TU Delft

Delft University of Technology

Document Version
Final published version

Licence
CC BY

Citation (APA)
Kuki, R., Sasaki, T., Al Alsadi, A., Gañán, C., & Yoshioka, K. (2026). The End of Anarchy? Understanding the Life of HTTP Exploits Used in IoT Malware Infections. In *ASIA CCS 2026 - Proceedings of the 21st ACM ASIA Conference on Computer and Communications Security* (pp. 128-143). Association for Computing Machinery (ACM). https://doi.org/10.1145/3779208.3785282

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
In case the licence states “Dutch Copyright Act (Article 25fa)”, this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

The End of Anarchy? Understanding the Life of HTTP Exploits Used in IoT Malware Infections

Ryu Kuki
Yokohama National University
Yokohama, Japan
kuki-ryu-dr@ynu.jp

Takayuki Sasaki
Yokohama National University
Yokohama, Japan
sasaki-takayuki-yv@ynu.ac.jp

Arwa Al Alsadi
Delft University of Technology
Delft, Netherlands
A.Alsadi@tudelft.nl

Carlos Gañán
Delft University of Technology
Delft, Netherlands
Yokohama National University
Yokohama, Japan
C.HernandezGanan@tudelft.nl

Katsunari Yoshioka
Yokohama National University
Yokohama, Japan
yoshioka@ynu.ac.jp

Abstract

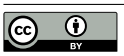
In recent years, while the threat of cyber attacks and malware targeting the vulnerabilities of the Internet of Things (IoT) has become more serious, various efforts have been made to mitigate the spread of these threats and the damage they cause. This study aims to understand the mechanisms behind the increase and decrease in HTTP-based exploits targeting IoT devices, and to verify the effectiveness of current prevention and mitigation measures. By integrating attack analysis observed in honeypots, collecting malware samples, and publicly available vulnerability information, we visualize the temporal changes in the number of attacks over the lifetime of vulnerabilities, and identify the impact of life events such as the release of the proof-of-concept (PoC) or the weaponization into malware on changes in attack trends. Comparing attack volumes for vulnerabilities discovered from 2008 to 2024, we discover differences in attack trends depending on the age of the device and vulnerability. Our results show that attacks targeting new devices with adequate security measures tend to decrease relatively quickly, while attacks targeting older devices with inadequate measures often persist until the end of the device's life, by being weaponized in malware. These findings confirm that existing security measures are effective to a limited extent, but at the same time, they also show that continuous improvement and proactive measures are necessary to prevent the exploitation of vulnerabilities from continuing for long periods of time.

CCS Concepts

• **Security and privacy → Web protocol security**; **Malware and its mitigation**.

Keywords

Honeypot, Malware, Vulnerability



This work is licensed under a Creative Commons Attribution 4.0 International License.
ASIA CCS '26, Bangalore, India
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2356-8/26/06
https://doi.org/10.1145/3779208.3785282

ACM Reference Format:
Ryu Kuki, Takayuki Sasaki, Arwa Al Alsadi, Carlos Gañán, and Katsunari Yoshioka. 2026. The End of Anarchy? Understanding the Life of HTTP Exploits Used in IoT Malware Infections. In *ACM Asia Conference on Computer and Communications Security (ASIA CCS '26), June 1–5, 2026, Bangalore, India.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3779208.3785282

1 Introduction

Exposed Internet of Things (IoT) devices such as routers, cameras, NAS, and NVRs, etc., are prime targets for attackers. In particular, HTTP-based exploits are one of the most important categories of IoT vulnerabilities. Researchers have looked into the landscape of IoT vulnerability exploitation. Alrawi et al. statically analyzed IoT malware binaries and reported that an increasing number of IoT vulnerabilities are exploited [1]. Alsadi et al. statically and dynamically analyzed 17K IoT malware binaries from different sources from 2015 to 2020. They found that 68 vulnerabilities, spanning from 12-year-old to most recent ones, are exploited [2] and observed that they were rarely abandoned once weaponized into malware binaries. Alsadi et al. also looked at the attackers' selection of vulnerabilities and identified several factors that explain their selections, such as device exposures, the availability of proof-of-concept codes, and availability and complexity of security patches [3].

While these studies provide valuable insights into how IoT exploits are selected and used in IoT malware, the whole life cycle of these exploits remains unclear. For example, how long does it take for IoT exploits to reach their peak of infections? What factors determine the size of this peak? Once peaked, how long does it take for them to decline? How do the declines relate to security practices deployed in the devices, such as automated firmware updates? Previous studies fail to grasp the whole landscape of IoT exploits as they limit their observations on the exploits weaponized into malware binaries. Indeed, IoT exploits may have different life cycles that do not involve weaponization into malware binaries, and the previous studies miss all these cases. Previous studies have also not looked at the lifetime of exploits, specifically how attacks decline.

In this paper, we aim to answer the following research questions: **(RQ1) How do attack trends differ depending on the age of vulnerabilities? (RQ2) What are the stages and events of IoT exploit life cycles, and how are they related?**

To answer RQ1, we compared recent exploit lifetimes for a wide range of IoT vulnerabilities disclosed from 2008 to 2024. As a result, we found differences in attack and vulnerability volumes among different device types. The results allowed us to identify three different time periods. For older vulnerabilities (–2012), attacks are now observed only sporadically, and their frequency remains low. This decrease can be attributed to the diminishing effectiveness of exploits as the product's life cycle ends and the number of active devices declines. In contrast, large-scale attacks are consistently observed for vulnerabilities from 2013 to 2018, as many devices were used without firmware updates and were left infected with malware. For more recent vulnerabilities (2019–), attacks tend to decline quickly, likely due to the widespread adoption of security measures such as automatic firmware updates. These results suggest that manufacturers' efforts to enhance security have had a positive impact. We refer to this trend as *the end of anarchy*. Moreover, our findings confirm that deploying and updating IoT devices with appropriate security measures can prevent prolonged vulnerability exploitation.

To answer RQ2, we observed attacks in honeypots for over 5 years and tagged 127 vulnerabilities of IoT devices for time-series analysis. Based on the honeypot observation, we focus on three stages of these exploits: start of attack, peak of attack, and decline of attack. We also utilize other data sources, such as NVD, CNVD, Vulners, Exploit-DB, GitHub, Packet Storm, security advisory, and device manuals, and surface analysis results of captured malware, to spot exploits' significant life events, *vulnerability disclosure*, *PoC release*, *weaponization in malware*, and *security patch release*, to analyze how these events affected their lives. At the start of an attack, we found a high correlation between PoC releases and an initial attack. In some extreme cases, an initial attack was observed within 24 hours after a PoC release. As the attacks peaked, we found typical cases of exploits weaponized into the malware binaries, multiplying the magnitude of attacks. On the other hand, we also observed some exploits launched on a large scale from fewer cloud-based hosts without being weaponized into malware binaries. We also confirmed that while patch releases and automatic firmware updates have contributed to the decline in attacks, weaponization of the exploits in malware binaries has slowed it down.

In summary, our contributions are:

- We conduct the first study to understand the life of IoT exploits using a mixed-method approach that combines five years of honeypot observations, malware analysis, and a survey of public information, including security advisories and device manuals.
- We found that manufacturer efforts have significantly reduced the lifetime of exploits on recent devices when compared to older devices with no security measures.
- We defined the three stages of IoT exploits (start, peak, and decline of attack) and analyzed the impact of key life events such as PoC release, weaponization in malware, and security patch release.

2 Life cycle of IoT exploits

Exploits, like any other software, experience different stages during their lifetime. Understanding the life cycle of IoT exploits is essential for developing effective security measures. This section explores the stages and key events that influence the exploitation and attack scale targeting IoT vulnerabilities. As an example, Figure 1 shows the observed attacks exploiting CVE-2020-8515, illustrating a typical life cycle of an IoT exploit. This timeline covers the period from disclosure at the NVD to the start, peak and decline of the attack. During this period, the other key events such as the release of security patches, PoC code and weaponization into malware have occurred. For CVE-2020-8515, the first attack occurred a day after the PoC release, followed by a rapid increase after weaponization into malware, a peak, and a gradual decline. This example highlights how these key events can significantly impact the trajectory of IoT exploits. By focusing on these events and their effects, we aim to provide a comprehensive understanding of IoT exploit life cycles.

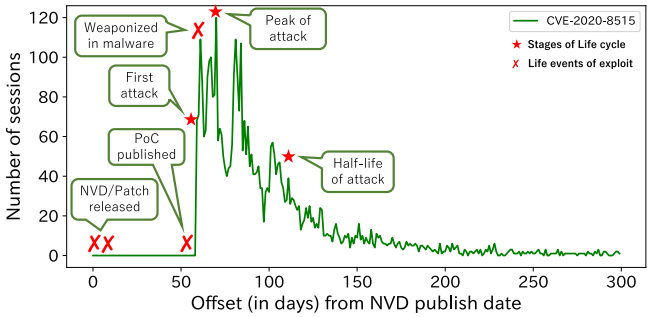


Figure 1: Exploit life events and number of attack observations in our honeypots for CVE-2020-8515

2.1 Stages of the exploit life cycle

This section defines the three key stages of an exploit's life cycle: the start of the attack, the peak of the attack, and the decline of the attack.

Start of attack: The observation of an exploit in the wild represents a critical event, marking the shift from a theoretical vulnerability to an active threat. To identify this stage, we define the start of the attack as the first instance of each exploit being observed in our honeypots. Because our visibility is limited to these sensors, this time is left-censored relative to the onset of exploitation in the wild and may postdate the earliest attacks. We will further discuss this in Section 5.1.

