PaperW8: An IoT Bricking Ransomware Proof of Concept

Calvin Brierley, Jamie Pont, Budi Arief, David J. Barnes, Julio Hernandez-Castro
School of Computing, University of Kent
[c.r.brierley,j.pont,b.arief,d.j.barnes,jch27]@kent.ac.uk

ABSTRACT

Internet of Things (IoT) devices are used in many facets of modern life, from smart homes to smart cities, including Internet-enabled healthcare systems and industrial control systems. The prevalence and ubiquity of IoT devices makes them extremely attractive targets for malicious actors, in particular for taking control of vulnerable devices and demand ransom from their owners. The aim of this paper is twofold: to investigate the viability of a ransomware-type attack being carried out on IoT devices; and to explore what damage can be inflicted upon devices after they have been compromised. To test whether ransomware is a viable method for attacking IoT devices, we developed our own proof of concept malware for Linux-based IoT devices dubbed "PaperW8". We looked at feasible ways for infecting IoT devices, as well as potential methods for gaining control and applying persistent changes to the target device. We successfully created a proof of concept ransomware, which we tested against six vulnerable IoT devices of various brands and functions, some of which are known to have been targeted in the past but are still widely in use today. Developing this proof of concept tool allowed us to identify the main requirements for a successful ransomware attack against IoT devices. We also determined some limitations of IoT devices that may discourage attackers from developing IoT-specific ransomware, while highlighting workarounds that more determined attackers may use to overcome these obstacles. This paper has demonstrated that IoT ransomware is a credible threat. We implemented a proof of concept tool that can compromise many IoT devices of varying types. We envisage that this work can be used to assist current and future IoT developers to improve the security of their devices, and also to help security researchers in implementing more effective ransomware countermeasures, including for IoT devices.

KEYWORDS

IoT, Ransomware, Malware, Bricking, Security

ACM Reference Format:
Calvin Brierley, Jamie Pont, Budi Arief, David J. Barnes, Julio Hernandez-Castro. 2020. PaperW8: An IoT Bricking Ransomware Proof of Concept. In *The 15th International Conference on Availability, Reliability and Security (ARES 2020), August 25–28, 2020, Virtual Event, Ireland*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3407023.3407044

1 INTRODUCTION

Ransomware is a form of malware that denies victims access to their resources until a payment is made [40, 62]. The Internet of Things (IoT) is a growing collection of internet-capable embedded devices, and has been a focus of malware authors in recent years [25, 48]. However, attackers have yet to realise *IoT ransomware* on a large scale, instead currently opting to use malware on infected devices to mine cryptocurrency[37, 50] or to perform denial of service attacks [15, 39].

IoT malware that performs attacks in the "background"—such as cryptojacking or denial of service attacks—generally does not severely impact the owner of the device and can go unnoticed for long periods of time. In contrast, it is explicitly necessary for ransomware to alert the victim to the fact that they have been attacked, in order to extract payment.

If attackers can create effective ransomware for IoT devices, this could be very damaging for both consumers and IoT manufacturers, regardless of the profitability of such ransomware. This observation is the main motivation behind our research. We aim *to investigate the viability of a ransomware-type attack on IoT devices*, while at the same time, we would like *to explore the extent of the damage that such an attack can cause*.

Contributions. The key contributions of our paper are:

- We present a novel IoT ransomware proof of concept that shows that carefully designed malware could be used to extort victims via threatening to permanently disable their devices.
- We demonstrate how this malware is compatible with a number of Linux-based IoT devices with varying purposes and functionalities, such as cameras and routers.
- We show how ransom notes can be displayed to the owners of IoT devices once their device has been infected.
- We prove that this is a realistic threat, by using a method to permanently disable the devices in a way that renders them virtually irreparable by an average user, but still allow an attacker to restore them remotely.
- Finally, we discuss possible mitigation and remediation methods for both IoT consumers and manufacturers.

The rest of the paper is organised as follows. Section 2 provides some background on traditional types of ransomware and of relevant IoT malware. Section 3 examines previous attempts and related work on creating IoT ransomware, along with the challenges that IoT devices present to ransomware authors. Section 4 outlines the steps we took to develop our proof of concept IoT ransomware called "PaperW8", as well as how it functions and its limitations. Section 5 shows the results of attacking multiple IoT devices with PaperW8. Section 6 discusses a number of potential countermeasures that developers can implement in order to dissuade or even prevent attackers from developing ransomware for their platform.

Section 7 outlines the limitations of our current research and provides recommendations for further work. Finally, Section 8 concludes our paper.

2 BACKGROUND

There is a wealth of research papers discussing IoT security issues. Investigation into ransomware—and malware in general—is also a very active area of research. However, limited academic analysis has been published on the subject of *IoT-based ransomware*. This section will provide background information on ransomware and general IoT malware, while a closer look into IoT ransomware will be given in Section 3.

2.1 Ransomware

Ransomware is a type of malware designed to extort a victim through the restriction or exposure of their valuable data [27]. This can take multiple forms, such as restricting access to assets through encryption [40], disabling features of a device [58], or threatening to release private information [11, 26]. For the negative consequences of the ransomware to be reversed or prevented, the victim must pay a ransom, normally through the use of cryptocurrency (such as Bitcoin or Monero), within a specified time frame [33, 45]. Consumers, manufacturers, companies and entire cities have been hit with ransomware attacks across the world; some examples are given below:

- The city of Florida had its water pump stations taken offline, forcing residents to make utility payments via mail. The council eventually sent $600,000 to attackers in an attempt to regain access to their critical systems [19]. Luckily, the decryption keys purchased from the attackers allowed them to successfully recover 90 percent of their data [71].
- The UK's National Health Service (NHS) was attacked by the "WannaCry" ransomware, which caused estimated damages totalling £92 Million [49, 57]. The ransomware also prevented access to patient records and caused the cancellation of approximately 19,494 medical appointments.
- NotPetya, a modified variant of the Petya ransomware family, paralysed several multinational companies by encrypting the master boot record of infected systems, resulting in an estimated $10 billion in damages [30].

One study of ransomware payments from 19,750 victims estimated that $16 million in Bitcoin was extracted over a two year period [33]. This—paired with the potential damages to devices, the reputation of companies, or in some cases human life—displays the real threat that ransomware poses to our society.

2.2 IoT Malware

The Internet of Things (IoT) is a network of embedded devices and sensors connected to the Internet, allowing them to communicate, interact and share information [17]. Some examples of IoT devices include smart TVs, cameras, wearables and cars. While it can be argued that any device that is capable of connecting to the Internet can be considered an "IoT device", in the context of this work we will consider an IoT device as *an Internet-connected device, with constrained capabilities, designed to perform a predefined function.*

There have been several strains of malicious software (malware) created to target IoT devices. Several examples are outlined below.

2.2.1 Mirai. Mirai is arguably the most famous IoT malware, especially as an IoT botnet that can cause massive damage through Distributed Denial of Service (DDoS) attacks [15]. Mirai was used to target DNS provider Dyn, rendering many popular websites such as Github and Twitter inaccessible [72]. It was also notable because it performed a 1 Tbps DDoS attack against French provider OVH [28]. With the release of the source code [35] on HackForums in September 2016 [14], several variants were quickly developed boasting additional exploits, extra functionality and a larger list of target devices [12, 60].

