IEICE TRANS. INF. & SYST., VOL.E103–D, NO.7 JULY 2020

PAPER Special Section on Information and Communication System Security

1-day, 2 Countries—A Study on Consumer IoT Device Vulnerability Disclosure and Patch Release in Japan and the United States*

Asuka NAKAJIMA[†a)], **Takuya WATANABE**[†], ***Nonmembers***, **Eitaro SHIOJI**[†], **Mitsuaki AKIYAMA**[†], ***Members***, ***and*** **Maverick WOO**[††], ***Nonmember***

SUMMARY With our ever increasing dependence on computers, many governments around the world have started to investigate strengthening the regulations on vulnerabilities and their lifecycle management. Although many previous works have studied this problem space for mainstream software packages and web applications, relatively few have studied this for consumer IoT devices. As our first step towards filling this void, this paper presents a pilot study on the vulnerability disclosures and patch releases of three prominent consumer IoT vendors in Japan and three in the United States. Our goals include (i) characterizing the trends and risks in the vulnerability lifecycle management of consumer IoT devices using accurate long-term data, and (ii) identifying problems, challenges, and potential approaches for future studies of this problem space. To this end, we collected all published vulnerabilities and patches related to the consumer IoT products by the included vendors between 2006 and 2017; then, we analyzed our dataset from multiple perspectives, such as the severity of the included vulnerabilities and the timing of the included patch releases with respect to the corresponding disclosures and exploits. Our work has uncovered several important findings that may inform future studies. These findings include (i) a stark contrast between how the vulnerabilities in our dataset were disclosed in the two markets, (ii) three alarming practices by the included vendors that may significantly increase the risk of 1-day exploits for customers, and (iii) challenges in data collection including crawling automation and long-term data availability. For each finding, we also provide discussions on its consequences and/or potential migrations or suggestions.

key words: *consumer IoT, vulnerability disclosure, patch, exploit, measurement*

1. Introduction

As our society continues to increase its reliance on computers large and small, vulnerabilities and their lifecycle management are gradually becoming a matter of public safety. In response, many governments around the world have started to investigate regulating and improving computer security through legislation and standards setting. For example, the National Telecommunications and Information Administration in the United States has started a multistakeholder process since 2015 in regards to Internet of Things (IoT) security upgradability and patching [2]; similarly, the Ministry of Internal Affairs and Communications in Japan has also initiated a task force in 2017 aimed to study and improve the IoT security posture of Japan [3]. As we may expect, understanding the past and current practices of the stakeholders in IoT vulnerability lifecycle management and identifying potential improvements in these practices are both important steps towards improving computer security and public policies.

Existing work on *vulnerability lifecycle management* can be classified into two groups based on where it falls with respect to the time of discovery of a vulnerability. At a high level, work in the pre-discovery group focuses on the prevention and discovery of vulnerabilities, whereas work in the post-discovery group focuses on the disclosure and notification of vulnerabilities as well as mitigations. In large part due to the myriad of ways and places for vulnerabilities to creep in and the depth and breadth of vulnerability prevention and discovery techniques, the literature in the pre-discovery group is vast and continues to expand rapidly[**]. In comparison, the literature in the post-discovery group is considerably smaller and it contains two lines of work that are particularly relevant to this paper. The first line studies the patch release behavior of commercial and/or open-source developers (collectively referred to as "vendors" in this paper). For example, previous studies have investigated the timeliness and prioritization of patch releases, behavioral differences between commercial and open-source vendors, and external factors that may improve their behaviors [7]–[13]. The second line studies the characteristics of patches and vulnerabilities. This includes the measurements of diverse properties such as the number of exploitations, the longevity of vulnerabilities, the size and complexity of their patches, and the rate of patch deployments [14]–[19].

Incidentally, the overwhelming majority of the aforementioned studies were dominated by mainstream software packages and web-based applications. Based on our literature search, we are not aware of any prior work that focused on *consumer IoT devices*, many of which are products by vendors that have a small or even non-existent representation in the datasets used by previous studies. With the rapid rise of consumer IoT devices in recent years, we believe it is high time to expand our knowledge in vulnerability lifecycle management in regards to these devices and their vendors.

Manuscript received August 2, 2019.
Manuscript revised January 18, 2020.
Manuscript publicized March 24, 2020.

[†]The authors are with NTT Secure Platform Laboratories, Musashino-shi, 180–8585 Japan.
[††]The author is with Carnegie Mellon University, Pittsburgh, PA 15213, United States.

*An earlier version of this paper [1] had been presented at ACM AsiaCCS 2019.

a) E-mail: asuka.nakajima.db@hco.ntt.co.jp

DOI: 10.1587/transinf.2019ICP0004

**As examples, the reader may refer to relevant surveys on vulnerability discovery techniques (e.g., [4]–[6]) and recent conference proceedings in the fields of computer security and software engineering.

As our first step towards filling this void, this paper presents a pilot study on the vulnerability disclosures and patch releases of prominent consumer IoT vendors. A key novelty in our study is the recognition that, even though we are in a global economy, consumer purchase decisions in different countries are heavily *localized*. Therefore, individual markets may show entirely different trends and risks, and yet these trends and risks may also be correlated across markets due to global trade. To cater for these possibilities, we have thus decided to allocate our effort by markets and analyze them accordingly. In this paper, we will present our study using data covering three prominent consumer IoT vendors in Japan and three in the United States as well as their geographical subsidiaries in Australia, China, and Germany (Table 1).

As we will explain in Sect. 3, we selected the included vendors by approximating consumer perception of prominence in the two markets. Specifically, our procedure prioritized vendors with the most number of results when searching for the phrase "wireless routers" at a popular shopping website in each market. Since our preliminary data analysis indicated the need for manual investigations to fill in missing data fields, we ended up selecting three out of the top four vendors in each site due to the reasons given in Sect. 3.2.

After the vendor list was fixed, we collected the vulnerability disclosures of all consumer IoT devices by the included vendors. This involves retrieving all CVE entries [20] of these vendors from the National Vulnerability Database (NVD) [21] and keeping only those that are about consumer IoT devices. In addition, we also heavily depended upon the Japan Vulnerability Notes (JVN) iPedia [22]. For example, the NVD does not provide the public disclosure date of a vulnerability; however, perhaps a little-known fact outside of Japan, the JVN analysts have determined this information in a best-effort manner for each vulnerability in the JVN. By combining these two sources, public exploit databases, and other sources such as vendor security advisories, release notes, and mailing list discussions, we were able to *complete* each included CVE entry to contain (i) affected product name, (ii) affected version, (iii) patched version, (iv) disclosure date, (v) exploit release date, and (vi) patch release date (Fig. 3). Our effort has resulted in 53 and 230 completed CVE entries for respectively Japan (JP) and the United States (US), covering the 12 years from 2006 to 2017.