Peak of attack: Following the start, the intensity of attacks typically increases until it reaches a peak (e.g., see maximum in Figure 1). We define this peak as the day when the number of attacks against a specific vulnerability is at its highest during our observation period. This peak can be influenced by factors such as the number of vulnerable devices online and whether the exploit has been weaponized into malware. The criteria and methods for identifying this peak are detailed in Section 5.2.

Decline of attack: After reaching its peak, the frequency of attacks generally begins to decrease. We determine the decline phase by monitoring when the attack scale finally falls below 50% and subsequently 10% of the peak level. The rate of this decline can vary, influenced by the implementation of security measures

like automatic firmware updates. This phase and its indicators are further explored in Section 5.3.

2.2 Life events of exploit life cycle

The life cycle of an exploit is significantly influenced by several key life events that mark critical points in its evolution and impact. Understanding these events helps in assessing how vulnerabilities progress from discovery to widespread exploitation and eventual mitigation. This section outlines the major life events that affect the exploit life cycle and describes how they are identified and defined.

Vulnerability disclosure: The first major life event is the disclosure of the vulnerability. Ideally, a vulnerability is privately communicated to the product developer before any public release. Once a security patch is confirmed and released, the vulnerability details are made public through platforms like the NVD or security advisories. In the absence of explicit disclosure information, we use the NVD publication date as a reference point for identifying the start of an exploit's life cycle. This approach allows us to investigate the subsequent life events and events associated with each exploit.

PoC release: After a vulnerability is discovered, proof-of-concept (PoC) code may be created and disseminated through various channels, including underground markets [4] or public platforms like GitHub and Exploit-DB. The release of PoC code can significantly impact the start of an attack, as it provides attackers with a ready-made tool for exploiting the vulnerability. The effects of PoC release on attack initiation are examined in Section 5.1.2.

Weaponization in malware binary: Another critical event is the weaponization of exploits into malware. When exploits are embedded in malware binaries, the potential for widespread and prolonged attacks increases. This event can lead to a surge in attack scale and duration. The impact of this life event on the peak of attacks is explained in Section 5.2.2, and the impact on the decrease of attacks is explained in Section 5.3.3.

Release of security patches: Another event that impacts the life of an exploit is the release of security patches. Ideally, patches are issued before public disclosure, but in practice patch releases can be delayed. In addition, older IoT devices often lack automatic update mechanisms, so even when a patch becomes available, it is not applied and the devices continue to be used in a vulnerable state, which extends the effective lifetime of exploits. Furthermore, with the End of Support (EoS) of products, there are cases where security patches are never provided, resulting in vulnerabilities being permanently left open and the risk of attacks becoming permanent. The effectiveness of security patches in reducing attack frequency is analyzed in Section 5.3.1, and the effectiveness of automatic firmware updates is analyzed in Section 5.3.2.

3 Data collection

We collected publicly available information on vulnerabilities and exploits, along with attack data obtained through our honeypots.

3.1 Attack observation using honeypot

We have implemented an attack monitoring and analysis system consisting of a honeypot, a malware analysis component, and an attack tagging component (Figure 2). The malware samples collected by the honeypot are analyzed by the malware analysis component,

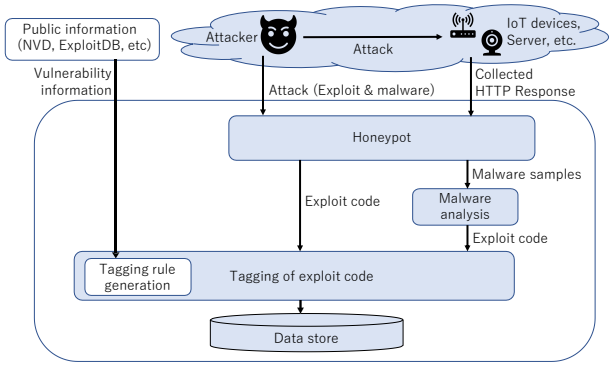


Figure 2: Attack monitoring and analysis system

which enables the extraction of exploit codes using surface analysis. The exploit codes obtained from the malware analysis and the observations made by the honeypot are then passed to the tagging component. Within this component, known attacks are tagged with a corresponding CVE-ID or target device information if a CVE-ID is not assigned.

3.1.1 Honeypot. One challenge in monitoring attacks against IoT devices is the wide range of IoT devices used for different purposes and deployed in different sectors, including homes, critical infrastructure, agriculture, and more. Given that attackers focus on specific devices, a typical honeypot that emulates only one type of device would fail to lure attacks that target different devices.

To observe attacks targeting a wider variety of IoT devices, we utilized X-POT [5], an adaptive HTTP honeypot. X-POT extends its emulated devices by collecting HTML responses from IoT devices, based on access from attack source hosts on the Internet that are potentially associated with malware-infected devices.

The honeypot comprises two components: an attack observation component that imitates vulnerable devices and captures exploits and malware, and an HTTP response collection component that scans back to the IP addresses to retrieve the HTML content of vulnerable devices.

The attack observation component imitates vulnerable IoT devices by responding with HTML files that are collected by the HTTP response collection component. Then, it monitors exploits sent by attackers as a result of imitation of vulnerable devices. In addition to observing attacks, the attack observation component retrieves malware samples by analyzing the HTTP requests associated with the attacks. Specifically, the honeypot examines the HTTP requests for download commands such as *curl* and *wget* and extracts a malware download URL.

To identify these potentially infected devices, the HTTP response collection component scans the IP address range around the host IP address that accessed the HTTP path. Specifically, if the attack observation component does not have an HTML response associated with the accessed path, the HTTP response collection component captures the HTML response representing the device's web user interface (WebUI). This approach allows our honeypot to emulate a wide variety of devices and provides valuable insights into attacks targeting different types of IoT devices.

We deployed 22 honeypot instances across multiple regions, hosted on our and partner universities and on cloud providers (e.g., Amazon EC2, DigitalOcean, and Vultr; Table 3 in Appendix A).

3.1.2 Tagging of the attack traffic. To classify and label the attack data collected from our honeypots, we have implemented an original tagging process. This process is carried out through a series of systematic steps to ensure accurate and meaningful classification of observed attacks.

Extraction of attack traffic. From the previous steps, raw data is collected from honeypots, which includes various communications, potential attacks, and exploit codes. The next step is attack extraction, where we filter out irrelevant data to isolate potential attacks. This involves excluding routine communications such as network scans from IoT scan engines. To extract exploits from HTTP traffic observed by our honeypot, we focus on HTTP requests that contain specific commands such as `wget` and `curl` for downloading external files. These commands are primarily used in particularly critical exploits that download malicious scripts or malware binaries [2, 6]. We note that focusing on these specific commands significantly reduces false positives, while potentially creating false negatives. We discuss this issue in Section 6.2.

Tagging rule generation. Once we have extracted attack traffic from the raw honeypot data, we define tagging rules to categorize attacks at the level of individual exploits. Each tagging rule consists of the following fields: (1) *Signature*: String signatures that represent specific attack patterns (e.g., HTTP paths, parameter names, and characteristic payload fragments); (2) *Exploit tag*: Exploit identifiers such as CVE-ID or target device name; (3) *Reference*: Pointers to exploit information sources such as Exploit-DB, NVD, and related advisories; (4) *Device type*: Whether the exploit targets IoT or non-IoT devices; (5) *IoT type*: For IoT exploits, the specific device category. Among the IoT exploits we observed, almost half targeted vulnerabilities in routers. The remaining IoT exploits, referred to as non-routers, cover 26 device types (e.g., IP cameras, NAS, NVRs, and DVRs), categorized based on prior research [1, 7].

Tagging rules are described in YAML format. An example is shown in Figure 13 in Appendix B. Attacks matching this rule are tagged with `CVE-2017-6334`. The HTTP request path `/dnslookup.cgi` and the field name `host_name`, which are specific to this exploit, are used as the signature to identify the attack. The target of this attack is an IoT device, specifically a router.

When new attacks are observed that do not match any of the existing rules, we create new rules using the following pipeline. Extraction of characteristic strings and the initial search against vulnerability databases are automated, whereas subsequent confirmation and rule creation are performed manually. All steps are reviewed for accuracy before a rule is deployed.

(1) **Extraction of characteristic strings.** From the HTTP request of the observed exploit, we extract characteristic strings, in particular the HTTP path and the field name containing the payload. This step is automated using regular expressions and heuristics that capture common exploit indicators such as `curl`, `wget`, or encoded payloads.
(2) **Lookup in vulnerability databases.** We automatically search vulnerability information databases (NVD, CNVD, Vulners, Exploit-DB, etc.) using the extracted strings as keywords. When database results are insufficient or ambiguous, we supplement them with manual searches, such as web search, vendor advisories, and security blogs.
(3) **CVE and target verification, and rule creation.** If an exploit can be identified, we manually verify the associated CVE-ID and the attack target (IoT vs. non-IoT, device type), and create new tagging rules based on this information.

To improve the accuracy and consistency of CVE mapping, we adopt the following decision logic when assigning CVE tags to rules: (i) when a single CVE corresponds to multiple target paths, we create a separate rule for each pattern, all tagged with the same CVE; (ii) when several CVEs share the same path, we add discriminative predicates (e.g., parameter or field names and value patterns) to separate them into distinct rules; and (iii) when ambiguity cannot be resolved (e.g., chained or co-exploited CVEs), we associate comma-separated multiple CVE tags with a single rule.

All rules are cross-checked before deployment by two security researchers with 5 and 20 years of experience, respectively; disagreements are resolved through manual re-analysis of the HTTP requests and payloads, and the referenced external sources.