2.2.2 BrickerBot. BrickerBot is an IoT-based malware which is able to "brick" (i.e. damage a device in such a way that it no longer functions) vulnerable IoT devices [29]. Its author claims this was to prevent them from being added to criminal botnets such as Mirai. BrickerBot scanned for and exploited devices with known issues, then ran a list of shell commands which would render them inoperable by writing random data to flash memory, throttling services to near unusable levels and adding firewall rules that dropped all outgoing traffic [34, 65].

2.2.3 Silex. Silex is a more recent piece of IoT malware, which seems to have taken a similar route to BrickerBot by using shell commands to brick devices [22]. While Silex may be a new variant, some of the commands that were run have been directly lifted from the obfuscated BrickerBot source [34].

3 IOT RANSOMWARE AND RELATED WORK

As desktop-based ransomware and IoT botnets have both been successful on their respective platforms, the next logical step for attackers would be to attempt to emulate ransomware's success on IoT devices. However, due to the characteristics and constraints of IoT devices, the feasibility of IoT ransomware is limited by several factors, especially when compared to traditional "desktop" or even smartphone-based ransomware. Some attempts have been made but they had very limited success, as shown in the following sections.

3.1 Limitations of IoT

Discussed below are some challenges potential attackers would need to overcome for IoT ransomware to become a viable threat.

3.1.1 Asset Value. The success of current ransomware relies on the resource being ransomed having value to the victim, such as private information or irreplaceable documents. While this kind of file is commonly found on personal computers, IoT devices generally do not store this type of information. Most of their files are either user configuration settings or are responsible for running the system. These do not have much personal value to the user and are normally easy to replace by simply restarting or factory-resetting the device.

While there may be exceptions to the rule—such as IoT wearable health, location data, or stored photos—the lack of asset value would seriously limit the malware's scope to a small amount of devices with valuable data. The attacker would also need to create bespoke implementations for each device to process and store the obtained data for later use.

However, we must point out that the functionality of *any* device will have value to its user. Assuming an appropriate level of ransom demand, disabling access to this capability should therefore work universally, regardless of the type of device being attacked.

3.1.2 Persistence. The majority of IoT infections are not persistent by default, which is partially due to the nature of IoT storage. For the IoT devices we tested, the filesystem, kernel, configuration and bootloader partitions were stored in flash memory.

For a piece of malware to be persistent, the data stored in flash memory must be altered. The format and purpose of the stored data can vary on a device-to-device basis, as each is likely to have a different bootloader, Linux kernel and filesystem format. This diversity complicates the development of a universal and reliable method to modify flash memory without corrupting the existing firmware.

Due to this complexity, IoT malware rarely exhibits any form of persistence, which means that most can be removed by simply restarting the device. While there have been exceptions to this rule, such as with the IoT malware “torii” [42], or “Hide N’ Seek” [20, 44], the methods used by these to obtain persistence are not particularly sophisticated and rely on the changes made to the root filesystem through shell commands to persist after reboot, which will not work on all IoT devices. The devices used to test our proof of concept used read-only root filesystems, which makes them difficult to modify without completely re-flashing the partition with a new filesystem.

Mirai is an example of IoT malware which does not exhibit any form of persistence. As it does not interfere with the victim’s use of the device (other than the usual network overheads), it is hard for an average user to detect, and this will lower the likelihood of the host being restarted. Ransomware, however, is generally required to inform the user that they have been infected to collect a ransom [16], which will most likely result in the victim simply restarting the device. Surviving this restart is key for a IoT-ransomware to have any chance of success. Hence, for the ransomware to work effectively it will either need to be capable of obtaining persistence or become persistence-independent.

3.1.3 Communication. Ransomware requires a method of communication with the victim to demand a payment. For ransomware targeting personal computers such as WannaCry and Jigsaw, this is relatively simple, as there is a standard medium for communicating with users: the screen. A common approach is to display a window or image which includes information as to how to regain access to their lost resources. However, the nature of IoT devices is such that they often do not have a display, so the methods of communication will be heavily dependent on a device’s output medium.

For example, a Digital Video Recorder (DVR) would be expected to provide the user with a video feed via a connected screen, whereas a router simply delivers data it receives from the Internet. To effectively display a ransom note to the user, malware authors may need to be creative in order to communicate through each type of device in a way that the targeted user can easily find and understand.

3.1.4 Device Variation. IoT devices can vary a lot from one another, running on different hardware, architectures and operating systems. The tools and applications on the target device will also differ, as each IoT developer will likely only install what is fit for the device’s purpose. When creating IoT-based ransomware, an attacker will need to allow for this variation and still be able to hold devices to ransom effectively. If a ransomware is only compatible with a small number of devices, this could seriously limit its potential profit.

3.2 Known IoT Ransomware Implementations

There have been several previous attempts to implement ransomware on various IoT devices. Researchers at Avast demonstrated that an attacker could compromise a smart coffee machine, allowing them to display ransomware notes, boil water and run the grinder without any user interaction [32].

An LG smart TV was disabled by the Android-based malware “CYBER.POLICE” [23, 41], which eventually required LG support to provide an alternative method to perform a factory reset, which the victims could use to successfully restore the device.

At DefCon 24, PenTestPartners demonstrated their process to infect an IoT thermostat with ransomware by modifying the existing firmware [67]. The malware would allow attackers to show a simple ransomware message, lock out the victim and modify the temperature in their home.

These examples highlight the possibility for creative approaches in ransomware when attackers gain access to IoT devices that manage real-world resources, but these attempts have some limitations.

3.3 Limitations of Previous Implementations

While earlier approaches that target consumer devices are valid examples of ransomware that could potentially be profitable, they have issues that may prevent attackers from exploiting them in a practical setting.

- The previous Linux-based proofs of concept have creative methods for locking down and ransoming the user. However, this creativity comes with the burden of only working for that particular model or brand, which greatly limits the potential affected devices and therefore the total likely payout. In order to work on other devices, the ransomware may need heavy modification or to be entirely rewritten.
- Each example also seems to require *persistence*. The thermometer seems to use a JFFS2 filesystem which allows for live modification, while the coffee maker required firmware replacement. These methods may not be possible for other IoT devices, or require significant work to be made compatible. Additionally, the developers did not state whether the changes that they made would survive a factory reset.

In summary, in order to solve or circumvent the limitations described in this section, and create effective ransomware, an attacker has to design the malware to be generalisable—such that they could reasonably infect many different devices with limited modifications—and to provide the malware with a method to gain persistence, or ransom the devices in such a way that persistence is not required.

4 PAPERW8

We have created a proof of concept ransomware named “PaperW8”[1], which can be used to attack exploitable Linux-based IoT devices.

[1]In the simplest form, we wanted to remove all functionality from the affected device, turning it into just a “paper weight”, hence the name.