In Sect. 4, we will see that our inclusion criteria yielded surprising insights that may likely be missed if we did not include vendors from multiple markets and stratify our analyses accordingly. For example, we found that 86.8% of the vulnerability disclosures in our JP dataset were coordinated disclosures. Not only is this percentage astonishingly high, but it is also significantly higher than the corresponding US figure of 20.1%. We will drill into this difference and offer our observations about it in Sect. 4.3.

While analyzing the collected release notes and completing the missing information about the included vulnerabilities, we also uncovered three alarming vendor practices that may significantly increase the risk of 1-day exploits for customers due to patch-based exploit generation. First, we discovered all six vendors practiced *incremental patch release*, which refers to releasing a series of similar patches over time where each patch addresses the same vulnerability but in different device(s). This means a device patched later in a series would face an increased risk of 1-day exploits (Sect. 4.2.3). Second, we noticed that some vendors would release a patch in one geographic region earlier than in other regions. This leaves ample opportunities for what we dubbed *geographical arbitrage*—the potential to generate 1-day exploits by using a patch that was already released in another region (Sect. 5.1). Third, we observed that some vendors would seemingly stop releasing patches to a device in one region without making any End-of-Support announcement but continue to do so in other regions. This also leads to a security risk similar to that of geographical arbitrage (Sect. 5.2). For all three practices, we will provide examples either in the form of a known exploit or a vulnerability analysis to show that their risks can become real.

In summary, the major contributions of this paper are:
1. We selected three prominent consumer IoT device vendors in Japan and three in the United States and collected information on the vulnerabilities and the patches of their consumer IoT devices between 2006 to 2017. To ensure our data has high quality, we completed each of the 283 included CVE entries by cross-referencing the NVD, JVN, security advisories, release notes, and search engine results.
2. We characterized the vulnerability disclosures related to the included devices and the patch release behaviors of the included vendors. Our bi-country dataset enabled us to look for significant differences using both the inter-vendor and inter-market perspectives. The latter has revealed a stark contrast between how the vulnerabilities in our dataset were disclosed in the two markets.
3. Our study uncovered three alarming vendor practices that may significantly increase the risk of 1-day exploits for customers. For each practice, we demonstrate that its risk can become real either by identifying a known exploit or by providing our own vulnerability analysis to a relevant vulnerability.

2. Background

In this section, we will briefly review the concept of vulnerability lifecycles and its associated terminology. We will define the *roles* involved in the lifecycle events, the *events* themselves, and the classification of *exploits* based on the event ordering. Since we have chosen to use a terminology consistent with [23], we acknowledge that some of the definitions below contain clearly-marked quotes from [23].

2.1 Roles

Figure 1 illustrates the roles involved in the events on a vulnerability lifecycle and the relationships between these

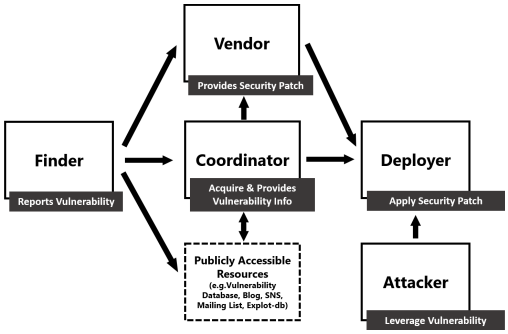


Fig. 1 Roles in the post-discovery phases of a vulnerability lifecycle. Each solid box represents a role, with brief explanation attached. The dashed box represents information that has been made public. An arrow represents a communication or an action among these entities.

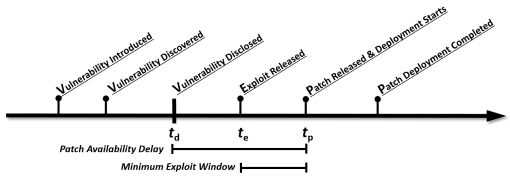


Fig. 2 One possible timeline of a vulnerability lifecycle. This study focuses on the *patch availability delay* $(t_p - t_d)$ and the *minimum exploit window* $(t_p - t_e)$, where t_d is the date of public vulnerability disclosure, t_p is the release date of the corresponding patch, and t_e is the release date of the earliest exploit against the vulnerability known to us. Note that either or both of these time differences can be negative in a timeline where the patch was released earlier than shown here.

roles.

Vendor is an individual or an organization that created or maintains a product. When a vulnerability is discovered in a product, the vendor is "the party responsible for updating the product containing the vulnerability" [23, §3.3] . This generally entails releasing a new firmware of the product either in whole or as a patch. For simplicity, we call both a *patch release*.

Coordinator is an individual or an organization that "acts as a relay or information broker between other stakeholders" [23, §3.5] . Nowadays, coordinators come in many forms, including Computer Security Incident Response Teams (CSIRTs), Product Security Incident Response Teams (PSIRTs), commercial brokers, third-party bug bounty programs, etc. A well-known coordinator is the CERT Coordination Center (CERT/CC) [24].

Finder is "an individual or an organization that identifies a potential vulnerability in a product or online service" [23, §3.1] . In general, a finder can be anyone, including users, vendor developers, third-party researchers, etc. When a finder is not internal to the vendor, this finder gets to decide whether to disclose the vulnerability, and if so, when and to whom. Specifically, a finder can decide to not disclose the vulnerability, or to privately disclose it to the vendor or a coordinator, or to publicly disclose it. Note that these three possibilities are not mutually exclusive because the decision of a finder may change over time.

Deployer is an individual or an organization that is "responsible for the fielded systems that uses or otherwise depend on products with vulnerabilities" [23, §3.4] . For example, the deployer can be a user of the product or a system administrator in a corporation.

Attacker is an individual or an organization that leverages a vulnerability in a product to perform malicious activities. Note that an attacker needs not be a finder of a vulnerability due to the availability of commercial or open-source exploit kits.

2.2 Events

Figure 2 depicts one possible timeline of a vulnerability lifecycle, on which there are six events: (i) the vulnerability is introduced by the vendor, (ii) the vulnerability is discovered by a finder, (iii) the vulnerability is publicly disclosed by a finder, a coordinator, or the vendor, (iv) the corresponding exploit is released, (v) the corresponding patch is released, and (vi) the patch reaches complete deployment. Note that while the first two events on any timeline are always (i) and (ii), the remaining four can happen in any order so long as (v) happens before (vi). In addition, there may also be multiple finders, further complicating the timeline.

In this paper, we denote the *public disclosure date* by t_d, the *patch release date* by t_p, and the *exploit release date* by t_e. The time difference $(t_p - t_d)$ is the *patch availability delay* and ideally it should be small or even negative. Similarly, $(t_p - t_e)$ is the *minimum exploit window*, which measures the time between patch availability and the release of the earliest exploit known to us. Note that the actual exploit window may be bigger than $(t_p - t_e)$ because an exploit unknown to us could have been available at a time before t_e.

