP International Conference on ICT Systems Security and Privacy Protection*, Oslo, Norway, 2021, pp. 317–331.

[5] A. Vetterl, R. Clayton, Bitter Harvest: Systematically Fingerprinting Low- and Medium-interaction Honeypots at Internet Scale, *12th USENIX Workshop on Offensive Technologies*, Baltimore, Maryland, USA, 2018, pp. 1–9.

[6] M. Frustaci, P. Pace, G. Aloi, G. Fortino, Evaluating Critical Security Issues of the IoT World: Present and Future Challenges, *IEEE Internet of Things Journal*, Vol. 5, No. 4, pp. 2483–2495, August, 2018.

[7] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, S. Shieh, IoT Security: Ongoing Challenges and Research Opportunities, *2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*, Matsue, Japan, 2014, pp. 230–234.

[8] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, C. Rossow, IoTPOT: Analysing the Rise of IoT

Compromises, *9th USENIX Conference on Offensive Technologies*, Washington, D.C., USA, 2015, pp. 1–9.

[9] H. Šemić, S. Mrdovic, IoT honeypot: A multi-component solution for handling manual and Mirai-based attacks, *2017 25th Telecommunication Forum*, Belgrade, Serbia, 2017, pp. 1–4.

[10] M. Wang, J. Santillan, F. Kuipers, ThingPot: an interactive Internet-of-Things honeypot, July, 2018, https://arxiv.org/abs/1807.04114.

[11] T. Luo, Z. Xu, X. Jin, Y. Jia, X. Ouyang, Iotcandyjar: Towards an intelligent-interaction honeypot for iot devices, *Black Hat*, Las Vegas, Nevada, USA, 2017, pp. 1–11.

[12] A. Vetterl, R. Clayton, Honware: A virtual honeypot framework for capturing CPE and IoT zero days, *2019 APWG Symposium on Electronic Crime Research*, Pittsburgh, Pennsylvania, USA, 2019, pp. 1–13.

[13] A. Tambe, Y. L. Aung, R. Sridharan, M. Ochoa, N. Tippenhauer, A. Shabtai, Y. Elovici, Detection of Threats to IoT Devices Using Scalable VPN-Forwarded Honeypots, *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy*, Richardson Texas, USA, 2019, pp. 85–96.

[14] M. Oosterhof, Cowrie SSH and Telnet Honeypot, 2015, https://www.cowrie.org/.

[15] U. Tamminen, Kippo honeypot, 2014, https://github.com/desaster/kippo.

[16] D. Fraunholz, D. Krohmer, S. D. Anton, H. D. Schotten, Investigation of cyber crime conducted by abusing weak or default passwords with a medium interaction honeypot, *2017 International Conference on Cyber Security And Protection Of Digital Services*, London, UK, 2017, pp. 1–7.

[17] F. Bellard, QEMU, a Fast and Portable Dynamic Translator, *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, Anaheim, California, USA, 2005, Article No. 41.

[18] J. Margolis, T. T. Oh, S. Jadhav, Y. H. Kim, J. N. Kim, An In-Depth Analysis of the Mirai Botnet, *2017 International Conference on Software Security and Assurance*, Altoona, Pennsylvania, USA, 2017, pp. 6–12.

[19] A. Sirotkin, Roll Your Own Embedded Linux System with Buildroot, *Linux Journal*, Vol. 2011, No. 206, June. 2011.

[20] IP-API.com, Geolocation API, 2022, https://ip-api.com/.

[21] B. Li, L. Han, Distance weighted cosine similarity measure for text classification, *International conference on intelligent data engineering and automated learning*, Hefei, China, 2013, pp. 611–618.

[22] J. Carrillo-Mondejar, J. M. C. Gomez, C. Núñez-Gómez, J. Roldán, J. L. Martínez, Automatic Analysis Architecture of IoT Malware Samples, *Security and Communication Networks*, Vol. 2020, Article No. 8810708, October, 2020.

[23] S. Fletcher, M. Z. Islam, Comparing sets of patterns with the Jaccard index, *Australasian Journal of Information Systems*, Vol. 22, pp. 1–17, March, 2018.

[24] J. Pastor-Galindo, P. Nespoli, F. G. Mármol, G. M. Pérez, The Not Yet Exploited Goldmine of OSINT: Opportunities, Open Challenges and Future Trends, *IEEE Access*, Vol. 8, pp. 10282–10304, Janunary, 2020.

[25] Jagex, RuneScape Online Community - Forums, 2022, https://www.runescape.com/community.