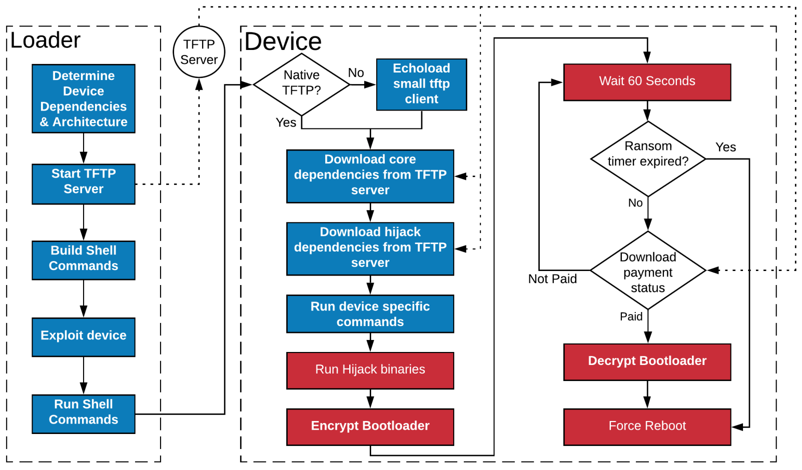
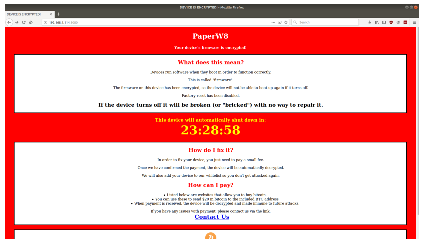


Figure 1: PaperW8 Attack Structure


DEVICE IS ENCRYPTED! - Mozilla Firefox

192.168.1.114:8080

PaperW8

Your device's firmware is encrypted!

What does this mean?

Devices run software when they boot in order to function correctly.

This is called "firmware".

The firmware on this device has been encrypted, so the device will not be able to boot up again if it turns off.

Factory reset has been disabled.

If the device turns off it will be broken (or "bricked") with no way to repair it.

This device will automatically shut down in:

23:28:58

How do I fix it?

In order to fix your device, you just need to pay a small fee.

Once we have confirmed the payment, the device will be automatically decrypted.

We will also add your device to our whitelist so you don't get attacked again.

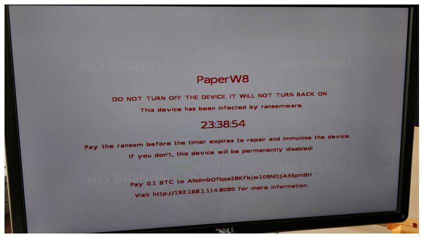
How can I pay?

- Listed below are websites that allow you to buy bitcoin.
- You can use these to send $20 in bitcoin to the included BTC address
- When payment is received, the device will be decrypted and made immune to future attacks.

If you have any issues with payment, please contact us via the link.

Contact Us

(a) Webserver-based


PaperW8

DO NOT TURN OFF THE DEVICE. IT WILL NOT TURN BACK ON

This device has been infected by ransomware.

23:38:54

Pay the ransom before the timer expires to repair and immunise the device.

If you don't, this device will be permanently disabled!

Pay 0.1 BTC to ANdmbQfbpa28KFbjw109NDjiA55pmBH

Visit http://192.168.1.114:8080 for more information.

(b) Framebuffer-based

Figure 2: Ransom Notes

Through this proof of concept, we want to identify roadblocks for attackers when developing ransomware, and then create and test countermeasures for helping security researchers, developers and end-users. In this section, we will describe the structure of PaperW8, how it operates and how we overcame the previously described limitations.

4.1 Structure

Infecting a device with PaperW8 involves multiple steps, including loading dependencies, hijacking communication channels and disabling the device. An overview of how the PaperW8 attack is structured is shown in Figure 1.

PaperW8 was designed with adaptability and generalisation in mind. While previous attempts at IoT-based ransomware target specific models and brands, PaperW8 aims to work on many different devices, such that as few modifications as possible would be necessary to extend the malware to new devices. We do this by limiting the number of requirements needed in order to ransom a device and providing "generic" communication modules, such that PaperW8 can hijack common communication channels to deliver an effective ransomware message.

4.2 Exploitation and Infection

To infect a device, PaperW8 needs to be able to run shell commands on the target device. This can be achieved in several ways, including through access via an exposed telnet port with weak authentication, or by utilising an existing exploit, such as command injection due to unsanitised input.

Once PaperW8 has gained access, its next aim is to upload its dependencies. First, the loader starts a Trivial File Transfer Protocol (TFTP) server on the attacking host; devices PaperW8 attacks will be forced to download the required dependencies into memory via the `tftp` utility, as it is often included on IoT devices.

If this utility is not included on the device, PaperW8 can instead use a technique known as "echoloading" to upload it. If there is an "echo" utility present on the device, PaperW8 can reconstruct a tftp client by splitting its binary into 50-byte hexadecimal encoded "chunks", then using the `echo` command on the remote host. We found a previous implementation of echoloading used to load cryptomining malware onto Internet-connected DVRs [36, 68], which we modified for our use. PaperW8 can then use the uploaded TFTP client to download components to the device.

After the dependencies have been uploaded, PaperW8 can kill any services that communicate with the user, to prevent the device from being used. PaperW8 will also kill any vulnerable services that were used to exploit the device, such that neither the user nor other attackers will be able to use the same vulnerability to regain access. Finally, PaperW8 will execute the uploaded binaries, taking full control of the device.

4.3 Permanent Denial of Service

The Memory Technology Device (MTD) subsystem is an "abstraction layer for raw flash devices" [3], and is commonly used in Linux-based IoT devices to interact with various forms of flash storage. Linux can use this to manage various partitions of data on attached flash devices which store essential components that run the device effectively, such as the bootloader, filesystem or application configurations. We chose to use the MTD subsystem to make persistent edits to our victim IoT devices, as it has been present in many of the devices that we have encountered, it is somewhat standardised and has been used by previous IoT bricking malware.

In our proof of concept, PaperW8 reads the bootloader partition into a buffer, encrypts it using AES-256-CBC, then writes it back to the same partition, which will mangle the bootloader such that it will fail to boot if the device is restarted. When using AES, the Initialisation Vector (IV) and the key are hard-coded for testing purposes, but it is reasonable to expect that these could be generated, stored and managed by a Command and Control (C&C) server in a real setting.

This approach is somewhat similar to those taken by some desktop-based ransomware which modify the Master Boot Record (MBR) in Windows operating systems, such as *Seftad* [38, 47] and *Petya* [9]. While we may not have achieved a persistent infection, we have managed to implement a "persistence-independent" extortion method, in that any attempts to recover the device by restarting it will result in it no longer functioning.

At this point, PaperW8 will have taken full control of the device, while restricting the victim's access. The device can be easily recovered by PaperW8, as it is aware of the method of decryption and keys, and will retain the ability to modify the flash memory until the device is reset. The victim, however, is unlikely to have any reasonable alternative, thus leaving paying the ransom as the only easy method of recovery.

4.4 Communication Hijacking

IoT devices communicate with their user in many ways. For each IoT device PaperW8 infects, PaperW8 should aim to "hijack" these methods of communication to send a ransom note. This allows the attacker to communicate with the victim while also restricting access to any of the existing functionality.

To maintain compatibility with the maximum number of devices, we provide optional extensions that can be used alongside PaperW8's normal behaviour to hijack the most common methods of communication used by IoT devices, which are shown below.

4.4.1 HTTP. The most common method used by the devices we tested was an HTTP server. This provides an easy method for users to interact with the device by visiting its IP address via a browser.

As part of infecting the device, PaperW8 uploads a compressed archive containing a webpage with a ransom note and a custom version of "Busybox". Busybox is a tool that provides many common Linux utilities in a single small executable [13]; our version includes a number of useful tools for the infection process, such as `httpd`, an HTTP daemon which can be used to host webpages. PaperW8 then kills and replaces the original web service by hosting the ransom note using `httpd` on the vacant port. Whenever the user attempts to connect to the device through the browser, they will instead be greeted with the ransom note, as shown in Figure 2a.