2.3 Exploit Classification and 1-Day Exploits

Using the event time points as defined above, a "0-day exploit" is an exploit for which t_e is before t_d. Such exploits are often considered to be the most dangerous because they can be extremely difficult to detect and prevent. In contrast, a "1-day exploit" is an exploit for which t_e is on or after t_d. In practice, attackers often actively seek such exploits post-disclosure in the hope to launch attacks before deployers deploy the corresponding patches.

Naturally, this latter type of attackers will be greatly assisted if an actual exploit or a full analysis of a vulnerability has been published. However, even the release of a patch by a vendor carries a significant risk of assisting such attackers because they may be able to reverse-engineer the details of a vulnerability from its patch. This technique is called patch-based exploit generation and it may even be automated in some case (e.g., [25]). As we will see in this paper, certain forms of patch release behavior exhibited by the included vendors may end up giving attackers more time to exercise this technique, which in turns increases the risk of 1-day exploits faced by customers.

3. Data Collection Methodology

Since a primary objective of our study is to characterize the trends and risks in the vulnerability lifecycle management of consumer IoT devices, a significant portion of our effort was spent in collecting accurate information on the vulnerability disclosures and patch releases related to the included products. To this end, we started by automated crawling and cross-referencing of multiple sources, including the NVD, JVN, security advisories, patches, and release notes. Afterwards, we inspected every remaining empty field and filled in the missing information using manual investigations. The end result is a dataset where *every* included CVE entry has been completed with the six pieces of crucial information described below. In this section, we will detail our data collection process and justify our choices throughout. Our overall data collection methodology is depicted in Fig. 3.

3.1 Identify Target Countries

Our study initially *did not* distinguish among countries and our plan was to include as many vendors as our budget allowed by selecting vendors with the most number of consumer IoT products. However, we quickly noticed that the vendors selected by this method were strongly US-centric and this raised questions regarding the applicability of our findings to non-US markets. For example, although Netgear ranked 2nd in the US using our procedure described in Sect. 3.2, it ranked only 8th in Japan and was preceded by prominent Japanese vendors such as Buffalo and IO-DATA. This gave us the inspiration to conduct a novel study that explores individual markets separately and look for differences. In the end, we settled on the markets of Japan (JP) and the United States (US) because of the following factors.

First, JP and US are large economies—both are in the top five in every list of countries ranked by the Gross Domestic Product (GDP), whether nominally or after adjustments for purchasing power parity. This means they are both important markets to study in regards to consumer IoT devices. Second, they have two of the oldest national CSIRTs, namely CERT/CC (est. 1988) and JPCERT/CC (est. 1996). This means we may reasonably expect them to have the most mature vulnerability coordination among all markets. Third, our team is highly familiar with both JP and US and we have members who are fluent in the languages of these markets. Thus, we were able to leverage our language skills to perform in-depth studies of the consumer IoT devices in these markets even when some of the relevant information was available only in the local language.

From this point on, "our dataset" in this paper refers to our combined dataset and we will call out a specific subset when needed, e.g., "the JP dataset".

3.2 Identify Target Vendors

Early on in our study, we were already aware that we did not have enough resource to include every consumer IoT vendor in these markets. Not only were there a large number of them given the broad interpretation of "consumer IoT devices", but we also noticed the crawled data contained a non-trivial number of missing fields that required manual investigations. Therefore, we decided to prioritize prominent vendors and included as many of them as our budget allowed. To this end, we used the number of wireless routers offered by a vendor to approximate its prominence and included *every* consumer IoT devices by these vendors in our study.

While not perfect, we believe this procedure is sufficiently justified. First, since consumers tend to buy from familiar brands and consumers are already familiar with wireless routers due to their ubiquity, we believe vendors with a large number of wireless routers are prominent. Second, we recognize that vendors offering wireless routers also tend to offer many other consumer IoT devices. Indeed, over 29% of the 450 products in our final dataset comprises products in other categories. These include networked surveillance cameras (e.g., Buffalo WNC01WH), NAS (e.g., IO-DATA HDL-F160), smart plugs (e.g., D-Link DSP-W215), smart switches (e.g., D-Link DGS-1500), and VPN appliances (e.g., Netgear SSL312). Thus, we believe our dataset well-approximates a large portion of consumer IoT devices.

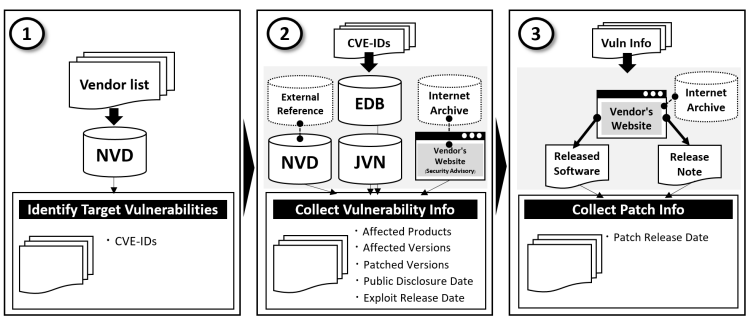


Fig. 3 Overview of Data Collection Methodology. (1) Identify relevant vulnerabilities related to included vendors using the NVD—Sect. 3.3. (2) Collect vulnerability information, including disclosure dates, affected products, affected versions, patched versions, and exploit release dates from related databases—Sect. 3.4. (3) Collect patch release information via vendor websites and patch releases—also Sect. 3.4.

Our actual vendor selection procedure is as follows. First, using our understanding of the two markets, we decided to use Kakaku.com and Amazon.com as a representative shopping site for JP and US respectively. Then, we ranked the vendors using the number of products returned to the query "wireless routers" on the two sites[†]. Table 1 shows the top four vendors in our search conducted in April 2018.

Our original plan was to include the top three vendors in each market. However, we ended up selecting three out of the top four vendors in both. In JP, the 2nd-ranked vendor was ELECOM, which is indeed a well-known consumer device vendor in Japan. However, we discovered that ELECOM (surprisingly) did not have any CVE entry. Therefore, we extended the JP dataset to include the 4th-ranked vendor NEC. In US, the top-ranked vendor was Synergy Digital. However, this vendor also did not have any CVE entry because it mainly sells batteries for other devices and many of its product listings happened to include the phrase "wireless routers". Therefore, we excluded this vendor and extended the US dataset to include the 4th-ranked vendor D-Link.

We note that our data collection effort also included a few geographical subsidiaries of the included vendors. The data from these subsidiaries will be explained and used in Sect. 5.