Tagging of attack traffic. With the tagging rules established, we apply them to the extracted attack traffic in order to classify individual attacks. We match each HTTP request and payload against the signatures defined in the tagging rules and assign the corresponding tags when a match is found. This process produces a labeled dataset in which each attack instance is associated with its CVE-ID or device target. We apply the same tagging procedure not only to data collected after the rules are created but also retrospectively to all attack data collected before that time. This ensures that the entire dataset is labeled in a consistent manner.

To ensure the accuracy and reliability of the tagging process, we perform quality assurance checks. We manually review samples of the tagged data and confirm that the assigned tags correctly represent the underlying exploits. When we identify systematic errors or ambiguous matches, we revise the tagging rules and signatures and then reapply them to improve tagging accuracy.

Analysis of tagged data. Finally, we analyze the tagged data to derive insights and identify trends by aggregating it to summarize patterns in attack types and targets. Reports then detail findings, including attack distributions across CVEs and device types.

3.1.3 Malware analysis. We performed a surface analysis on the malware samples we collected to identify the exploits they weaponized. Surface analysis focuses on examining the observable characteristics of malware without extensive deconstruction of its code. Specifically, we extracted human-readable strings from the malware using the `strings` command. This process allowed us to gather data such as file paths, network addresses, and other indicators embedded within the samples.

To identify the exploits embedded in the malware binaries, we compared these extracted strings with signatures in the tagging rules. The tagging rule generation is discussed in Section 3.1.2. This comparison provided insights into which exploits the malware was designed to leverage.

Table 1: Honeypot observation statistics

Observation period: July 21, 2019 – August 14, 2024	
All HTTP sessions	226,720,844
Attack HTTP sessions	3,757,534
Attack source IPv4 addresses	195,452
Collected malware samples (Unique SHA1 hashes)	35,826

We did not unpack or fully reverse engineer packed samples, which meant that some exploits might have been obscured by obfuscation or encryption techniques used in the malware. Unpacking and more detailed reverse engineering could uncover additional exploits and deeper insights, but our current approach was focused on analyzing the strings and metadata without performing extensive deconstruction of the malware files.

3.1.4 Statistics of honeypot data. We collected the following data during 5 year observation period (from July 21, 2019 to August 14, 2024). We observed 3,757,534 attack HTTP sessions from 195,452 IP addresses and collected 35,826 malware samples (Table 1).

3.2 Public vulnerability and exploit information

We used a wide variety of data sources to collect information about vulnerabilities and exploits that have been published during the last decades.

3.2.1 Vulnerability information. We obtained vulnerability information from the following public sources and used them as a reference for the exploit life events. Information was retrieved on August 14, 2024, when we analyzed the honeypot data.

National Vulnerability Database (NVD) [8]: We collected 261,858 CVE-IDs published by August 2024 and vulnerability information (Description, Published Date, CVSS Score, Reference, CWE-ID, etc.) from NVD.

Chinese National Vulnerability Database (CNVD) [9]: We collected 116,405 CNVD-IDs and vulnerability information (Description, Published Date, Type of vulnerability, Reference, etc.) from CNVD.

Vulners [10]: We searched for vulnerability information and exploit codes in the Vulners database via the API.

3.2.2 Security patch information. We manually collected information on the availability of security patches and whether devices have automatic firmware update features in order to investigate the manufacturer's efforts.

Security advisories: We manually collected patch availability and release dates from security advisories published by product vendors or third parties. Specifically, we checked the websites of the manufacturers of the devices targeted by the exploits observed in the honeypots. In addition, we searched the name of the device or the name of the device manufacturer to see if any advisories had been issued by security firms such as Tenable or Rapid7. Thus, we collected 86 security advisories.

Device manuals: For each exploit observed in our honeypot, we identified the corresponding affected device models and collected their user manuals from the official manufacturer websites. We examined the manuals of all representative devices affected by the exploits, except in cases where the manuals were no longer available (e.g., end-of-life devices). The manuals were analyzed to determine the availability of security measures, such as automatic firmware updates.

3.2.3 Public exploits (PoC). After a vulnerability is discovered, PoC (proof-of-concept) may be created and published to demonstrate the feasibility of an attack that exploits the vulnerability. While these PoCs are useful in verifying vulnerabilities and providing countermeasures, there is a risk that PoCs can be easily exploited to implement an attack code targeting the vulnerability or to incorporate the PoCs into malware. To clarify the impact of the PoC release on the life cycle of the exploit, we collected PoC information from the following public sources on August 14, 2024.

Exploit-DB [11]: We cloned the official Exploit-DB GitLab repository [12] and collected 46,104 exploit codes and their metadata, including CVE-IDs and publication dates.

GitHub [13]: We searched GitHub using "CVE" as a keyword and collected 14,323 repositories whose title or description contains CVE-IDs, together with their publication dates.

Packet Storm [14]: We collected 50,581 exploit codes and their associated metadata from Packet Storm, including CVE-IDs and publication dates.

Using these CVE-centric matches across the three sources, we decided for each vulnerability whether a PoC was publicly available and, if so, recorded the earliest publication date observed among the matched artifacts.

3.2.4 IP address information. To determine whether attack source IP addresses observed by our honeypot are those of the hosting services or the ISPs, we used IPinfo [15], which provides organization information on IP address owners.

4 Overview of life of IoT exploits

We observed 127 exploits against IoT devices, of which 94 were assigned CVE-ID. In this section, we analyze long-term trends of **94 exploits** spanning a wide range of disclosure dates. Exploits without an assigned CVE-ID were excluded, as it was difficult to accurately link them with vulnerability disclosure dates or the availability of PoC information.

Our dataset is substantially larger than prior work on HTTP-based IoT exploits. For example, prior research [3] analyzes 59 unique exploits, whereas we analyze 94 unique exploits with CVE-ID, enabling more comprehensive coverage.

Table 2 shows the statistics of the analysis result. The detailed list is shown in Table 4 in Appendix F. Of the 94 exploits, PoCs of 49 exploits are available, 33 exploits are weaponized in malware, security patches of 55 vulnerabilities are available, and for 7 exploits, affected devices support automatic firmware updates. In terms of IoT type, there was no significant difference between router and non-router exploits, except that router exploits were more likely to be weaponized into malware. Specifically, 24 out of 33 were router exploits and 9 were non-router exploits.

As an overview, Figure 3 summarizes the temporal progression of attacks across all 94 exploits. It visually reveals differences in overall attack trends among old, middle-age, and recent vulnerabilities, and shows how key life events influence the lifecycle of each exploit. The color intensity indicates the number of attack requests,

Table 2: Statistics of analyzed IoT exploits

	Router	Non-router	Total
Exploits observed by our honeypot	63	64	127
CVE-ID assigned	51	43	**94**
Of CVE-ID assigned, disclosed after the honeypot deployment	27	25	**52**
Patch available	24	31	55
PoC available	26	23	49
Weaponized in malware	24	9	33

with red indicating attack requests from hosting services and blue indicating attack requests from ISPs. The | mark indicates the NVD disclosure date. The ● mark indicates the PoC release date, and the ♦ mark indicates when the exploit was first found in malware binaries collected by our honeypot. As discussed in Section 3.1.3, in order to identify the exploits weaponized in the malware samples, we compared the extracted strings from the samples against the signatures of tagging rules.

From these overall pictures, we can see the following:

- **PoC release:** Many cases of attacks were observed immediately after PoC releases. We examined the occurrence of PoC releases using public information such as Exploit-DB and GitHub in Section 5.1.2. In addition, we measured the time between the PoC release and the first observed attack.
- **Weaponization in malware:** The rate of exploits weaponized in malware is 35.1% (33 out of 94), which shows that it is important to observe both malware and honeypots. Some exploits had increased attacks after being weaponized into malware and persisted for a long period. We analyzed the relationship between this occurrence and peak height (Section 5.2.2) and decline speed (Section 5.3.3).
- **Peak and decline of attacks:** As for exploits capable of observing life events from the beginning (from honeypot deployment: July 21, 2019), the number of attacks often quickly reaches peaks, and then gradually decreases. We analyzed the relationship between decline speed and security patch release (Section 5.3.1) and automatic firmware update (Section 5.3.2).

4.1 Long-term trend of IoT exploits

Various measures have been implemented to counter the threat of attacks targeting IoT vulnerabilities, but the effectiveness of these measures has not been fully clarified. In this section, we will look at the long-term trends of these exploits to see if there are any signs of improvements due to the security measures. Namely, we analyze different ages of exploits targeting 94 vulnerabilities from 2008 to 2024. From Figure 3, we observe that IoT vulnerabilities can be categorized into the following three ages.

Old vulnerabilities(–2012): A small number of attacks and no large-scale attacks are observed on these old vulnerabilities. We can infer that attacks on these old vulnerabilities are inactive due to the small and decreasing number of devices with these vulnerabilities.

Middle-age vulnerabilities(2013–2018): Large-scale attacks have been ongoing continuously on these middle-age vulnerabilities. We infer that devices with these middle-age vulnerabilities are still in use, attracting attacks, while no sufficient measures, such as automatic updates, are applied to protect them.

Recent vulnerabilities(2019–): Large-scale attacks on the recent vulnerabilities do occur from time to time but do not last long compared to middle-age vulnerabilities.

One of the reasons why exploitation has clearly become less prolonged in recent years is that many new IoT devices have functions for security measures, such as automatic firmware updates. Automatic firmware updates help to quickly apply security patches to active devices after a vulnerability is discovered, shortening the lifespan of exploits. We investigated the existence of a firmware auto-update function by referring to the devices' manuals. Specifically, we examined 52 exploits that were disclosed after the start of honeypot observations and identified that the median half-life of an attack was 2 days with automatic firmware updates and 3 days with user notifications. We discuss this in detail in Section 5.3.2.