4.4.2 DNS. Routers sometimes provide DNS services to clients that connect via DHCP. By killing and relaunching the DNS service, PaperW8 can configure it to redirect all HTTP requests to the IP of the router, creating a malicious captive portal. If PaperW8 also hijacks the HTTP server, whenever the user attempts to access a webpage via HTTP, they will instead be shown the ransom note.

4.4.3 Framebuffer. Some IoT devices are designed to use a Graphical User Interface (GUI) as a method of communication. On the Linux operating system, the "framebuffer" can be used to modify output to attached displays. By creating a small application that implements a modified version of LittlevGL [4], an open-source library for building GUIs, PaperW8 can communicate with a Linux framebuffer device to display a ransom note. This note will include a simple description, a countdown timer and a URL to the hijacked webserver, where the victim can find a larger ransom note with further information, as shown in Figure 2b.

4.5 Recovery

As part of the infection process, PaperW8 aims to shut down the vulnerable services that initially gave us access. If services are shut down, the user should not be able to gain the same level of access to the running system as the attacker. At this point, it would be very difficult for a user to regain usage of the device.

As updating the bootloader of a device is generally not advised [63, 64, 69, 73], this feature is unlikely to be included in the factory reset process. If a reset is attempted, it might not fix a bricked device, as the bootloader will remain unchanged.

However, if the victim has access to the required tools, recovery could be attempted by using an external programmer to manually reprogram the flash chip with the original bootloader. This could be achieved by identifying and using an on-board debug interface, communicating with the storage device directly by desoldering it from the board, or using a test clip. This would also require the victim to either decrypt the bootloader, or somehow obtain the original bootloader.

The diversity of IoT devices also makes things harder in this case, as finding a solution for the victim would require identifying debug interfaces, storage locations or the correct bootloader for each attacked device and model. This level of expertise is unrealistic to expect from an average user, and in the absence of specialised knowledge or equipment, they will most likely perceive that paying ransom to the attacker is the only viable recovery option.

After the partition has been encrypted and the ransomware message has been displayed, the victim will have a limited time to make a payment to the attacker. A C&C server is typically used by botnet owners in order to manage a large number of infected devices. For this proof of concept, we have implemented a simplified pseudo C&C server to simulate payment and confirmation for testing purposes. In a real world setting, a more secure method of key management and confirmation would be used. Using a bash script, PaperW8 will periodically check for a predetermined value on our attacking host via TFTP to see if the victim has "paid". If the expected value is found, PaperW8 will decrypt the bootloader and reboot the device. If, however, the victim does not make a payment in time, PaperW8 can force the device to reboot, and—as a consequence—brick it.

5 TESTED DEVICES AND RESULTS

We tested our malware against a collection of devices with different purposes and from different brands. We chose these devices as each of these devices were vulnerable such that an attacker could run shell commands and have previously been targeted by IoT malware. While the creation of the exploits is out of scope of this paper, we are confident in the assumption that similar vulnerabilities will continue to exist and will be found in future devices.

For each device, we will give a general overview and explain how it could be exploited. We will then show how we implemented each of the aforementioned techniques to perform an effective ransomware attack. A summary of the devices we tested and the results from our tests are shown in Table 1.

5.1 HG532 TalkTalk Router

The HG532 is a popular router built by Huawei and sold in a number of countries. Our test device was distributed in the UK by "Talk-Talk", an Internet Service Provider. The router was found to be vulnerable to a remote code execution attack through the UPnP implementation on port 37215 [54]. We adapted exploit code that was readily available online for use with our ransomware loader, which we then used to gain control of our test device [7].

5.1.1 Communication Hijacking. Users interact with the device either actively through the configuration page or passively by simply browsing the internet. To hijack these communication channels, PaperW8 forced the device to to download the HTTP hijacking dependencies. It then killed the processes which were responsible for serving the router's vulnerable UPnP and web services and changed the DNS server configuration to redirect all HTTP requests to the router, creating a malicious captive portal. Finally, it started an `httpd` server to host the ransom note on port 80, the default HTTP port. Any user that attempted to make an HTTP request while connected to the router would now be redirected to the ransom note, an example of which can be seen in Figure 2a.

5.1.2 Device Bricking. The router splits its flash chip into a number of different partitions, which are listed in the file `/proc/mtd`. The layout is shown in Figure 3. In our case, we chose to encrypt the bootloader partition, `mtd0`. We then reset the device, which failed to boot. The only indication that the device was powered was a flashing LED.

We were able to activate the factory reset process, but the device was unable to recover. We believe that this was due to the device only resetting partitions that are frequently changed, as they are more likely to become corrupted.

After bricking the device, we wanted to confirm that the encryption had worked as intended. To test this, we desoldered the flash chip (MX29LV320ETTI-70G [46]) and used a flash programmer to manually read the contents. As expected, the bootloader was encrypted using the correct scheme. After decryption, we were able to get the original data, proving that successful recovery is possible.

5.2 R6250 Netgear Router

Several routers produced by Netgear, including the R6250, were found to be vulnerable to command injection via the web server [43, 53]. There are around 855 R6250 devices exposed to the internet are listed on Shodan, while the total number of routers potentially vulnerable to this exploit is approximately 19,470.

5.2.1 Communication Hijacking. This router used the same communication channels as the HG532, and we used a similar approach to infect it. We exploited this device through a command injection vulnerability in the webserver, which we used to download and run a shell script; this allowed us to hijack the webserver and display our ransom note. We also managed to hijack the DNS server to redirect HTTP requests to the router without any modifications.

5.2.2 Device Bricking. The router had 18 different partitions, including `mtd0`, which contained the bootloader. This was our initial target. However, the developers had set the partition to be read-only when booting. Any attempts to set the partition to be writeable were unsuccessful, so we instead targeted `mtd14`, which contains the Linux kernel and root filesystem. After encrypting this partition, we rebooted the device and found it unresponsive.

Attempting to factory reset the device as described in the user manual was unsuccessful [51]. However, due to the bootloader still being intact, we found a method to recover the device using tftp that, while not mentioned in the official manual, could be feasibly achieved by a user without any specialist tools [52].

5.3 TV-7104HE MVPower Digital Video Recorder (DVR)

Internet-connected DVRs allow users to access multiple connected cameras remotely. At the time of writing, Shodan lists approximately 94,000 MVPower devices that are publicly accessible [10]. The exploit we used leveraged an undocumented "feature" which allows an unauthenticated user to access a root shell through HTTP GET parameters on the `/shell` page [66].

5.3.1 Communication Hijacking. We identified the main communication channels as the configuration web server and the Graphical User Interface (GUI). The GUI displayed the current state of the device and the live feed of any connected cameras on attached monitors, which users could interact with using a USB mouse and keyboard.

The web server was hijacked in a similar manner to the previous devices. Unfortunately, due to the method of exploitation, killing the web service to replace it would prevent the exploit from finishing.