3.3 Identify Target Vulnerabilities

Having selected the vendors, we used the NVD to identify all CVE entries related to the consumer IoT devices of these vendors in three steps. First, we retrieved all CVE entries involving these vendors from the NVD. While this step is straightforward in principle, in practice we had to screen for *orthographical variants* of vendor names. Using the vendor I-O DATA as an example, we were able to discover three different spellings in the NVD vendor list: "io-data", "i-o_data", and "i-o_data_device". Table A· 1 in Appendix A shows our best-effort screening result of the name variants of the included vendors. Second, we manually excluded non-consumer-IoT entries, which explains the drop in #CVE-IDs between Tables 1 and 2. For example, we excluded **CVE-2017-2137**, which pertains to an access restriction bypass in Netgear ProSAFE Plus Configuration Utility and is not related to a vulnerability in a consumer IoT device. Third, we restricted our dataset to CVE entries in the 12 years 2006–2017 due to a data availability issue. Specifically, as we will explain below, we relied on the Japan Vulnerability Notes (JVN) iPedia [22][†††] for several pieces of crucial information related to each included CVE ID. However, the JVN feed appears to be curated since 2006 only and thus we were not able to include any earlier year.

Table 1 Summary statistics of the included vendors. The #CVE-IDs column counts all 2006–2017 CVE entries that involve the indicated vendor and not just its consumer IoT devices.

Country	Vendor	Ranking	#CVE-IDs	Inclusion
Japan	Buffalo	#1	22	✓
	ELECOM	#2	0	
	IO-DATA	#3	29	✓
	NEC (Aterm[††])	#4	3	✓
United States	Synergy Digital	#1	0	
	Netgear	#2	56	✓
	Linksys	#3	56	✓
	D-Link	#4	142	✓

[†]The exact URLs used were, respectively, https://kakaku.com/pc/wireless-router/ and https://www.amazon.com/s/other?k=wireless%20routers&rh=n%3A172282%2Cn%3A541966%2Cn%3A172504%2Cn%3A300189&pickerToList=lbr_brands_browse-bin.

[††]Aterm is the brand name of wireless routers by the vendor NEC.

3.4 Collect Vulnerability & Patch Information

After obtaining all CVE entries related to the consumer IoT products of the included vendors, we proceeded to collect the following six pieces of information about each included vulnerability.

(1) Affected Products and (2) Versions. Starting with a CVE entry from the NVD, we extracted the affected product names and versions from respectively the `product_name` and `version_value` fields. Although we were always able to extract the former from the NVD, we encountered a number of empty fields for the latter, which mirrors a similar experience reported in [26]. To combat this, we cross-referenced every included CVE entry from the NVD with its corresponding entry in the JVN and extracted the missing information from the latter, if available.

In addition, we also extracted affected product names and versions from vendor security advisories, which were published by the vendors to inform their customers regarding product vulnerabilities. We obtained these advisories by searching for the CVE IDs on search engines and retrieving from the vendor websites, or from the Internet Archive [27] if a desired page was no longer available. In the cases where the information extracted from these advisories differed from that in the NVD and JVN, we employed the former because we believe the vendor is better-positioned to provide the most accurate information. Finally, in the rare cases where we did not obtain any version information from any of the above sources, we relied on search engines to identify external references such as blog posts and mailing list archives for manual extraction.

(3) Patched Versions. The procedure to collect patched versions started off similarly to that of (1) and (2), i.e., we first extracted from the NVD, JVN, and vendor security advisories. Then, we performed further extraction from the release notes in the patches released by the vendors and, in a few cases, resorted to manual investigations using search engines.

(4) Public Disclosure Date. The public disclosure date of a

[†††]JVN is a national vulnerability database in Japan, which is organized by JPCERT/CC and Information-technology Promotion Agency (IPA). The JVN website contains multiple data feeds. In this paper, "the JVN feed" to refer to the JVN iPedia feed.

vulnerability, denoted t_d, is the earliest date when the vulnerability was disclosed in a publicly-accessible resource such as vulnerability databases, blog posts, or public mailing lists. Unfortunately, the NVD does not contain this information. However, perhaps a little-known fact outside of Japan, JPCERT/CC and Information-technology Promotion Agency (IPA) have dedicated resource to curate this information for every entry in the JVN [28]. Thus, we relied on the JVN for the public disclosure date of the included vulnerabilities.

(5) Exploit Release Date. For each included vulnerability, we also attempted to collect the earliest release date of a publicly-known exploit if one was available in either Exploit Database (EDB) [29] or Metasploit [30]. We denote this date by t_e in this paper and it is our best-effort estimation of the exploit release date. EDB is an archive site of public exploits and can be searched using CVE ID. For this source, we used a tool called `cve_searchsploit` [31] to search and collect the exploit release dates from EDB. Metasploit is a penetration testing platform which includes exploit code for many known vulnerabilities. For this source, we collected exploit release dates from the public database organized by Rapid 7 [32].

(6) Patch Release Date. During the extraction of patched versions in (3), we also collected the patch release date t_p and one of our sources was release notes. To our surprise, some release notes either did not contain a date, or in the case of Buffalo US, contained dates of the source documents from which the notes were translated. In these cases, we resorted to using the first date among the following as a best-effort estimation: (i) the release date listed by the vendor website, or the Internet Archive if a desired page was no longer available, (ii) the file date of the release note if it was in an archive, (iii) the file date of the archive itself, and (iv) the latest date of the other files in the same archive, unpacked recursively using binwalk [33].

3.5 Limitations

Before we continue to our data analysis in Sect. 4, we will discuss several known limitations to our dataset here.

Public Data Only. We are aware that our data contains only publicly-available vulnerabilities and that when a vendor fixes a vulnerability that was found internally, the vendor is not required to create a CVE entry nor to mention there was a vulnerability in any document. In addition, at the time of data collection, some vulnerabilities might be under embargo or in the backlog due to limited analyst resource. Thus, the number of CVE entries in our dataset is only a lowerbound on the number of vulnerabilities that had been found at the time of data collection. Similarly, we depended upon Exploit Database and Metasploit to collect the exploit release dates. Our reliance on these public databases means (i) we likely have collected fewer exploit release dates because these two databases are not exhaustive, (ii) our exploit release dates can be later than actual even considering only public sources, e.g., an exploit could be released in a mailing list before making its way into a database, and (iii) our exploit data may contain regional bias due to the English-dominance in the databases we used.

Data Quality. Although we have dedicated much effort to ensure data quality through cross-referencing and in some cases manual investigations, our extracted data can only be as accurate as its sources. For example, if a release note contained a typographical error in its date, our data would inherit that error. In addition, as explained in Sect. 3.4, we have resorted to estimating the patch release date using file dates for release notes that do not contain a date.

Vendor and Product Bias. Our vendor selection method favors vendors with many wireless router products. Although our personal experience informs us that the vendors included in our study are indeed prominent consumer IoT vendors, nowadays the consumer IoT markets are filled with many niche but by no means uncommon products such as smart light-bulbs, smart refrigerators, etc. We are aware that niche vendors focusing on these products were not selected, and that these products are not well-represented in our dataset because the included vendors do not offer these products.