[26] M. S. Pour, J. Khoury, E. Bou-Harb, HoneyComb: A Darknet-Centric Proactive Deception Technique For Curating IoT Malware Forensic Artifacts, *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, Budapest, Hungary, 2022, pp. 1–9.

[27] S. Herwig, K. Harvey, G. Hughey, R. Roberts, D. Levin, Measurement and analysis of Hajime, a peer-to-peer IoT botnet, *Network and Distributed Systems Security Symposium*, San Diego, California, USA, 2019, pp. 1–15.

[28] J. Leyden, Mysterious Hajime botnet has pwned 300,000 IoT devices, April, 2017. https://www.theregister.com/ 2017/ 04/27/hajime_iot_botnet/

[29] S. Yamaguchi, White-Hat Worm to Fight Malware and Its Evaluation by Agent-Oriented Petri Nets, *Sensors*, Vol. 20, No. 2, Article No. 556, January, 2020.

[30] S. Edwards, Ioannis Profetis, Hajime: Analysis of a decentralized internet worm fot IoT devices, Rapidity Networks, October, 2016.

[31] 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, Y. Zhou, Understanding the Mirai Botnet, *26th USENIX Security Symposium*, Vancouver, BC, 2017, pp. 1093–1110.

[32] A. Marzano, D. Alexander, O. Fonseca, E. Fazzion, C. Hoepers, K. Steding-Jessen, M. Chaves, I. Cunha, D. Guedes, W. Meira, The Evolution of Bashlite and Mirai IoT Botnets, *2018 IEEE Symposium on Computers and Communications*, Natal, Brazil, 2018, pp. 00813–00818.

[33] K. Angrishi, Turning internet of things (iot) into internet of vulnerabilities (iov): Iot botnets, February, 2017, https://arxiv.org/abs/1702.03681.

[34] S. Gatlan, Exposed Docker APIs Abused by DDoS, Cryptojacking Botnet Malware, *BleepingComputer*, June, 2019, https://www.bleepingcomputer.com/news/security/ exposed-docker-apis-abused-by-ddos-cryptojacking-botnet-malware/.

[35] M. Donno, N. Dragoni, A. Giaretta, A. Spognardi, Analysis of DDoS-capable IoT malwares, *2017 Federated Conference on Computer Science and Information Systems*, Prague, Czech Republic, 2017, pp. 807–816.

[36] H3LLS1NG, Shadow-Network/perl-scripts, 2020, https://github.com/Shadow-Network/perl-scripts.

[37] Phl4nk, Ircbot, February, 2018, https://github.com/phl4nk/ircbot.

[38] J. Carrillo-Mondéjar, J. L. Martínez, G. Suarez-Tangil, Characterizing Linux-based malware: Findings and recent trends, *Future Generation Computer Systems*, Vol. 110, pp. 267–281, September, 2020.

Biographies

Javier Carrillo-Mondéjar received the B.Sc. and M.Sc. degrees in Computer Science and Engineering and the Ph.D. degree in Advanced Computing Technologies from the University of Castilla-La Mancha, Spain, in 2016, 2017, and 2022, respectively. His research interests are related to malware detection and classification techniques, with a particular focus on IoT/firmware cybersecurity.

José Roldán-Gómez obtained a B.S. degree in Computer Engineering in 2017, a M.S. degree in Computer Engineering in 2018, and the Ph.D. degree in Advanced Computing Technologies in 2023, all of them at the University of Castilla-La Mancha. His main interests are artificial intelligence applied to threat detection in IoT environments and automatic rule generation in CEP engines.

Juan Manuel Castelo Gómez received his M.Sc degree in Computer Science from the University of Castilla La Mancha (Spain) in 2017, and in 2021 obtained his Ph.D in Advanced Information Technologies. His research interests are related to cybersecurity, especially digital forensics, as well as malware detection.

Sergio Ruiz Villafranca received a BSc degree in Computer Engineering from the University of Castilla-La Mancha in 2017. Since 2021, he is a Ph.D student in Advanced Information Technologies at the aforementioned university. His research interests are related to cybersecurity, especially Industrial Internet of Things, machine learning for anomaly detection.

Guillermo Suarez-Tangil is with IMDEA Networks Institute. His research focuses on systems security and malware analysis and detection. In particular, his area of expertise lies in the study of smart malware, ranging from the detection of advanced obfuscated malware to the automated analysis of targeted malware. Before joining IMDEA, he was Lecturer at King’s College London (KCL) and before that senior research associate at University College London (UCL), where he was also actively involved in other research areas involved with detecting and preventing hate in OSN and Mass-Marketing Fraud.