Table 1: Result of Tested Devices

Device Name	Exploitation			Hijacking Method			Disabling	
	Architecture	Exploit	Number of Exposed Devices[2]	Web-server	DNS	Frame-buffer	Bricked	Failed Factory Reset
HG532 Router	Mipsel	CVE-2017-17215	Unknown	✓	✓	N/A	✓	✓
R6250 Router	Arm	CVE-2016-6277	850[4]	✓	✓	N/A	✓	Partial
MVPower DVR	Armv5l	Backdoor Shell [66]	94,171	✓	✗	✓	✓	✓
WiPG-1000	Armv5l	CVE-2019-3929	Unknown	✓	✗	✓	✓	✓
932L Camera	Mipsel	CVE-2019-10999	90,359	✓	✗	N/A	✓	✓
5020L Camera	Mipsel	CVE-2019-10999	73,533[3]	✓	✗	N/A	✓	✓

2 The number of devices that were publicly exposed to the internet, checked via Shodan on the 9th December 2019.
3 Checked via Shodan on the 18th February 2020.
4 Checked via Shodan on the 26th March 2020.

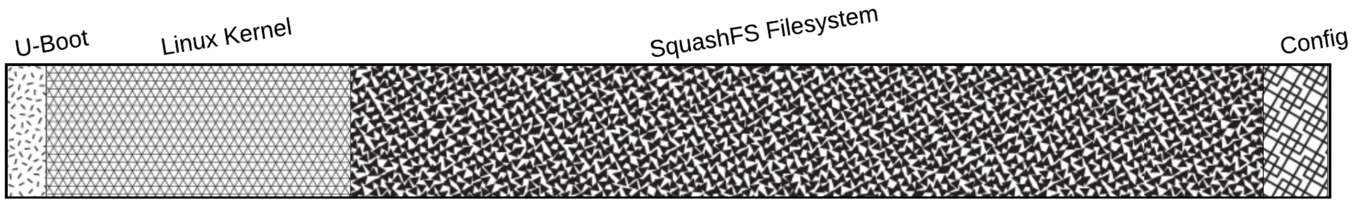


Figure 3: Layout of MTD partitions in HG532's flash memory

As the default HTTP port was being populated by the web server, we instead hosted the ransom note on port 8080.

We then moved on to hijack the GUI, we killed any processes that manage the interface, preventing it from being able to respond to any user input. We then used the `framebuffer` hijacking module to communicate with the Linux framebuffer device at `/dev/fb0` to display a ransom note on any attached monitors. We then ran the BusyBox `watchdog` utility to continually report that everything was running correctly, in order to avoid any automatic reboots.

Finally, we killed the `dvr_app` process, which prevented any further access to the vulnerability we used to exploit the device.

5.3.2 Device Bricking. The DVR's `/proc/mtd` is not as informative, containing only letters as partition names. We identified the usage of each partition by extracting and analysing each section with Binwalk [59] and decided to encrypt the `mtd0` partition, which contained a U-Boot bootloader [2]. Attempts to reboot the device were unsuccessful, and while the LEDs did show signs of activity, no output was displayed and no attempts were made to connect to the router.

We believe that the factory reset function only covered resetting the user configuration of the device, it was also only be accessible via the device's GUI. After we bricked the device, this was no longer available, preventing any reset attempts.

5.4 WiPG-1000 Presenter

The WiPG-1000 is a presenter produced by "WePresent". Users can access it via the local network to stream presentations to connected screens or projectors. It is however, expensive, with some vendors selling it for up to £399 [5] and the 1600W model (which was also found to be vulnerable) for up to £1,029 [1]. We managed to exploit this device using a previously disclosed method involving a command injection vulnerability in the web interface [18, 56].

5.4.1 Communication Hijacking. While the main method of communication for the presenter is through any connected screens, the WiPG-1000 also provides a webserver. We were able to use PaperW8 to hijack both with almost no modification.

After gaining access to the device, we first killed the applications which were responsible for running the webserver and GUI. We then ran the webserver and framebuffer modules in PaperW8 to display our ransom notes via a webpage and any connected screens.

5.4.2 Device Bricking. The presenter had 11 partitions. For this device, we targeted `mtd10`, which contained a U-Boot bootloader. After successfully encrypting the partition, we attempted to restart the device. Upon rebooting, the button normally used to activate the device was unresponsive, and there was no output to the attached screen or any connections made to the router. Attempts to factory reset the device via the reset button also proved unsuccessful.

5.5 5020L and 932L D-Link Cameras

These IP cameras can be used indoors or outdoors and can be accessed at its IP address through a web browser, or from anywhere via the "mydlink" platform [24]. A 2016 report from Shodan lists the 932L model as the most popular publicly accessible D-Link product and the 5020L as the sixth most popular [6]. We made use of a known buffer overflow vulnerability [55] that was shown to affect several devices which use the `alphapd` web server, including both of these models. Buffer overflow vulnerabilities allow attackers to overwrite areas in memory, such as critical application status information. While there was proof of concept exploit source code made available [70], this only covered certain versions of the 5020L and 930L (not 932L) models. While this allowed us to exploit the 5020L model relatively easily, we needed to modify the exploit to be compatible with the 932L model.

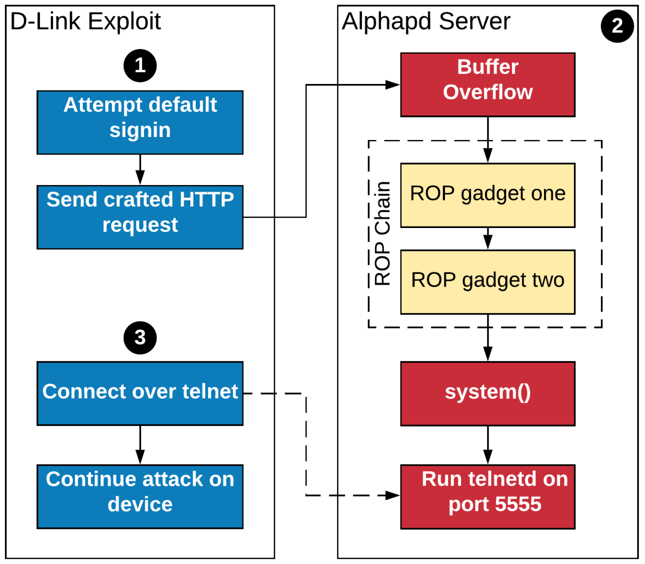


Figure 4: CVE-2019-10999 Exploit Structure

The exploit allows us to overwrite the "return address", which defines the next instruction to execute when the current function returns. This allows an attacker to choose the address of the next instruction to execute. This is still somewhat limiting as the attacker can only use instructions that are already available as part of the application (unless the attacker is able to inject their own instructions using shellcode).

To circumvent this limitation, attackers can make use of "Return Oriented Programming" (ROP) [61]. By identifying small sections of existing assembly which end in a return instruction (known as "gadgets"), attackers can perform complex actions by constructing "ROP chains". By "returning" to each gadget in the chain, attackers can implement actions by combining code that was already available in the binary. We were able to exploit the D-Link camera via a crafted HTTP request and start a telnetd daemon, giving us access to a remote shell with root privileges. An overview of the exploit process is shown in Figure 4.

A limitation of this exploit is that the attacker must have access to an authenticated session on the camera's vulnerable web application. In this instance, we assume the victim has not changed the default admin password, which is blank by default.

5.5.1 Communication Hijacking. The cameras could be viewed and configured through a webpage hosted on port 80, which serves the camera feed. As the exploit used crashes the webserver after completion, PaperW8 could replace the webserver with the ransomware message in a similar manner to the previous devices.