4. Analysis of Patch Release Behavior

In this section, we will start with an overview of our dataset and provide statistics for it (Sect. 4.1). Then, we will analyze the timing of the included patch releases and characterize the included vendors based on their patch release behaviors (Sect. 4.2). Finally, we will investigate the differences between the JP and US statistics in our study by demonstrating a stark contrast between how the vulnerabilities in our dataset were disclosed in the two markets (Sect. 4.3).

4.1 Dataset Overview

As explained in Sect. 3, we selected three prominent consumer IoT device vendors in Japan and three in the United States and collected information on the vulnerabilities affecting their consumer IoT devices and the corresponding patches. Table 2 summarizes our dataset. In total, our dataset contains 283 CVE entries spanning 2006–2017. Compared with Table 1, we dropped (i) 15 entries that are unrelated to consumer IoT devices and (ii) 10 entries that are not in the JVN and that we are unable to determine their disclosure dates using other sources. The included vulnerabilities affect 450 products and the vendors released 551

Table 2 Dataset summary. The number inside brackets is #CVE-IDs where we were able to find at least one t_p date.

Country	Vendor	#CVE-IDs				#Products	#Patches	#Exploits
		Total	Low	Medium	High			
JP	Buffalo	22(20)	1(1)	17(15)	4(4)	71	105	0
	IO-DATA	28(24)	5(3)	13(11)	10(10)	57	88	0
	NEC (Aterm)	3(3)	0(0)	3(3)	0(0)	26	35	0
JP Total		53(47)	6(4)	33(29)	14(14)	154	228	0
US	Netgear	43(25)	1(1)	17(12)	25(12)	107	106	21
	Linksys	52(17)	4(2)	18(3)	30(12)	31	40	12
	D-Link	135(61)	9(6)	74(27)	52(28)	158	177	35
US Total		230(103)	14(9)	109(42)	107(52)	296	323	68
Total		**283(150)**	20(13)	142(71)	121(66)	**450**	**551**	**68**

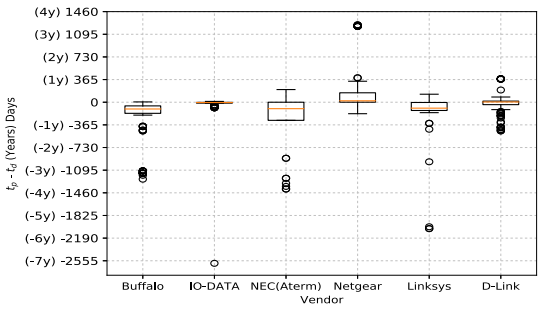


Fig. 4 Box-plots of patch availability delay ($t_p - t_d$) of the included vendors.

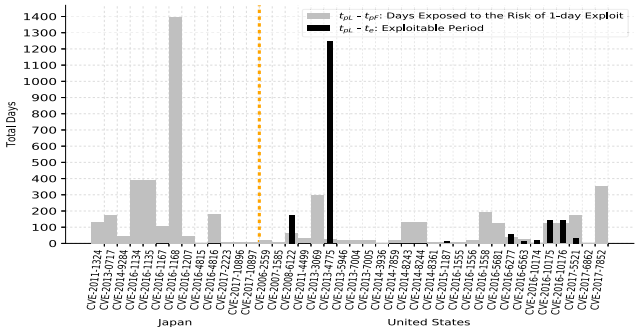


Fig. 5 Number of days where a device was exposed to exploitation risk due to incremental patch releases. This risk materialized in **CVE-2016-6563** and **CVE-2017-5521**.

patches against them. We also collected 68 exploits against these vulnerabilities from Exploit-DB and Metasploit.

In addition, we also classified the included CVE IDs using the Common Vulnerability Scoring System (CVSS), a *de facto* standard in measuring the severity of software vulnerabilities using a score between 0 (least severe) and 10 (most severe). Although the current version of CVSS is v3 released in 2015, our study employed CVSS v2 because CVE entries before 2015 do not contain CVSS v3 scores. Based on CVSS v2, the severity of vulnerabilities can be classified in three categories: Low (0.0–3.9), Medium (4.0–6.9), and High (7.0–10.0). The number of included CVE IDs in each category is shown in columns 4 to 6 of Table 2. We see that vulnerabilities of medium severity form the majority in both our JP (62%) and US (47%) datasets. In addition, high-severity vulnerabilities also accounted for a significant percentage in both datasets (26% and 47%, respectively).

4.2 Characterizing Patch Releases and Vendor Behaviors

In this subsection, we will analyze the patch releases and characterize the vendor behaviors in our dataset by focusing on the 150 CVE IDs of which we have collected at least one patch release date (see Table 2). Throughout this subsection, we will pay special attention to identify differences between the JP and US datasets. In Sect. 4.3, we will provide our explanation for several of the identified differences.

4.2.1 Patch Availability Delay

Figure 4 shows the patch availability delay ($t_p - t_d$) of each included vendor as a box-plot, produced with `matplotlib.pyplot.boxplot` in Python Matplotlib 2.0.0 using the default settings[†]. Using Fig. 4, we classify the vendors into three categories based on when a vendor tends to release security patches: (1) *around*, (2) *before*, and (3) *after* the public disclosure date. First, we classify IO-DATA and D-Link as (1) because their boxes are small and tightly concentrated around $y = 0$. In our dataset, the median patch availability delay of IO-DATA and D-Link are respectively -5 and 4 days. Second, we classify Buffalo, NEC (Aterm), and Linksys as (2) because their boxes are located underneath $y = 0$. Incidentally, the median delay for these three vendors are all close to -100 days in our dataset. Finally, we classify Netgear as (3) because its box is located above $y = 0$. The median delay for Netgear in our dataset is 23 days.

[†]Each box-plot visualizes the distribution of ($t_p - t_d$) values for each vendor. In particular, each box extends from the 1st to 3rd quartile, with the median marked by the band. The whisker extends from the lowest to the highest value within 1.5 IQR, where IQR is defined to be the difference between the 1st and 3rd quartile, and the circles represent outliers beyond the whisker.