Answer to RQ1. In the oldest age group (–2012), attacks are observed sporadically, but their frequency is currently not high because the product life cycle has ended and the number of active devices has decreased. Large-scale attacks are consistently observed in the intermediate age group (2013–2018) because many devices are still being used in a vulnerable state. In recent age group (2019–), there has been a trend towards a decrease in attacks over a short period of time thanks to security measures such as automatic firmware update functions.

5 Analysis of exploit life events

The typical life of exploits includes the start of an attack, a peak, and a gradual decline in activity, as depicted in Figure 1.

To make these stages measurable, in this section we focus on the **52 exploits** shown in Table 2, whose CVE-IDs were disclosed after the start of our honeypot observations. Restricting to this subset ensures that all key life events can be analyzed starting from the time of the first observed attack.

First, we will provide an overview of the findings before going into more detail about each of the milestones. Figure 4 summarizes the offsets between the NVD publication date (day 0) and each life event and life stage. Security patches are typically released before public disclosure, and PoCs follow within a few days. While attacks sometimes begin immediately after disclosure, the median delay to the first observed attack is 72 days. In the earliest case, we observed malware weaponization one day after NVD publication and found no evidence of weaponization before the disclosure date. Attacks peak 209 days after disclosure, drop to half of the peak by 405 days, and fall to 10% by 432 days.

Attack decline is slower for router exploits than for non-router exploits, consistent with routers being more likely to be weaponized into malware (Section 4). We report the router and non-router breakdown in Appendix E (Figure 16 and Figure 17).

In the following, we examine each life event.

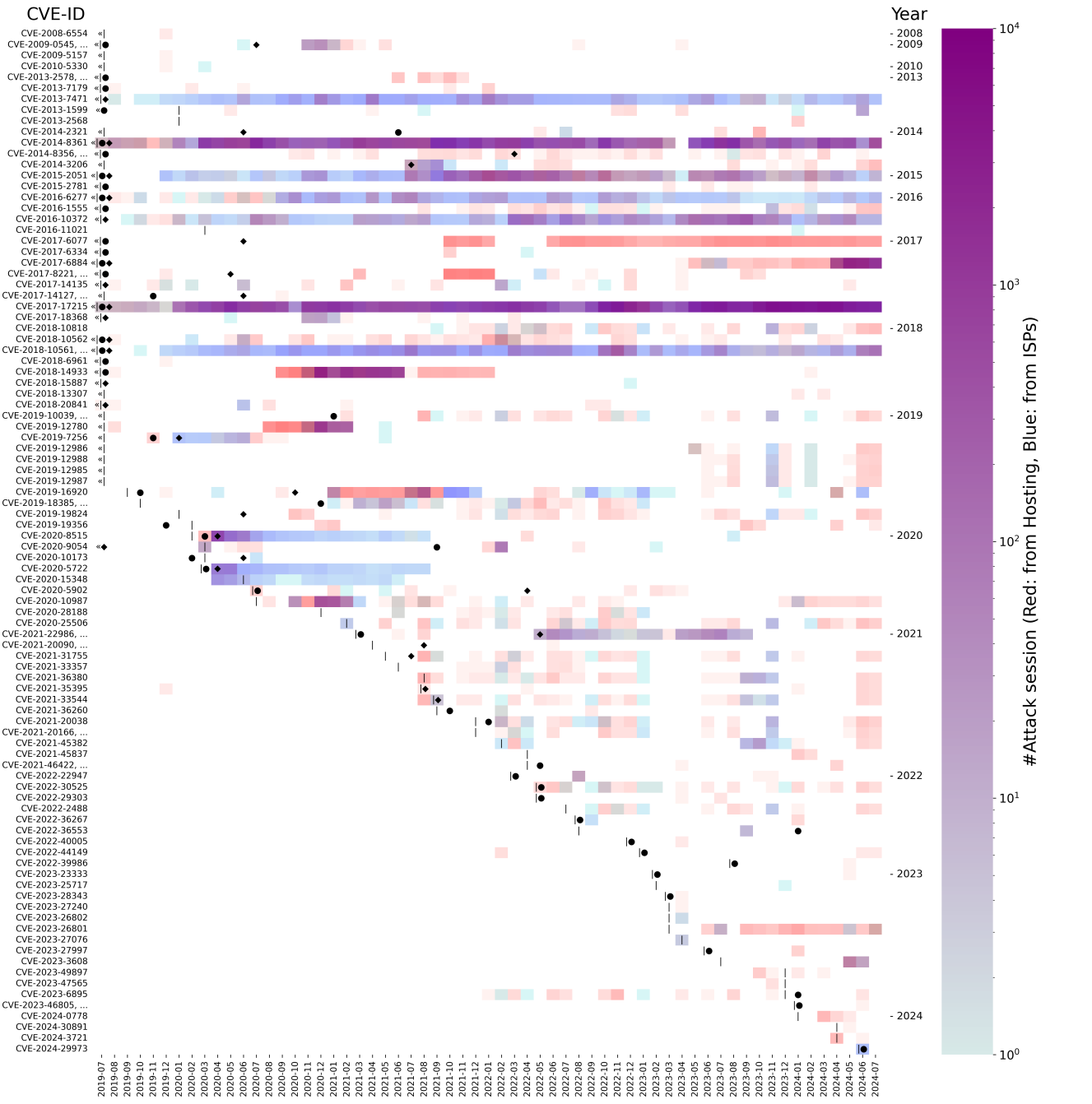


Figure 3: Overview of exploits targeting different vulnerabilities observed by honeypots

The color intensity indicates the number of attack requests, with red indicating more attack requests from hosting services and blue indicating more attack requests from ISPs.
The | mark indicates the date of NVD disclosure, the ● mark denotes the date of the PoC release, and the ♦ mark indicates the initial weaponization of the exploit in malware binaries.

5.1 Start of attack

Here, we investigate the start of the attack in terms of the attack source of the first attack and the PoC availability of the exploit.

5.1.1 Where does the first attack come from? There are two possible lives of an exploit: a case where the attack is launched from the hosting service and then weaponized in malware, and another case where the attack is weaponized in malware from the start. In the first scenario, the attacks mainly originate from the hosting service where the attacker's infrastructure is hosted. In the second, the attacks mainly originate from the ISPs where the infected consumer IoT devices are located. We examined the AS (Autonomous System)

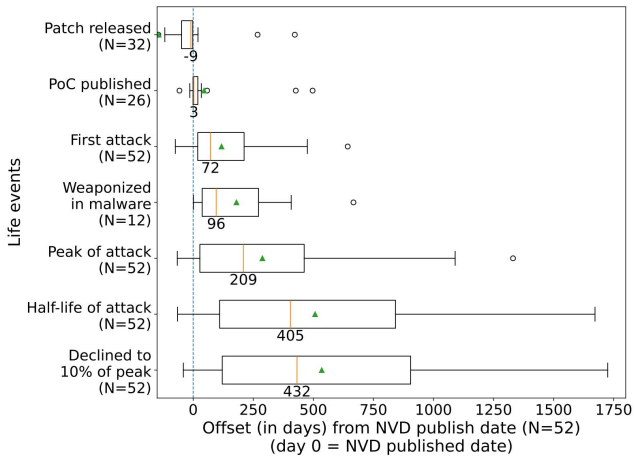


Figure 4: Time offset of life events to vulnerability disclosure

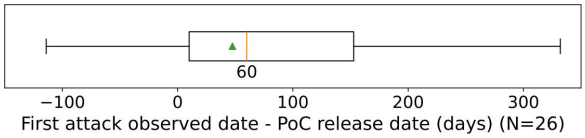


Figure 5: Days from PoC release to the first attack

of the source IP addresses of the attacks observed by our honeypot to see if the first attack came from a hosting service or an ISP.

The results showed that 71.2% of the initial attacks came from hosting services. This suggests that the lives of most exploits start at hosting service servers rather than malware.

5.1.2 PoC availability of attacks observed by honeypot. We investigated whether PoCs were available for the vulnerabilities targeted by the 52 exploits observed in the honeypots. As a result, the existence of PoC was confirmed for 26 vulnerabilities observed by our honeypot. The mapping between exploits and PoCs was based on the tags (CVE-IDs) assigned to each exploit by the tagging rules described in Section 3.1.2. We then checked whether a PoC existed in GitHub repositories, Exploit-DB, or Packet Storm that included the CVE-IDs in their metadata. If at least one of these sources provided a code artifact referring to the same CVE, we considered the PoC to be "available." As discussed in Section 6.2, PoCs published on other sites or those without an explicit CVE-ID may not have been captured, which could result in false negatives.

If PoCs are available, attackers can use them to carry out attacks quickly. Thus, for 26 exploits, we investigated the days from PoC release to the first attack observation date (Figure 5). Attacks were observed immediately after the PoC was released, with a median duration of 60 days. In 4 cases, a first attack was observed within 24 hours after a PoC release. The release of a PoC means that enough information is publicly available to assemble an exploit, and an attack could be launched at any time. Previous research [3] has discussed that if there is a PoC, it becomes easier to incorporate exploit code into malware. In our study, we observed that attacks can follow PoC releases quickly in a non-negligible subset of cases.