5.5.2 Device Bricking. Both cameras have several partitions which are helpfully labelled in `/proc/mtd`. They are identically named and structured, but differ slightly in size and content. We decided to encrypt the `mtd1` partition, which contained the bootloader on both devices. We then forced the devices to reboot, which seemed to sufficiently disable them. After rebooting, the LEDs would no longer light and any attempts to access the website or factory reset the device did not succeed.

5.6 Summary

Each device we tested, despite being manufactured and configured independently for very different purposes, was vulnerable to communication hijacking. In each case, a webserver was the most reliable method to present a ransom note, as it was a communication method used by each device and allowed us to provide the most information. We also found that almost all the tested devices were vulnerable to a permanent denial of service attack and that the factory reset functionality provided by the vendors was ineffective when attempting to recover.

6 COUNTERMEASURES

There are several countermeasures that we believe would be very effective in preventing our proof of concept attack from succeeding. Possible ideas that we would like to suggest include:

- **Making partitions read-only.** If developers do not need to update certain partitions (such as the bootloader), forcing them to be read-only could prevent attackers from encrypting important static information. Technology such as eMMC provides a "Write Protect" feature that allows manufacturers to permanently disable writing to certain partitions at the hardware level [74]. However, other aspects such as extra cost, compatibility, and efficiency are key to large scale embedded device design and must also be considered.
- **Restoring the bootloader through factory reset.** If the bootloader could also be restored, it should allow the victim to fully recover the device after partitions are encrypted. While they may lose their configuration settings, this will allow them to recover the functionality and (if possible) update the device to prevent further exploitation. However, this may require that the factory reset functionality works independently of the main system.
- **Implementing the principle of "least privilege".** In each of our examples, we needed to gain access to the MTD devices in the `/dev` directory to perform damaging changes. As all of the exploits that we used allowed us to gain root access, we were able to modify partitions easily. By running applications at a lower privilege level and requiring root level access only to modify any system essential partitions, attackers could be prevented from making device-damaging changes. After exploiting the device, attackers would ideally only have the application's level of access and would only be able to modify non-essential partitions, such as the device configuration settings.
- **Device updates.** Users should update their devices regularly to patch known vulnerabilities, preventing malware from gaining access.

7 LIMITATIONS AND FURTHER WORK

While we have successfully demonstrated the feasibility of our approach in creating IoT-based ransomware, some limitations pose interesting open challenges to be addressed in the future.

7.1 Limitations

7.1.1 Device Cost and Ransom Pricing. In traditional ransomware, the amount of the ransom demand can vary drastically. This can be partly attributed to the worth of the assets encrypted being highly subjective. Price can also be impacted by the income and willingness to pay criminals [21, 31]. In PaperW8's case however, we disable the functionality of the IoT device, therefore the ransom price has an objective upper bound: the retail price of a suitable replacement plus the cost of the disruption caused (e.g. to business) and the inconvenience of replacing it. If the ransom greatly exceeds the price of a "new" device of the same type and the associated costs, victims may refuse to pay and instead simply buy another [41], and possibly shift to another brand entirely.

However, while IoT devices are assumed to be comparatively cheap and replaceable when compared to desktops or data, this may not always be the case. Device replacement may also be out of budget for the user, depending on their income or the price and availability of the device. A smart TV, for example, may be deemed too expensive to replace for some users. Additionally, if a large number of devices of a particular brand are infected at the same time, the device's developer may not have the stock or resources to provide replacements or repairs to affected customers. Losses in revenue or production due to a device becoming unavailable may also influence a victim's likelihood of payment, especially if we consider the time needed for a replacement to arrive.

7.1.2 Premature Rebooting. When victims are shown the ransom note, they may panic or simply ignore the ransom note, and reboot the device. As PaperW8 will no longer be able to recover the device, there will be no motivation to pay the ransom, which may limit the effectiveness of the ransomware.

Future malware could modify the filesystem so only the functionality of the device is disabled when booted. This would allow the user to reboot, but would require gaining persistence for each targeted device, which will increase the complexity of development. It may also be easier to recover the device via flash memory editing or via factory reset.

7.1.3 HTTPS Warnings. When using the DNS hijacking module, if a victim visits websites using HTTPS, it will instead display a warning as the modified website that we return is not what is expected. This may cause the victim to not see the ransomware message and instead assume that the router is faulty. If the victim restarts the router, it will be bricked and the attacker will not be able to extract payment. This could potentially be circumvented, as demonstrated by VPNFilter's `ssler` module [8] which downgrades HTTPS requests to insecure HTTP requests, if not enforced by the remote host.

7.2 Further Work

7.2.1 Monetization Options. Alternative methods to monetise IoT ransomware could be investigated. For example, attackers could focus on the companies that provide the product, pressuring them to pay on behalf of their customers or receive severe backlash from victims.

If part of a large IoT ransomware campaign, it is within reason that the targeted devices will not have readily available stocks to replace the compromised ones. This could put the manufacturer under extra pressure, and customers may be compelled to switch to a competitor to continue operation.

7.2.2 Cyberweapon Potential. The potential disruption caused by a very large IoT ransomware-like campaign to critical infrastructures and companies cannot be overstated, particularly if the ransomware has some worm capabilities, to the point that techniques not too dissimilar to the ones we highlight in this work could form the basis of cyberweapons acting under the guise of a ransomware attack, with no ulterior economic profitability in mind.

7.2.3 Privacy Invasion. IoT devices provide a number of sensors that are useful to their users. These sensors could potentially be used by attackers to invade the privacy of their victims. If attackers were able to obtain sensitive information from a vulnerable device, such as images via a camera or a user's internet history through a router, they would no longer be upper bounded by the device's worth when demanding a ransom and could instead request much larger amounts.

8 CONCLUSIONS

In this work, we investigated the viability of ransomware on IoT devices. To demonstrate how an attacker may overcome the various limitations presented by IoT devices, such as the lack of easy means of communication with the victim, we proposed PaperW8, a bricking ransomware that hijacks existing communication methods to provide a ransom note to the victim. We tested PaperW8 against six IoT devices of different brands and purposes to successfully demonstrate the viability of our attack. We were able to hold them to ransom and prove that the device could be bricked in a way to prevent easy recovery. We believe PaperW8's highly destructive nature makes it a valid method of ransoming Linux-based IoT devices and could be adapted by attackers to target other new devices if they are found to be vulnerable. In addition, we proposed several countermeasures to avoid similar attacks, and we suggest that they should be investigated and quickly deployed in order to prevent attackers from using this powerful and damaging attack strategy in the near future.

ACKNOWLEDGEMENT

The work was partly supported by the UK EPSRC EP/P011772/1 project on the EconoMical, PsycHologicAl and Societal Impact of RanSomware (EMPHASIS).