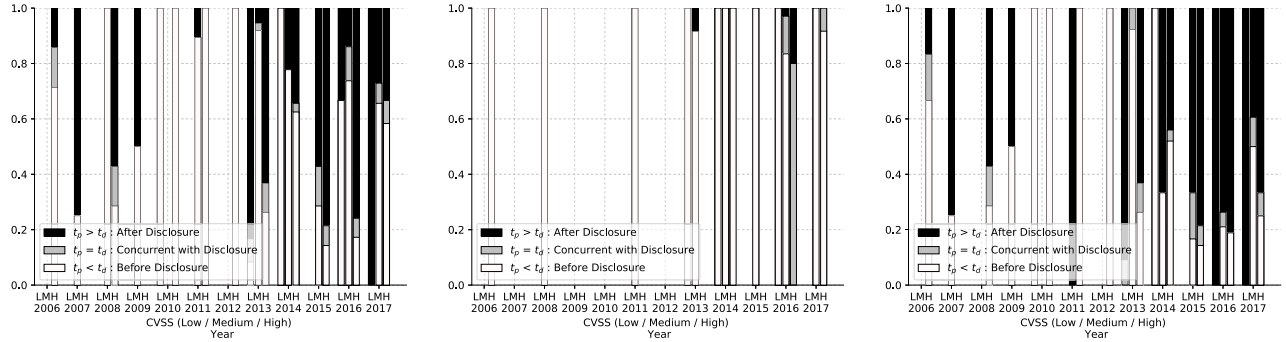
4.2.2 Minimum Exploit Window

Closely related to the above is the minimum exploit window ($t_p - t_e$) of the included vulnerabilities. In our dataset, we found seven exploits that were released before their corresponding patches. Although these seven exploits are all in the US dataset, we remark that this may be because the two exploit databases we used are international and thus may have regional bias in their exploits. (We are not aware of any JP-focused exploit database.)

4.2.3 Incremental Patch Release

Our study of the included patch releases indicates that all six vendors practiced *incremental patch release*, which refers to the release of a series of patches over time where each patch addresses the same vulnerability but in different device(s). This reflects a common phenomenon where many consumer IoT devices share similar software components and thus also vulnerabilities. In our dataset, 62.4% of the patches were released incrementally in a series and they were associated with 40 CVE IDs.

Unfortunately, this vendor behavior carries an inherent security risk. Specifically, whenever a vendor engages in incremental patch release (whether knowingly or unknowingly), attackers may race to discover the vulnerability in similar devices and exploit it before the vendor releases patches for those devices—effectively a form of 1-day exploitation. To characterize this risk, we first measured the number of days between the last and the first patch releases ($t_{pL} - t_{pF}$) for each CVE ID that exhibits incremental patch release in our dataset. These numbers are visualized as the gray bars in Fig. 5. Our measurement shows that the aver-



(a) All (JP + US) (b) Japan (c) United States

Fig. 6 Timeliness of patch releases by the included vendors over 2006–2017, partitioned by vulnerability severity.

age of these numbers is 122.5 days, with a median of 36 days and a maximum of 1399 days.

In addition, for each of these CVE IDs, if the last patch in the series was released after an exploit for the vulnerability appeared, i.e., there was an exploit window ($t_{pL} > t_e$), we also show a black bar for the CVE ID in Fig. 5 to visualize the length of this time gap. In total, we identified nine exploit windows among these CVE IDs and two of these windows started during an incremental patch release ($t_{pF} < t_e < t_{pL}$). The latter means that the 1-day risk had materialized and corresponds to the two CVE IDs where the black bar is shorter than the corresponding gray bar. (While the nine exploit windows identified all reside in the US dataset, please see our remark on potential regional bias in Sect. 4.2.2.)

4.2.4 Patch Release Timeliness

Following [13], we have also attempted to detect any temporal trend in the timeliness of patch releases for vulnerabilities of each CVSS severity (L/M/H). Here we categorize the patch release timings into three categories: (1) pre-disclosure (t_p ¡ t_d), (2) concurrent with disclosure ($t_p = t_d$), and (3) post-disclosure (t_p ¿ t_d).

Figure 6 (a) shows the trend in our dataset. We see that the post-disclosure patches (colored in black) account for a large portion across all three categories in our dataset. Even more troubling, we see no sign of reduction of them over time, suggesting that the vendors had been unable to improve their ability to release patches in advance of disclosures. Interestingly, once we break down our dataset by markets, Fig. 6 (b) and 6 (c) show that the timings of category (3) are largely due to the US dataset. This was what sparked our inter-market investigation in Sect. 4.3.

4.2.5 Patch Release Timing

In addition to investigating patch release timeliness by proportion as we did in Fig. 6, we can also do this by number. Using the same classification as in Sect. 4.2.4, Fig. 7 visualizes our dataset on the left, followed by the breakdowns for JP and US. Focusing on the former, we are encouraged to see that more than 1/2 of the included patches were released before the corresponding vulnerabilities were disclosed. Unfortunately, we also see that about 1/3 of the included patches were released after disclosure. Again, if we break down by markets, we see a stark contrast where the overwhelming majority of the included patches in the JP dataset were released before disclosure.

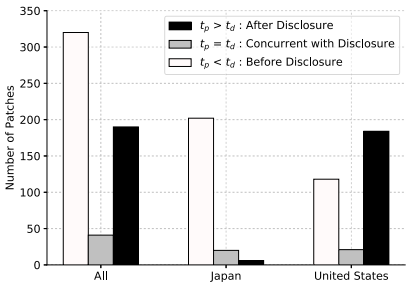
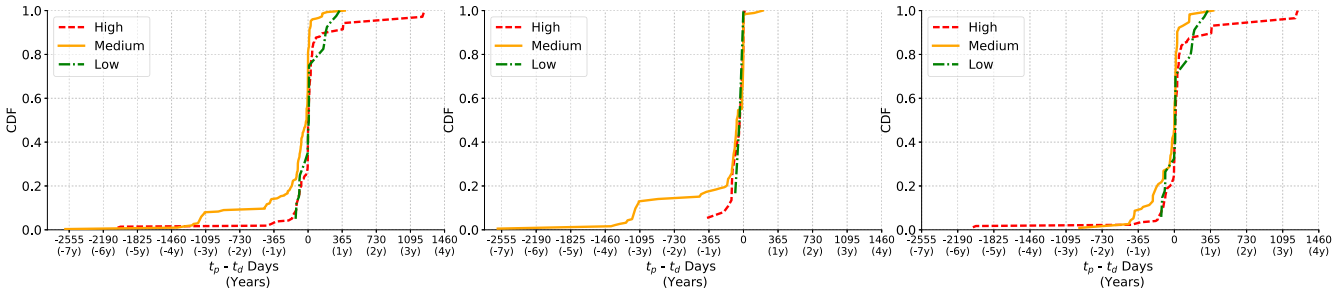


Fig. 7 Number of patches released before, concurrent with, and after disclosure, along with breakdowns for JP and US.