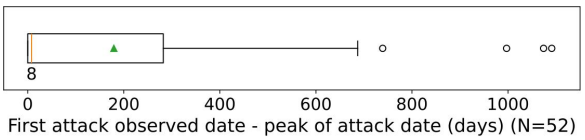


Figure 6: Days from the first attack to the peak of the attacks

5.1.3 Case study. To illustrate the impact of PoC release on the number of attacks, we show an example from our honeypot observations. Figure 1 shows attacks targeting CVE-2020-8515, an OS command injection vulnerability in DrayTek routers. Disclosed by NVD on February 1, 2020, a patch was released on February 6. While attacks were initially absent, botnet activity surged on April 1, just two days after the PoC was published on GitHub on March 30.

Figure 14 and Figure 15 in Appendix C and D show the PoC from GitHub and the attack HTTP requests observed in our honeypots. Apart from the payload content, the attack requests match the PoC perfectly, strongly suggesting that the PoC release triggered the implementation of malware in the botnet.

Summary of the start of the attack. The results showed that 71.2% of the initial attacks came from hosting. The existence of PoC was confirmed for half of the vulnerabilities observed by our honeypot. Attacks were sometimes observed immediately after the release of a PoC. In 4 cases, the first attacks were observed within 24 hours of the PoC release.

5.2 Peak of attack

Next, we investigate attack peaks. Specifically, we analyze how fast an attack expands and reaches its peak and whether the malware has any impact.

5.2.1 Time from start of attacks to peak of attacks. The speed at which attacks peak is critical for assessing attacker strategies and defender response times. Thus, for 52 exploits, we analyzed the time from the first observed attack to its peak (Figure 6). We defined the peak for each exploit as the day with the highest number of requests observed in our honeypots, based on the daily counts of exploit attempts. As a result, with a few exceptions, most attacks peaked quickly, with a median time of 8 days.

From the attacker's perspective, it seems reasonable that the scale of the attack peaks immediately after the first attack since the attacker needs to attack and spread the infection before countermeasures are implemented. From a defensive point of view, this result indicates the need to implement countermeasures as soon as a vulnerability is disclosed to quickly contain the spread of the attack.

5.2.2 Increased attacks after weaponization in malware. Weaponization of an exploit in malware is typically followed by a rapid increase in attack volume. Among the 12 exploits for which we confirmed weaponization, 11 reached their peak attack scale after the exploit first appeared in malware. For example, CVE-2020-5722, CVE-2020-8515, CVE-2020-14882, and CVE-2021-31755 in Figure 3.

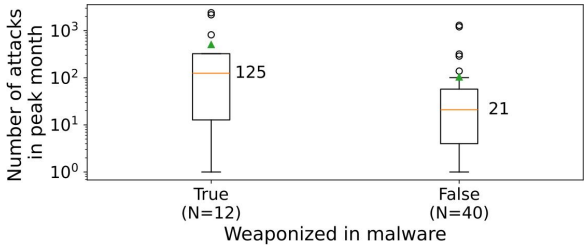


Figure 7: The impact of weaponization in malware on peak attack size

Figure 7 shows that the median number of attack sessions when weaponized in malware was 125 sessions per month, compared to 21 when not weaponized in malware. The average values are 507 and 104, respectively. This is consistent with the intuition that a large number of attacks are carried out from botnets.

Figure 8 shows the cumulative number of attacks for each of the 52 exploits. The blue bar graph shows attacks from ISP networks, and the red bar graph shows attacks from hosting services. The ♦ mark in the vulnerability label indicates that we have determined that it has been weaponized into malware. It can be inferred that many of the vulnerabilities with a large number of attacking hosts tend to have exploits weaponized in malware, and that infected IoT devices on ISP lines are being used in attacks. There are a few exceptions, such as CVE-2020-15348 and CVE-2020-35665, which may reflect botnets that were already compromised via other exploits or the limited observation capabilities of our honeypots.

On the other hand, there are also cases where vulnerabilities that are not weaponized into malware are attacked on a large scale from a small number of hosts on a hosting service. This indicates two attacker strategies: scaling attacks through infected IoT devices, and conducting mass scanning and exploitation from attacker controlled hosting infrastructure. These results also imply that malware only analysis [2, 3] is insufficient to capture overall attack trends, and that combining honeypot based attack observation with malware analysis provides broader coverage.

5.2.3 Case study. We present a case study in which the weaponization of exploit code in malware coincides with a sharp increase in attacks. Figure 9 shows the time series of attack hosts targeting CVE-2020-8515 observed by our honeypots. Attacks initially originated from a small number of hosts on a hosting service, but the number of attacking hosts increased markedly after April 1, 2020. Based on the malware samples we collected, we confirmed that the CVE-2020-8515 exploit was weaponized on April 1, 2020, which suggests a shift to attacks from infected hosts that propagated further infections and enabled large-scale attacks. This case indicates that weaponization in malware can trigger attack escalation.

Summary of the peak of attack. The attack scale peaks at a median of 8 days after the attack begins. Exploits weaponized in malware tend to have a larger peak attack scale than those that are not, due to infection spread.

5.3 Decline of attack

Here, we discuss the lifetime of an exploit. Given the efficiency of the attack, an attacker may end up exploiting a vulnerability when it is no longer feasible to do so, i.e., when the number of devices with unapplied security patches for the vulnerability has decreased sufficiently. We investigated the period of time it takes for the scale of attacks against each vulnerability to finally drop to less than half or 10% of its peak.

5.3.1 Impact of security patch on the decline of attack. If a security patch is not released, attacks can persist for extended periods. Figure 10 shows a median half-life of 20 days without a patch, compared to 1 day when a patch is available. The average half-lives are 240 and 178 days, respectively, indicating that security patches significantly shorten attack duration. However, attacks sometimes persist even with available patches, likely due to users failing to apply them. This highlights the importance of timely patch application and the need for device manufacturers to implement automatic patch and firmware updates. Next, we examine the role of automatic updates in reducing attack duration.

5.3.2 Impact of automatic firmware updates on the decline of attack. Attacks targeting vulnerabilities of devices with automatic firmware updates are declining rapidly. For 52 exploits, we examined whether the target devices had automatic firmware updates and measured the time it took for the attacks to be reduced by half. Figure 11 shows that attacks against devices with automatic update functionality are halved in a median of 2 days, compared to 3 days for devices equipped only with new firmware notification functions. The average days are 9.5 and 40.5, respectively.

Examining the average days and the right ends of the box plot, we see that devices with only firmware notifications tend to have a longer tail, indicating prolonged attacks compared to devices with automatic updates. The "Unknown" category includes devices with automatic updates, new firmware notifications, and those with neither, likely contributing to an even longer tail.

These results suggest that attackers stop exploiting vulnerabilities once they are fixed, as attacks become ineffective. Therefore, the presence of automatic update or notification mechanisms is associated with earlier suppression of attacks.

5.3.3 Impact of weaponization in malware on the decline of attack. Exploits weaponized in malware exhibit a slower decline in attack size. In Figure 12, the half-life is 119 days in median and 410 days on average for exploits weaponized in malware, compared to 1 day in median and 139 days on average when weaponization is not observed. This indicates a long tailed distribution of half-lives for weaponized exploits.

If attackers actively manage their infrastructure and select suitable exploits, the decline should not depend on whether attacks originate from cloud infrastructure or IoT device botnets. The slower decline for weaponized exploits may instead reflect reduced attacker control, where infected devices continue attacking for a long time. Consistently, Figure 3 shows higher weaponization rates for older exploits: 57.1% (20 out of 35) for old-age exploits (–2012) and mid-age exploits (2013–2018), versus 22.0% (13 out of 59) for young-age exploits (2019–).