REFERENCES

[1] [n.d.]. Barco wePresent WiPG-1600W wireless presentation system Desktop HDMI + VGA (D-Sub). https://www.amazon.co.uk/wePresent-WiPG-1600W-wireless-presentation-Desktop/dp/B076T41MVC [Accessed: April 2020].
[2] [n.d.]. Das U-Boot – the Universal Boot Loader. https://www.denx.de/wiki/U-Boot [Accessed: April 2020].
[3] [n.d.]. General MTD documentation. http://www.linux-mtd.infradead.org/doc/general.html [Accessed: April 2020].
[4] [n.d.]. LittlevGL. https://littlevgl.com/ [Accessed: April 2020].
[5] [n.d.]. WePresent WIPG-1000-P. https://www.ballicom.co.uk/wepresent-r9866100eu-.p1408805.html [Accessed: April 2020].
[6] 2016. D-Link Internet Report. https://dlink-report.shodan.io/ [Accessed: April 2020].
[7] 2017. Huawei Router HG532 - Arbitrary Command Execution. https://www.exploit-db.com/exploits/43414 [Accessed: April 2020].
[8] 2018. https://blog.talosintelligence.com/2018/06/vpnfilter-update.html [Accessed: April 2020].
[9] 2019. https://www.fortinet.com/blog/threat-research/petya-s-master-boot-record-infection.html [Accessed: April 2020].
[10] 2019. Shodan MVPower DVR Search. https://www.shodan.io/search?query=JAWS%2F1.0 [Accessed: April 2020].
[11] Lawrence Abrams. 2020. BitPyLock Ransomware Now Threatens to Publish Stolen Data. https://www.bleepingcomputer.com/news/security/bitpylock-ransomware-now-threatens-to-publish-stolen-data/ [Accessed: April 2020].
[12] Akamai. 2017. Satori Mirai Variant Alert. https://www.akamai.com/uk/en/multimedia/documents/state-of-the-internet/satori-mirai-variant-alert-threat-advisory.pdf [Accessed: April 2020].
[13] Erik Andersen. [n.d.]. BusyBox: The Swiss Army Knife of Embedded Linux. https://busybox.net/about.html [Accessed: April 2020].
[14] Anna-senpai. 2017. [FREE] World's Largest Net:Mirai Botnet, Client, Echo Loader, CNC source code release. https://hackforums.net/showthread.php?tid=5420472 [Accessed: April 2020].
[15] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. 2017. Understanding the mirai botnet. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*. 1093–1110.
[16] Budi Arief, Andy Periam, Orcun Cetin, and Julio Hernandez-Castro. 2020. Using Eyetracker to Find Ways to Mitigate Ransomware. In *Proceedings of the 6th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP,*. INSTICC, SciTePress, 448–456. https://doi.org/10.5220/0008956004480456
[17] Sachin Babar, Antonietta Stango, Neeli Prasad, Jaydip Sen, and Ramjee Prasad. 2011. Proposed embedded security framework for internet of things (iot). In *2011 2nd International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE)*. IEEE, 1–5.
[18] Jacob Baines. 2019. Crestron AM/Barco wePresent WiPG/Extron ShareLink/Teq AV IT/SHARP PN-L703WA/Optoma WPS-Pro/Blackbox HD WPS/InFocus LiteShow - Remote Command Injection. https://www.exploit-db.com/exploits/46786 [Accessed: April 2020].
[19] BBC. 2019. Florida town pays $600,000 virus ransom. https://www.bbc.co.uk/news/technology-48704612 [Accessed: April 2020].
[20] Bogdan Botezatu. 2018. Hide and Seek IoT Botnet resurfaces with new tricks, persistence. https://labs.bitdefender.com/2018/05/hide-and-seek-iot-botnet-resurfaces-with-new-tricks-persistence/ [Accessed: April 2020].

[21] Edward Cartwright, Julio Hernandez Castro, and Anna Cartwright. 2019. To pay or not: game theoretic models of ransomware. *Journal of Cybersecurity* 5, 1 (2019), tyz009.

[22] Larry Cashdollar. 2019. SIRT Advisory: Silexbot bricking systems with known default login credentials. https://blogs.akamai.com/sitr/2019/06/sirt-advisory-silexbot-bricking-systems-with-known-default-login-credentials.html [Accessed: April 2020].

[23] Catalin Cimpanu. 2016. Android Ransomware Infects LG Smart TV. https://www.bleepingcomputer.com/news/security/android-ransomware-infects-lg-smart-tv/ [Accessed: April 2020].

[24] D-Link. [n.d.]. mydlink. https://www.mydlink.com/ [Accessed: June 2020].

[25] Sergiu Gatlan. 2019. IoT Attacks Escalating with a 217.5% Increase in Volume. https://www.bleepingcomputer.com/news/security/iot-attacks-escalating-with-a-2175-percent-increase-in-volume/ [Accessed: April 2020].

[26] Sergiu Gatlan. 2019. Maze Ransomware Demands $6 Million Ransom From Southwire. https://www.bleepingcomputer.com/news/security/maze-ransomware-demands-6-million-ransom-from-southwire/ [Accessed: April 2020].

[27] Alexandre Gazet. 2010. Comparative analysis of various ransomware virii. *Journal in computer virology* 6, 1 (2010), 77–90.

[28] Dan Goodin. 2016. Record-breaking DDoS reportedly delivered by >145k hacked cameras. https://arstechnica.com/information-technology/2016/09/botnet-of-145k-cameras-reportedly-deliver-internets-biggest-ddos-ever/ [Accessed: April 2020].

[29] Dan Goodin. 2017. BrickerBot, the permanent denial-of-service botnet, is back with a vengeance. https://arstechnica.com/information-technology/2017/04/brickerbot-the-permanent-denial-of-service-botnet-is-back-with-a-vengeance/ [Accessed: April 2020].

[30] Andy Greenberg. 2018. The Untold Story of NotPetya, the Most Devastating Cyberattack in History. https://www.wired.com/story/notpetya-cyberattack-ukraine-russia-code-crashed-the-world/ [Accessed: April 2020].

[31] Julio Hernandez-Castro, Edward Cartwright, and Anna Stepanova. 2017. Economic analysis of ransomware. *Available at SSRN 2937641* (2017).

[32] Martin Hron. 2019. The Internet of Thing: How a single coffee maker's vulnerabilities symbolize a world of IoT risks. https://blog.avast.com/avast-hacked-a-smart-coffee-maker [Accessed: April 2020].

[33] Danny Yuxing Huang, Maxwell Matthaios Aliapoulios, Vector Guo Li, Luca Invernizzi, Elie Bursztein, Kylie McRoberts, Jonathan Levin, Kirill Levchenko, Alex C Snoeren, and Damon McCoy. 2018. Tracking ransomware end-to-end. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 618–631.

[34] Janit0r. 2017. BrickerBot Source. https://github.com/JeremyNGalloway/mod_plaintext.py/blob/master/mod_plaintext.py [Accessed: April 2020].

[35] Paras Jha. 2017. Mirai Source. https://github.com/jgamblin/Mirai-Source-Code [Accessed: April 2020].

[36] "kalocpzoep". 2014. https://twitter.com/a3019397/status/463349646073794561 [Accessed: April 2020].

[37] Simon Kenin. 2018. Mass MikroTik Router Infection – First we cryptojack Brazil, then we take the World? https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/mass-mikrotik-router-infection-first-we-cryptojack-brazil-then-we-take-the-world/ [Accessed: April 2020].

[38] Amin Kharraz, William Robertson, Davide Balzarotti, Leyla Bilge, and Engin Kirda. 2015. Cutting the gordian knot: A look under the hood of ransomware attacks. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 3–24.

[39] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. 2017. DDoS in the IoT: Mirai and other botnets. *Computer* 50, 7 (2017), 80–84.