4.2.6 Fix Prioritization

Finally, we have also investigated whether high severity vulnerabilities get patched more promptly. This question was addressed in [19] for open-source software. Our finding is shown in Fig. 8 (a), which shows the cumulative distribution functions (CDFs) of the patch availability delay ($t_p - t_d$) for each severity category (L/M/H) in our dataset. We see that there does not seem to be any prioritization at all. Specifically, the CDF of the delay for high-severity vulnerabilities (red) remains around 0.9 well into one year post-disclosure, while the CDFs of the other two categories have already reached 1.0 at that time. As an extreme example of this lack of prioritization, Netgear took 1247 days to release the patch for **CVE-2013-4775** even though its severity score was 7.8 (High). In addition, we also provide a breakdown by markets in Figs. 6 (b) and 6 (c), although we note that the low number of post-disclosure patches in the JP dataset would not allow us to discern any pattern.



(a) All (b) Japan (c) United States

Fig. 8 CDFs of patch availability delay ($t_p - t_d$) for each vulnerability severity level.

4.3 Contrasting Between Our JP and US Datasets

In Sect. 4.2, we witnessed that the overwhelming majority of the patches in our JP dataset were released either concurrently or before public disclosures, whether in relative proportion (Fig. 6) or in absolute number (Fig. 7). That strongly hinted us to discover one of the unexpected findings of this study. Specifically, given the similar technological maturity of the two countries, how did the included JP vendors attain such marvelous statistics? Inspecting our JP dataset, we found that only 3 of the 53 included CVE IDs were filed by a vendor. This ruled out the possible explanation that the three included JP vendors tended to disclose internally-discovered vulnerabilities after releasing the corresponding patches, which would have tilted the statistics in favor of them over their US counterparts. Therefore, we were left with the hypothesis that somehow the finders in our JP dataset tended to perform coordinated disclosures. To check this hypothesis, we have thus classified every disclosure in our dataset using the following procedure.

4.3.1 Classification Categories

As a first step, we have decided to classify the vulnerability disclosure process of each of the 283 included CVE entries into the following three categories:

(1) Coordinated Disclosure. This is when we have evidence that the detail of the vulnerability was publicly disclosed on or after a patch of it was released.

(2) Full Disclosure. This is when we have evidence that the detail of the vulnerability was publicly disclosed before a patch of it was released.

(3) Unknown. This is when we were unable to identify any patch release date (t_p) for the vulnerability, or when we do not have evidence of whether the disclosure was in one of the two categories above.

4.3.2 Classification Method

Since we were unable to identify a reliable source to obtain disclosure classifications, we had chosen to perform this step by ourselves via the following steps.

Step 1. For a CVE ID related to a JP vendor (a "JP CVE ID" for short), we started with its JVN entry, which may contain information about whether the vulnerability was coordinated or not. Specifically, if the sentences "{{*Finder*}} *reported this vulnerability to IPA. JPCERT/CC coordinated with the developer under Information Security Early Warning Partnership.*" or similar are present in a JVN entry, then we have evidence that this disclosure was coordinated by the JPCERT/CC.

For a JP CVE ID whose JVN entry did not contain such sentences or for an US CVE ID, we turned to the external references listed in the corresponding NVD entry. In many cases, it would list the vulnerability report written by a finder, which we then looked through and determined whether the report was publicized before the patch release. In particular, reports in the form of blog posts or mailing list posts would often list the disclosure date, and many would further contain explicit wordings to indicate the nature of the disclosure. For example, Listing 1 shows a partial timeline contained in the report for **CVE-2016-1017**{**4,5,6**}, which clearly shows it was a Full Disclosure.

```
26.09.2016: Email sent to NETGEAR (security () netgear
     com)
            asking for PGP key, no response.
-snip-
20.12.2016: Public disclosure.
```

Listing 1: Partial timeline of **CVE-2016-1017**{**4, 5, 6**} as reported by the finder [34].

Step 2. If we could not classify a disclosure after Step 1, we would further looked into the relevant patch release notes and vendor security advisories to check for finder acknowledgement and report date. For example, for **CVE-2017-15909**, D-Link acknowledged a finder on its patch release notes by including "Reported: September 6, 2017 by: {{Finder}}". This provided us with the confidence that this was a Coordinated Disclosure.

Step 3. Lastly, for a disclosure that had not been classified in above steps, we classified it as Unknown.

4.3.3 Classification Result and Further Analysis

Figure 9 depicts our classification result. Confirming our

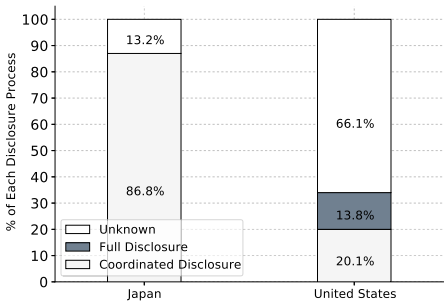


Fig. 9 Category percentages of different vulnerability disclosure processes in our JP and US datasets.

hypothesis that the finders in our JP dataset tended to perform coordinated disclosures, our classification puts over 86% of the vulnerabilities in our JP dataset in the Coordinated Disclosure category and none in the Full Disclosure category. In contrast, the former category accounts for only 20.1% in our US dataset and the latter category is at a sizable 13.8%.

Although we have confirmed our hypothesis, we do not have any explanation on why the finders in our JP and US datasets exhibited different tendencies. However, upon studying the identities of these finders, we did notice that the finders of 30 of the 53 JP CVE IDs declared an affiliation with a local IT security company. We have since then managed to contact one of the finders from this company to inquire about their practice. We have learned that they perform security audits either due to a client contract or as part of their own research. If a vulnerability is discovered under a contract where the client is not the vendor and the client agrees, or if it is discovered during their own research, they would report the vulnerability to IPA and JPCERT/CC, which would result in a coordinated disclosure. Therefore, given the distribution of finders in our JP dataset, the vulnerability disclosure practice of this one company would have yielded a high percentage of coordinated disclosures in our current study.

As for the finders of the 20 JP CVE IDs that were not reported by a vendor itself, we observed that each of them was responsible for at most 2 JP CVE IDs in our dataset. Unfortunately, we have not contacted them to learn about their practice. We believe a fruitful future study on the vulnerability disclosure practice of the two markets might be to conduct a survey among a broadened set of finders in these markets. For example, among other questions, we may ask the finders whether they acted in their personal or professional capacity when they disclosed a vulnerability and whether they were following a company policy if the latter. We anticipate that one potential obstacle to overcome might be on how to derive a contact email address for each finder since this information is generally not published in the data feeds.

5. Significant 1-Day Risks Uncovered

Throughout our data collection effort as described in Sect. 3, we had often leveraged search engines to retrieve documents related to the included vulnerabilities. For example, we would search for a CVE ID or an affected product name and extract information from the search results. This process had led us to uncover three alarming vendor practices that may significantly increase the risk of 1-day exploits for customers due to patch-based exploit generation.

The first practice is incremental patch release, which has already been presented in Sect. 4.2.3. While potentially dangerous, it is debatable whether this risk is avoidable because of the immense difficulty in identifying and patching *all* affected products at once. However, for the two other practices we present below, we believe they can be avoided with moderate investment by the vendors.