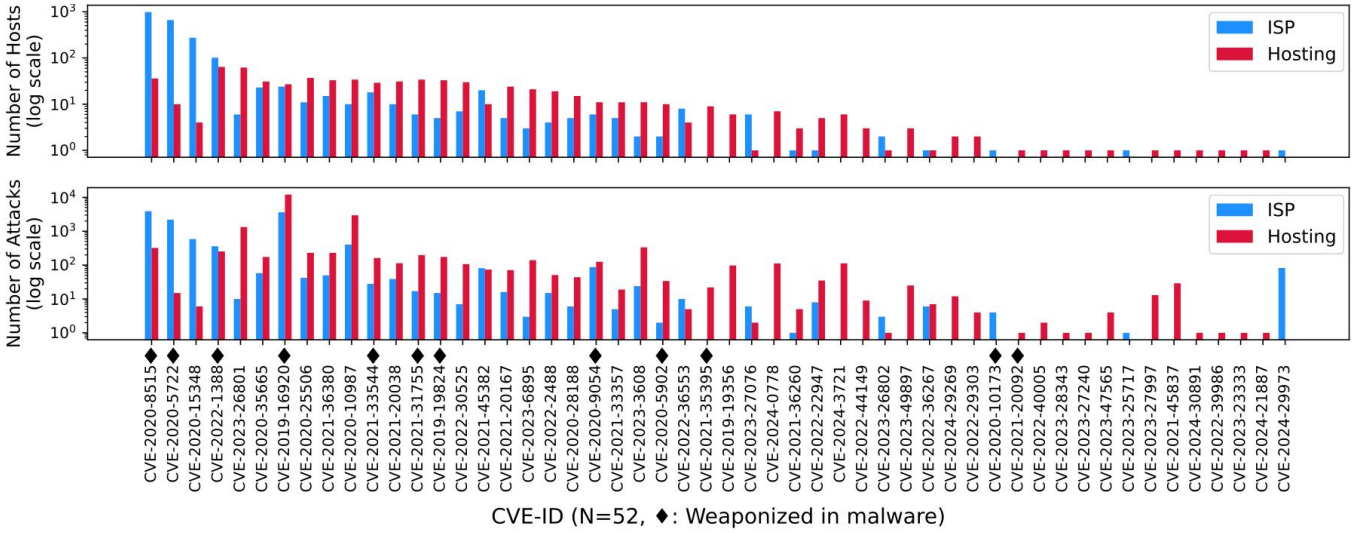


Figure 8: Scale of attacks observed by our honeypots

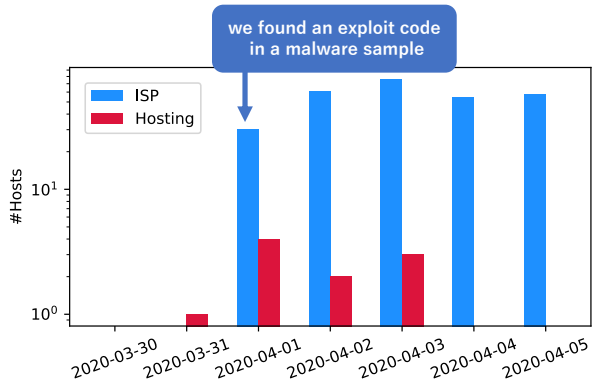


Figure 9: Number of attack hosts targeting CVE-2020-8515

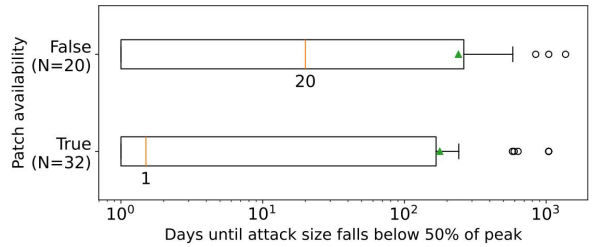


Figure 10: The impact of security patch on the half-life of attack scale

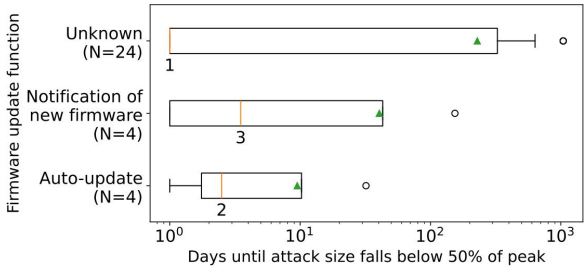


Figure 11: The impact of firmware auto-update on the half-life of attack scale

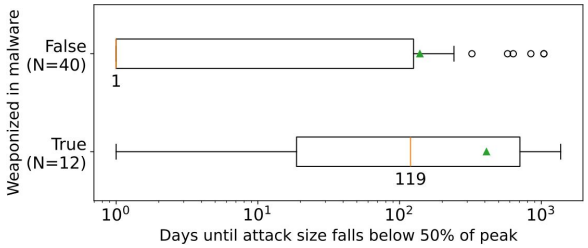


Figure 12: The impact of weaponization in malware on the half-life of attack scale

5.3.4 Case study. Here, we show an example of a decline of attacks. In Figure 1, which shows the number of attacks targeting CVE-2020-8515, after reaching a peak, the volume of attacks fluctuates up and down while gradually decreasing in scale. Eventually, 32 days after the peak, the scale of attacks decreases by half. After that, the scale of attacks decreased smoothly, and it was 91 days after the peak when it reached 10% of the peak. As shown in Figure 9, most of the attacks in the latter half of the attack were from ISPs' IP addresses, i.e., from malware. This is considered to be the reason why the decrease in the scale of the attack was slow in this case.

Summary of the decline of attack. Once an exploit is weaponized in malware, attacks tend to last longer than those that are not. In addition, when the patch is not available, the attack lasts longer than when the patch is available. In addition, if there is no automatic firmware update feature, the attack may continue for a longer period of time.

5.4 Summary of stages and events of the exploit life cycle

To consolidate the above analysis results of three stages, here we answer RQ2.

Answer to RQ2. We focus on three stages of exploits: the start of the attack, the peak of the attack, and the decline of the attack. At the start of the attack, attacks are launched immediately after a PoC is released. At the peak of the attack, exploits weaponized in malware tend to have a larger peak attack scale than those that are not. During the decline stage, attacks decrease rapidly for exploits targeting vulnerabilities for which patches are available, as well as for exploits targeting devices with firmware auto-update functionality. In addition, once an exploit is weaponized in malware, the speed of decrease is slower.

6 Discussion

6.1 Recommendations

On the basis of our analysis above, we recommend the following.

For IoT Device Manufacturers. Our measurements show that timely patching and automatic firmware updates clearly shorten attack duration. Vendors should provide security patches as early as possible and enable vendor managed automatic updates by default, especially when new vulnerabilities or PoCs are disclosed. They should also issue clear end of support notices and explicitly advise users to replace products that remain exposed and are still targeted in attacks.

For Network and Cloud Service Providers. IoT exploit attacks originate from both infected ISP devices and cloud hosted servers (Figure 8). ISPs and cloud providers should deploy filters for the commodity exploit and malware download patterns observed in this study, rate limit or block such traffic, and identify, isolate, or remove malicious hosts that repeatedly generate it.

For Users. Users of older or unsupported devices should replace them with newer models that receive security updates. When available, automatic firmware updates should be left enabled, and manual updates should be applied promptly after notifications.

For Rule Makers. Patch releases and automatic updates effectively reduce attack volume, so regulations and certification schemes should encourage these practices. For instance, the U.S. CyberTrustMark [16] includes QR codes linking to security update policies, while Japan's IoT labeling program [17] requires manufacturers to document policies prioritizing timely security updates.

6.2 Limitations

We conducted our analysis based on honeypot observations and publicly available data, which has the following limitations in terms of data collection capacity. Even with the limitations, we believe we have uncovered the life of IoT exploits.

Survey of discovery of vulnerabilities. The full extent of how and by whom vulnerabilities are discovered is still not known. It is observable that vulnerabilities are registered in NVD or used in attacks. However, sometimes vulnerabilities are discovered but not disclosed, and such cases are not fully investigated in this paper. Even with this limitation, NVD covers a large number of vulnerabilities and is considered sufficient to identify overall trends.

PoC collection. In this paper, PoC analysis is CVE-centric and only considers artifacts that explicitly include a CVE-ID in GitHub, Exploit-DB, or Packet Storm. PoCs for vulnerabilities without CVE-IDs or hosted on other or closed platforms may therefore be missed, which can introduce false negatives in our PoC availability measurements. Even with this limitation, these well-known sites are expected to cover the majority of publicly shared PoCs that influence real-world attacks.

Honeypot types and vulnerability types to be analyzed. We used an HTTP honeypot that only observes HTTP-based attacks. However, we can also say that the WebUI of IoT devices is a typical attack vector, and many other protocols used in IoT devices, such as CPE WAN Management Protocol (CWMP) and Simple Object Access Protocol (SOAP), are actually "HTTP-based". We believe that we should be able to grasp the trend of IoT exploits even with the limitations. Another limitation is that we only focused on remote code execution (RCE) attacks with particular commands, such as *curl* and *wget*. This works well to filter out the irrelevant traffic while it creates potential false negatives. With this limitation, we can still understand the trends of attacks, since most serious attacks that result in malware infection are RCE [18].

Limited observation period. The observation period of our honeypots is five years. Longer observation periods are needed to discuss long-term exploit milestone transitions of their life cycle. Even with this limitation, we are confident that we have been able to observe long-term trends, as we have a five-year observation period.

Malware analysis. We perform surface analysis on malware samples collected in our honeypots and use the results to tag exploits. This design implies that we may miss exploits that are embedded in heavily obfuscated or encrypted binaries. However, prior works [1, 19] report that only a small fraction of IoT malware uses packing and that static analysis still provides accurate visibility into embedded exploits in this ecosystem. In our dataset, only 7.65% of malware samples contain UPX header magic bytes, which suggests that any bias introduced by packed binaries is limited. A more complete treatment of highly obfuscated samples, including unpacking and selective manual analysis, is an important direction for future work.

7 Related work

Honeypots. A honeypot is a system that imitates a vulnerable system and observes cyber attacks targeting the vulnerability. X-POT [5] is an HTTP honeypot that can extend the devices to be

imitated by collecting HTML files of Web interfaces of IoT devices that are targets of cyber attacks. In addition, various other types of honeypots have been proposed, such as IoT-POT [20], which are honeypots using actual IoT devices, and honeypots that imitate remote management systems for infrastructure [21, 22]. There have been many analyses of attack data collected by honeypots. Nawrocki et al. summarized the analysis of honeypots and their data [23]. Analysis of the attack targets, in particular, has been a subject of interest for many studies, such as studies that analyzed the ports and protocols of the attack targets [24–27], studies that analyzed the services of the attack targets [28, 29].

Exploit life cycle. Prior work has analyzed exploits embedded in malware [2] by monitoring malware behavior by dynamic analysis, showing that IoT malware targets many vulnerabilities. Large-scale malware life-cycle studies summarize targeted vulnerabilities and devices [1], and prior work has also examined remote infections of Linux-based IoT devices [30]. Other studies report that PoC availability increases, while patch availability decreases, how frequently devices are targeted [3].

These previous studies have mainly focused on exploits in IoT malware. However, not all exploits are weaponized in malware, and some exploits are only used by attack hosts on cloud services. As a result, prior studies could not provide a comprehensive picture of the lifetime of an exploit from its exploit initiation to its convergence. In addition to these limitations, the studies did not provide a quantitative analysis of the actual increase or decrease in attack size, nor were they based on detailed time-series data. In our study, we analyzed honeypot observations and malware samples over the lifetime of multiple exploits, and by defining the milestones of the exploit life cycle, we revealed the impact of several life events on the scale of attacks.