[40] Eugene Kolodenker, William Koch, Gianluca Stringhini, and Manuel Egele. 2017. PayBreak: defense against cryptographic ransomware. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. 599–611.

[41] Michael Koopman. 2017. Preventing Ransomware on the Internet of Things. (2017). https://pdfs.semanticscholar.org/419f/8d22ba3eea1f4d9fbcd6b3b68d294504d276.pdf [Accessed: April 2020].

[42] Jakub Kroustek, Vladislav Iliushin, Anna Shirokova, Jan Neduchal, and Martin Hron. 2018. Torii botnet - Not another Mirai variant. https://blog.avast.com/new-torii-botnet-threat-research [Accessed: April 2020].

[43] Joel Land. 2016. Multiple Netgear routers are vulnerable to arbitrary command injection. https://www.kb.cert.org/vuls/id/582384/ [Accessed: April 2020].

[44] William Largent. 2018. New VPNFilter malware targets at least 500K networking devices worldwide. https://blog.talosintelligence.com/2018/05/VPNFilter.html [Accessed: April 2020].

[45] Kevin Liao, Ziming Zhao, Adam Doupé, and Gail-Joon Ahn. 2016. Behind closed doors: measurement and analysis of CryptoLocker ransoms in Bitcoin. In *2016 APWG Symposium on Electronic Crime Research (eCrime)*. IEEE, 1–13.

[46] Macronix. 2013. MX29LV320E T/B DATASHEET. https://www.macronix.com/Lists/Datasheet/Attachments/7647/MX29LV320E%20T-B%2\c%203V%2c%2032Mb%2c%20v1.3.pdf [Accessed: April 2020].

[47] Denis Maslennikov. 2010. New Seftad Ransomware Attacks Master Boot Record. https://securelist.com/and-now-an-mbr-ransomware/30626/ [Accessed: June 2020].

[48] Vladimir Kuskov Mikhail Kuzin, Yaroslav Shmelev. 2018. New trends in the world of IoT threats. https://securelist.com/new-trends-in-the-world-of-iot-threats/87991/ [Accessed: April 2020].

[49] Amyas Morse. 2017. Investigation: WannaCry cyber attack and the NHS. *National Audit Office* (2017). https://www.nao.org.uk/wp-content/uploads/2017/10/Investigation-WannaCry-cyber-attack-and-the-NHS.pdf.

[50] Troy Mursch. 2018. 200,000+ MikroTik routers worldwide have been compromised to inject cryptojacking malware. https://badpackets.net/200000-mikrotik-routers-worldwide-have-been-compromised-to-inject-cryptojacking-malware/ [Accessed: April 2020].

[51] Netgear. 2015. R6250 Smart WiFi Router User Manual. http://www.downloads.netgear.com/files/GDC/R6250/R6250_UM_13Apr2015.pdf [Accessed: April 2020].

[52] Netgear. 2018. How to upload firmware to a NETGEAR router using Windows TFTP. https://kb.netgear.com/000059634/How-to-upload-firmware-to-a-NETGEAR-router-using-Windows-TFTP [Accessed: April 2020].

[53] NIST. 2017. CVE-2016-6277 Detail. https://nvd.nist.gov/vuln/detail/CVE-2016-6277 [Accessed: April 2020].

[54] NIST. 2018. CVE-2017-17215 Detail. https://nvd.nist.gov/vuln/detail/CVE-2017-17215 [Accessed: April 2020].

[55] NIST. 2019. CVE-2019-10999. https://nvd.nist.gov/vuln/detail/CVE-2019-10999 [Accessed: April 2020].

[56] NIST. 2019. CVE-2019-3929 Detail. https://nvd.nist.gov/vuln/detail/CVE-2019-3929 [Accessed: April 2020].

[57] Department of Health and Social Care. 2018. Securing Cyber Resilience in Health and Care. https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/747464/securing-cyber-resilience-in-health-and-care-september-2018-update.pdf [Accessed: April 2020].

[58] Gavin O'Gorman and Geoff McDonald. 2012. *Ransomware: A growing menace*. Symantec Corporation.

[59] ReFirmLabs. 2019. Binwalk. https://github.com/ReFirmLabs/binwalk [Accessed: April 2020].

[60] Checkpoint Research. 2017. A New IoT Botnet Storm is Coming. https://research.checkpoint.com/new-iot-botnet-storm-coming/ [Accessed: April 2020].

[61] Ryan Roemer, Erik Buchanan, Hovav Shacham, and Stefan Savage. 2012. Return-oriented programming: Systems, languages, and applications. *ACM Transactions on Information and System Security (TISSEC)* 15, 1 (2012), 1–34.

[62] Nolen Scaife, Henry Carter, Patrick Traynor, and Kevin RB Butler. 2016. Cryptolock (and drop it): stopping ransomware attacks on user data. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*. 303–312.

[63] Chris Simmonds. 2016. Software update for IoTthe current state of play. https://elinux.org/images/f/f5/Embedded_Systems_Software_Update_for_IoT.pdf [Accessed: April 2020].

[64] Chris Simmonds. 2016. Software updates for embedded Linux: requirement and reality. https://mender.io/resources/whitepapers/_resources/Software%2520Updates.pdf [Accessed: April 2020].

[65] Iain Thomson. 2017. Forget Mirai – Brickerbot malware will kill your crap IoT devices. https://www.theregister.co.uk/2017/04/08/brickerbot_malware_kills_iot_devices/ [Accessed: April 2020].

[66] Andrew Tierney. 2016. New Mirai Variant Uses Multiple Exploits to Target Routers and Other Devices. https://www.pentestpartners.com/security-blog/pwning-cctv-cameras/ [Accessed: April 2020].

[67] Andrew Tierney. 2016. Thermostat Ransomware: a lesson in IoT security. https://www.pentestpartners.com/security-blog/thermostat-ransomware-a-lesson-in-iot-security/ [Accessed: April 2020].

[68] Johannes Ullrich. 2014. Coin Mining DVRs: A compromise from start to finish. https://isc.sans.edu/diary/Coin+Mining+DVRs%3a+A+compromise+from+start+to+finish./18071 [Accessed: April 2020].

[69] Arnout Vandecappelle. 2012. Upgrade without Bricking. https://elinux.org/images/6/61/Upgrading_Without_Bricking.pdf [Accessed: April 2020].

[70] Evan Walls. 2019. GitHub - fuzzywalls/CVE-2019-10999. https://github.com/fuzzywalls/CVE-2019-10999 [Accessed: April 2020].

[71] Danielle Waugh. 2019. Riviera Beach Manager: City has most of its data back after ransomware attack. https://cbs12.com/news/local/riviera-beach-manager-city-has-most-of-its-data-back-after-ransomware-attack [Accessed: April 2020].

[72] Chris Williams. 2016. Today the web was broken by countless hacked devices - your 60-second summary. https://www.theregister.co.uk/2016/10/21/dyn_dns_ddos_explained/ [Accessed: April 2020].

[73] Karim Yagh, Jon Masters, Gilad Ben-Yossef, and Philippe Gerum. 2008. *Building Embedded Linux Systems*. O'Reilly, Chapter 8, 219, 236, 297–298.

[74] Einav Zilberstein and Adi Klein. 2017. https://documents.westerndigital.com/content/dam/doc-library/en_us/assets/public/western-digital/collateral/white-paper/white-paper-emmc-security.pdf [Accessed: April 2020].