Security practices of IoT manufacturers. Perez et al. surveyed vulnerability management practices among IoT and non-IoT manufacturers [31]. They found that IoT manufacturers release patches more promptly than non-IoT vendors. Nakajima et al. examined vulnerability disclosures and patch release behaviors for consumer IoT devices from three Japanese and three U.S. vendors [32]. They found that most IoT manufacturers released patches in a timely manner. Emami-Naeini conducted interviews with experts and consumers to identify priorities for informing consumers about the risks associated with IoT devices [33]. Based on these insights, they proposed a privacy and security label that indicates whether security updates are automatic and how long the device manufacturer will continue to provide updates.

As discussed above, our study also shows the importance of security practices by IoT manufacturers, such as patch releases and automatic firmware updates, in terms of attack observation using our honeypot and malware analysis.

8 Ethics considerations

We designed our study to maximize the potential benefits and minimize any possible harm that could result from our research, with reference to the Menlo report [34].

- **Analysis of life of exploits.** This paper does not disclose specific attack methods and is, therefore, unlikely to benefit the attackers. On the other hand, it provides valuable insights into attacker strategies for defenders, which we believe makes the publication of this paper highly worthwhile.
- **Honeypot observation.** We utilized the honeypot to ensure that attackers could not misuse the honeypot. Specifically, we configured the honeypot to prevent attackers from using it as a stepping stone and to ensure that malware does not execute on the honeypot.
- **Responsible disclosure.** We discovered a zero-day vulnerability through honeypot observation and reported it to the national CERT. The vulnerability has been fixed by the manufacturer, and a CVE-ID (CVE-2023-49897) was issued. We also shared information with the national CERT when known vulnerabilities started to be exploited by actual attacks.

9 Artifact/Open science

We will share the following data used in this paper.

- Honeypot data such as malware samples and attack HTTP requests. As the data can be used for attacks, we will share the data upon request from researchers. In addition, data may not be provided to organizations listed on a national export control list.
- Tagging rules of exploits
- Links to public data such as vulnerability information and PoCs
- Links to Manuals of the devices targeted by exploits
- Scripts used to analyze the above data

10 Conclusion

In this study, we conducted a comprehensive analysis of attacks observed in honeypots, collected malware samples, and publicly available vulnerability information. In the start of attack, vulnerabilities publicly disclosed with PoCs are highly susceptible to immediate attacks. These attacks begin soon after the vulnerabilities are disclosed. To the peak of the attack, attackers may scale up their attacks by weaponizing exploit code into malware or using hosting services under their control. We have illustrated several cases where attacks rapidly escalated due to the inclusion of exploits in malware. In the decline stage of the attack, certain vulnerabilities are no longer targeted, while others continue to be exploited for an extended period of time. In particular, we observed that exploits that are weaponized in malware have a longer lifetime than those that are not. We also confirmed that exploits targeting devices with published security patches and automatic firmware updates have shorter lifetimes.

Acknowledgement

A part of these research results were obtained from the commissioned research(No.JPJ012368C08101) by National Institute of Information and Communications Technology (NICT), Japan. This work was supported by JSPS KAKENHI Grant Number 21KK0178. This publication is also supported by the RAPID project (Grant No. CS.007) of the Dutch Research Council (NWO).

References

[1] O. Alrawi, C. Lever, K. Valakuzhy, R. Court, K. Snow, F. Monrose, and M. Antonakakis, "The circle of life: A Large-Scale study of the IoT malware lifecycle," in *30th USENIX Security Symposium (USENIX Security 21)*, Aug. 2021, pp. 3505–3522.

[2] A. A. Al Alsadi, K. Sameshima, J. Bleier, K. Yoshioka, M. Lindorfer, M. van Eeten, and C. H. Gañán, "No spring chicken: Quantifying the lifespan of exploits in iot malware using static and dynamic analysis," in *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, ser. ASIA CCS '22, 2022, p. 309–321.

[3] A. A. A. Alsadi, K. Sameshima, K. Yoshioka, M. V. Eeten, and C. H. Ganan, "Bin there, target that: Analyzing the target selection of iot vulnerabilities in malware binaries," in *RAID*, 2023.

[4] J. Burgess, "Malware and exploits on the dark web," 2022.

[5] S. Kato, R. Tanabe, K. Yoshioka, and T. Matsumoto, "Adaptive observation of emerging cyber attacks targeting various iot devices," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021, pp. 143–151.

[6] H. Victor, S. Kobayashi, and T. Yamauchi, " Analyzing Post-injection Attacker Activities in IoT Devices: A Comprehensive Log Analysis Approach ," in *2023 Eleventh International Symposium on Computing and Networking Workshops (CANDARW)*. Los Alamitos, CA, USA: IEEE Computer Society, Nov. 2023, pp. 292–297. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CANDARW60564.2023.00055

[7] X. Feng, Q. Li, H. Wang, and L. Sun, "Acquisitional rule-based engine for discovering Internet-of-Things devices," in *27th USENIX Security Symposium (USENIX Security 18)*. Baltimore, MD: USENIX Association, Aug. 2018, pp. 327–341. [Online]. Available: https://www.usenix.org/conference/usenixsecurity18/presentation/feng

[8] "National Vulnerability Database," https://nvd.nist.gov/, Online.

[9] "Chinese National Vulnerability Database," https://www.cnvd.org.cn/, Online.

[10] "Vulners," https://vulners.com/, Online.

[11] "Exploit-DB," https://www.exploit-db.com/, Online.

[12] "The official exploit-database repository," https://gitlab.com/exploit-database/exploitdb, Online.

[13] "GitHub," https://github.com/, Online.

[14] "Packet Storm," https://packetstormsecurity.com/, Online.

[15] "IPinfo.io," https://ipinfo.io/, Online.

[16] Federal Communications Commission (FCC), "U.s. cyber trust mark," https://www.fcc.gov/CyberTrustMark, 2024.

[17] J. Information-technology Promotion Agency, "Japan cyber star (jc-star)," https://www.ipa.go.jp/en/security/jc-star/index.html, 2024.

[18] R. Kaksonen, K. Halunen, and J. Röning, "Vulnerabilities in iot devices, backends, applications, and components," in *Proceedings of the 9th International Conference on Information Systems Security and Privacy*. Scitepress, 2023.

[19] S. Kumar, P. Ahlawat, and J. Sahni, "Iot malware detection using static and dynamic analysis techniques: A systematic literature review," *SECURITY AND PRIVACY*, vol. 7, no. 6, p. e444, 2024. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/spy2.444

[20] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, "Iotpot: Analysing the rise of iot compromises," in *9th USENIX Workshop on Offensive Technologies (WOOT 15)*, 2015.

[21] T. Sasaki, A. Fujita, C. Ganan, M. van Eeten, K. Yoshioka, and T. Matsumoto, "Exposed infrastructures: Discovery, attacks and remediation of insecure ics remote management devices," in *2022 IEEE Symposium on Security and Privacy*, 2022, pp. 1308–1325.

[22] J. Franco, A. Aris, B. Canberk, and A. S. Uluagac, "A survey of honeypots and honeynets for internet of things, industrial internet of things, and cyber-physical systems," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2351–2383, 2021.

[23] M. Nawrocki, M. Wählisch, T. C. Schmidt, C. Keil, and J. Schönfelder, "A survey on honeypot software and data analysis," *CoRR*, vol. abs/1608.06249, 2016. [Online]. Available: http://arxiv.org/abs/1608.06249

[24] F. Pouget, M. Dacier *et al.*, "Honeypot-based forensics," in *AusCERT Asia Pacific Information Technology Security Conference*, 2004.

[25] J. Francois, O. Festor *et al.*, "Activity monitoring for large honeynets and network telescopes," *International Journal on Advances in Systems and Measurements*, vol. 1, no. 1, pp. 1–13, 2008.

[26] S. Almotairi, A. Clark, G. Mohay, and J. Zimmermann, "Characterization of attackers' activities in honeypot traffic using principal component analysis," in *2008 IFIP International Conference on Network and Parallel Computing*, 2008, pp. 147–154.

[27] O. Thonnard and M. Dacier, "A framework for attack patterns' discovery in honeynet data," *digital investigation*, vol. 5, pp. S128–S139, 2008.

[28] E. Alata, V. Nicomette, M. Kaaniche, M. Dacier, and M. Herrb, "Lessons learned from the deployment of a high-interaction honeypot," in *2006 Sixth European Dependable Computing Conference*, 11 2006, pp. 39 – 46.

[29] G. Portokalidis, A. Slowinska, and H. Bos, "Argos: An emulator for fingerprinting zero-day attacks for advertised honeypots with automatic signature generation," in *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006*, ser. EuroSys '06, 2006, p. 15–27.

[30] H. Li, Q. Huang, F. Ding, H. Hu, L. Cheng, G. Gu, and Z. Zhao, "Understanding and detecting remote infection on linux-based iot devices," in *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, ser. ASIA CCS '22, 2022, p. 873–887.

[31] S. R. Perez, M. Van Eeten, and C. H. Ganan, " Patchy Performance? Uncovering the Vulnerability Management Practices of IoT-Centric Vendors ," in *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, May 2024, pp. 1198–1216.

[32] A. Nakajima, T. Watanabe, E. Shioji, M. Akiyama, and M. Woo, "A pilot study on consumer iot device vulnerability disclosure and patch release in japan and the united states," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, ser. Asia CCS '19. Association for Computing Machinery, 2019, p. 485–492.

[33] P. Emami-Naeini, Y. Agarwal, L. Faith Cranor, and H. Hibshi, "Ask the experts: What should be on an iot privacy and security label?" in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, pp. 447–464.

[34] "The menlo report: Ethical principles guiding information and communication technology research," https://www.dhs.gov/sites/default/files/publications/CSD-MenloPrinciplesCORE-20120803_1.pdf, 2012.

[35] "CVE-2020-8515-PoC," https://github.com/imjdl/CVE-2020-8515-PoC, Online, this work is distributed in the Apache License 2.0. (https://www.apache.org/licenses/LICENSE-2.0).

A Honeypot location

Table 3: Deployed honeypot location

VPS/Cloud	Country Code	# IP
AWS	AU,BH,BR,DE,HK,IE,IN,JP,KR,SE,SG,US	12
DigitalOcean	IN,NL,SG	3
Vultr	CA,FR,GB,JP,SG	5
University	JP, VN	2
Total		22

B Example of YAML tagging rule

```
- signature:
  - /dnslookup.cgi
  - host_name
  memo: https://www.exploit-db.com/exploits/42257
  tag: CVE-2017-6334
  device_type: iot
  iot_type: router
```

Figure 13: Tagging rules for CVE-2017-6334

C Example of PoC code

```
def run_cmd(self, cmd):
  try:
    headers = {
      "UserAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv
          :75.0) Gecko/20100101 Firefox/75.0"
    }
    url = self.url + "/cgi-bin/mainfunction.cgi"
    data = "action=login&keyPath=%27%0A%2fbin%2f" + cmd + "%0A%27&
        loginUser=a&loginPwd=a"
    res = req.post(url=url, data=data, timeout=(10, 15), headers=
        headers)
    if res.status_code == 200:
      return res.text
    else:
      return ""
  except Exception as e:
    return ""
```

Figure 14: PoC released on GitHub (CVE-2020-8515) [35]

D Example of an attack using PoC

```
POST /cgi-bin/mainfunction.cgi HTTP/1.1
User-Agent: XTC
Host: 127.0.0.1
Content-Length: 1000
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

action=login&keyPath='/bin/sh –c 'wget http://XXX.XX.XX.XX/YYYY -O
/tmp/upnp.debug; chmod 777 /tmp/upnp.debug; /tmp/upnp.debug''
&loginUser=a&loginPwd=a   Malware Download & Execution Cmd
```

Figure 15: Attack requests observed in the honeypot (CVE-2020-8515)

E Life events of routers and non-router IoT devices

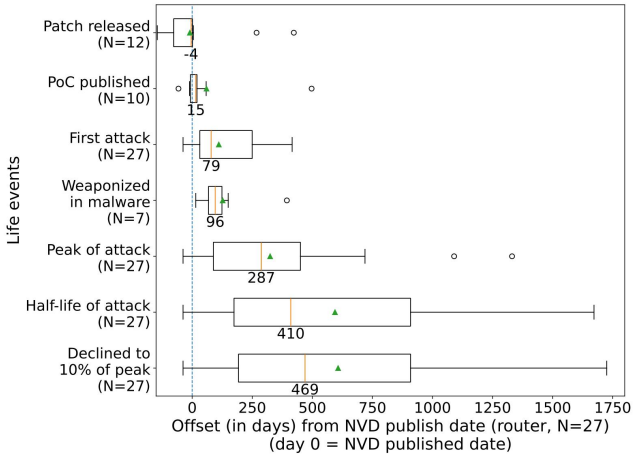


Figure 16: Time offset of life events to router vulnerability disclosure date

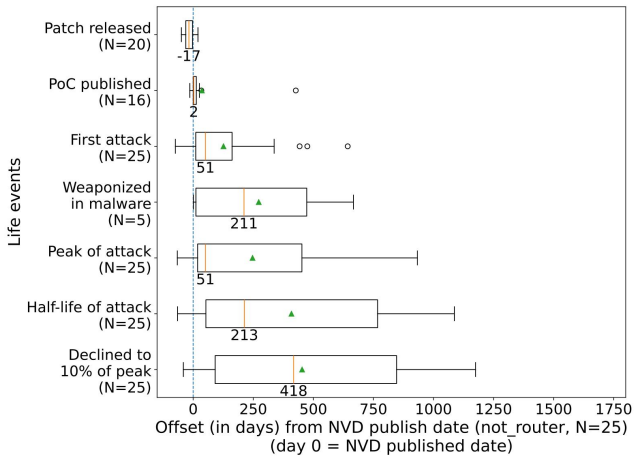


Figure 17: Time offset of life events to non-router vulnerability disclosure date

F Vulnerabilities targeted by the exploits observed by our honeypot

Table 4: Vulnerabilities we have confirmed to be exploited in the wild and status of their life events

CVE No.	PoC	Weaponized in malware	Patch avail.	Auto update
CVE-2008-6554	-	-	-	-
CVE-2009-0545, ...	✓	✓	✓	-
CVE-2009-5157	-	-	-	-
CVE-2010-5330	-	-	✓∗1	-
CVE-2013-2578, ...	✓	-	✓	-
CVE-2013-7179	✓	-	-	-
CVE-2013-7471	-	✓	✓∗1	-
CVE-2013-1599	✓	-	✓	-
CVE-2013-2568	-	-	-	-
CVE-2014-2321	✓	✓	✓	-
CVE-2014-8361	✓	✓	✓	-
CVE-2014-8356, ...	✓	✓	✓	-
CVE-2014-3206	-	✓	-	-
CVE-2015-2051	✓	✓	✓	-
CVE-2015-2781	✓	-	-	-
CVE-2016-6277	✓	✓	✓	-
CVE-2016-1555	✓	-	✓	-
CVE-2016-10372	-	✓	-	-
CVE-2016-11021	-	-	✓	-
CVE-2017-6077	✓	✓	✓	-
CVE-2017-6334	✓	-	✓	-∗2
CVE-2017-6884	✓	✓	✓	-
CVE-2017-8221, ...	✓	✓	-	-
CVE-2017-14135	-	✓	-	-
CVE-2017-14127, ...	✓	✓	-	-
CVE-2017-17215	✓	✓	-	-
CVE-2017-18368	-	✓	✓∗1	-
CVE-2018-10818	-	-	-	✓
CVE-2018-10562	✓	✓	-	-
CVE-2018-10561, ...	✓	✓	-	-
CVE-2018-6961	✓	-	✓∗1	-
CVE-2018-14933	✓	-	✓	-
CVE-2018-15887	-	✓	-	-
CVE-2018-13307	-	-	-	-
CVE-2018-20841	-	✓	-	-
CVE-2019-10039, ...	✓	-	-	-∗2
CVE-2019-12780	-	-	-	-
CVE-2019-7256	✓	✓	✓∗1	-
CVE-2019-12986	-	-	✓	-
CVE-2019-12988	-	-	✓	-
CVE-2019-12985	-	-	✓	-
CVE-2019-12987	-	-	✓	-
CVE-2019-16920	✓	✓	-	-∗2
CVE-2019-18385, ...	✓	-	✓∗1	-∗2
CVE-2019-19824	-	✓	-	-
CVE-2019-19356	✓	-	-	-
CVE-2020-8515	✓	✓	✓	✓
CVE-2020-9054	✓	✓	✓	-∗2
CVE-2020-10173	✓	✓	-	-
CVE-2020-5722	✓	✓	✓	-
CVE-2020-15348	-	-	-	✓
CVE-2020-5902	✓	✓	✓	-
CVE-2020-10987	-	-	-	-∗2
CVE-2020-28188	-	-	✓	✓
CVE-2020-25506	-	-	-	-
CVE-2021-22986, ...	✓	✓	✓	-
CVE-2021-20090, ...	-	✓	✓	-
CVE-2021-31755	-	✓	✓∗1	-
CVE-2021-33357	-	-	✓	-
CVE-2021-36380	-	-	✓	-
CVE-2021-35395	-	✓	✓	-
CVE-2021-33544	-	✓	✓∗1	-
CVE-2021-36260	✓	-	✓	-
CVE-2021-20038	✓	-	✓	-
CVE-2021-20166, ...	-	-	✓	-
CVE-2021-45382	-	-	-	-
CVE-2021-45837	-	-	✓∗1	✓
CVE-2021-46422, ...	✓	-	✓∗1	-
CVE-2022-22947	✓	-	✓	-
CVE-2022-30525	✓	-	✓	✓
CVE-2022-29303	✓	-	✓	-
CVE-2022-2488	-	-	-	-
CVE-2022-36267	✓	-	-	-
CVE-2022-36553	✓	-	✓	-
CVE-2022-40005	✓	-	✓	-
CVE-2022-44149	✓	-	-	-
CVE-2022-39986	✓	-	✓	-
CVE-2023-23333	✓	-	✓	-
CVE-2023-25717	-	-	✓	-
CVE-2023-28343	✓	-	-	-
CVE-2023-27240	-	-	-	-∗2
CVE-2023-26802	-	-	-	-
CVE-2023-26801	-	-	-	-∗2
CVE-2023-27076	-	-	-	-
CVE-2023-27997	✓	-	✓	-
CVE-2023-3608	-	-	-	-
CVE-2023-49897	-	-	✓	-
CVE-2023-47565	-	-	✓	-∗2
CVE-2023-6895	✓	-	✓∗1	-
CVE-2023-46805, ...	✓	-	✓	-
CVE-2024-0778	-	-	-	-
CVE-2024-30891	-	-	-	-
CVE-2024-3721	-	-	-	-
CVE-2024-29973	✓	-	✓	-∗2

∗1: Security patches/firmware have been released to fix the vulnerability, but the release date is unknown.
∗2: The device automatically checks for new firmware and notifies the user, but applying it requires user interaction.