5.1 Unsynchronized Patch Release

For many consumer IoT vendors that supply their products globally, a common practice is to establish regional subsidiaries to market and support products in the targeted regions. These subsidiaries usually provide product information, which includes security patches, via a regional website available in a local language. For example, Netgear China provides security patches via its regional support site http://support.netgear.cn/, which is published in Chinese. Since information such as CVE IDs and the model number in product names are not localizable, we encountered these sites regularly in our searches.

Unfortunately, we quickly noticed that different regional websites of the same vendor would often release a patch against the same vulnerability on different days. Since the knowledge of exploit generation from patches is now wide-spread and may even be automated in some cases (e.g., [25]), this behavior of *unsynchronized patch release* by regional subsidiaries likely increases the risk of 1-day exploits for customers in regions that receive the patch late. We dub this risk "*geographical arbitrage*"—the potential to generate 1-day exploits by using a patch that was released earlier in a different region.

To characterize this risk, we extended our dataset to include a few regional subsidiaries of the included vendors. These include the US subsidiary of Buffalo, the Germany (DE) and Australia (AU) subsidiaries of D-Link, and the China (CN) subsidiary of Netgear. We remark that this list is not meant to be exhaustive and merely reflects the subsidiaries that we had encountered often during our searches.

5.1.1 Patch Availability Delay

Table 3 summarizes the patch release behaviors of the included subsidiaries. For each subsidiary, column 3 shows the number of patches we collected from it and columns 4 and 5 show, respectively, its average and median patch availability delay ($t_p - t_d$). We see that the US subsidiaries of Buffalo, D-Link, and Netgear all lagged behind their peers by having the largest average delays. As for the median, D-Link US led its DE and AU peers by 5 and 8 days, whereas

Table 3 Statistics of patch release behaviors of the included subsidiaries and #patches shared with the US subsidiaries.

Vendor	Region	# of Patches	Average ($t_p - t_d$)	Median ($t_p - t_d$)	# of Patches shared w/ US
Buffalo	US	13	-95.9 days	-77 days	-
	JP	105	-276.4 days	-109 days	12
D-Link	US	177	-19 days	4 days	-
	DE	120	-38 days	9 days	103
	AU	65	-40 days	12 days	62
Netgear	US	106	174 days	23 days	-
	CN	53	91 days	15 days	51

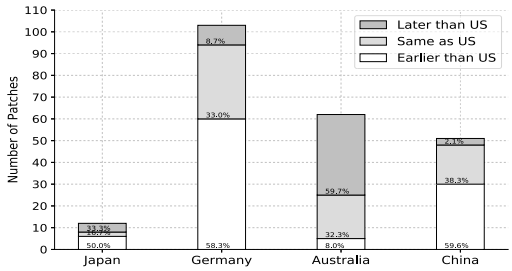


Fig. 10 Inter-subsidiary comparisons of patch release dates: Buffalo US vs. JP, D-Link US vs. DE, D-Link US vs. AU, and Netgear US vs. CN.

Table 4 Statistics of patch release gaps between the US subsidiaries and other included subsidiaries. A positive value means the US subsidiary lagged behind.

Vendor	Region	Average	Median	Maximum
Buffalo	Japan	-0.58 days	0.5 days	1 day
D-Link	Germany	23.7 days	2 days	366 days
	Australia	2.5 days	-1 day	218 days
Netgear	China	31 days	8 days	346 days

Buffalo US and Netgear US were 32 and 8 days slower than their CN and JP peers, all respectively.

5.1.2 Inter-Subsidiary Patch Release Date Comparison

Column 6 of Table 3 lists, for each non-US subsidiary, the number of patches shared with the corresponding US subsidiary. For these shared patches, we computed the relative timing of their release dates and the result is visualized in Fig. 10. Among the respective groups of shared patches, we found that (i) Buffalo JP led Buffalo US in 50% of the patches, (ii) D-Link US was behind D-Link DE in 58.3% of the patches but led D-Link AU in 59.7% of the patches, and (iii) Netgear CN led Netgear US in 59.6% of the patches. Our results shows that the patch releases by these subsidiaries were indeed often unsynchronized.

5.1.3 Inter-Subsidiary Patch Release Gap

Aside from the relative timing of the patch releases by different subsidiaries, we have also computed the *patch release gaps* among them. More precisely, for each shared patch, we computed the difference between the release date by the US subsidiary and that by the corresponding non-US subsidiary (e.g., $t_p^{\mathrm{US}} - t_p^{\mathrm{DE}}$). This statistics is shown in Table 4.

We see that the average patch release gaps were negative for Buffalo and positive for D-Link and Netgear. For the latter two, this means D-Link US and Netgear US on average lagged behind their included peers when releasing patches. For Buffalo, our data shows that Buffalo JP was only 0.58 days late compared to Buffalo US. However, when interpreting this gap, we must note that a patch that was released simultaneously in Japan and the United States would be counted as one day late in Japan due to their time zone difference.

Turning to columns 4 and 5, while the former paints a relatively assuring picture with small medians, the latter is truly alarming—against all three of the included non-US subsidiaries of D-Link and Netgear, we are able to identify a shared patch where the corresponding US subsidiary was well over 200 days behind. To show concrete examples of the patch release gaps and give a sense of the breadth of products affected, Table 5 lists the vulnerabilities and the corresponding products where the patch release gap is *at least* 30 *days*. We believe this table shows that the practice of unsynchronized patch release was indeed very common.

5.1.4 Exploit Release Time

Finally, we have also investigated the exploit availability of the included vulnerabilities using the exploit information in our dataset. In total, we found nine exploits that were released before any of the included subsidiaries released a patch and five exploits that were released in a gap, i.e., when one subsidiary had released a patch but at least one other had not. Although we do not have any evidence to believe that any of these five exploits was a result of geographical arbitrage, we have chosen **CVE-2017-5521** as an example in which a patch was released by Netgear CN before Netgear US and attached its analysis in Appendix C. We believe the simplicity shown in the analysis demonstrates that the risk of geographical arbitrage is very real.

5.2 Implicit End-of-Support

We discovered another alarming vendor practice that may also increase the risk of 1-day exploits for customers when we were collecting the patches released by the regional subsidiaries. Specifically, we sometimes encountered the situation where a subsidiary had *seemingly* stopped to release patches for a product it once supported, only to later discover that another subsidiary had continued to release patches for that product.

Table 6 shows our collection of these products. We see that these products had once been supported by the indicated subsidiary, but the latest version available from that subsidiary (columns 5 and 6) lagged behind the patched version that we were able to discover from a different subsidiary (columns 3 and 4). For example, the patch for **CVE-2017-5521** (WNDR3400v2 firmware version 1.0.0.54) was released by Netgear US on 2017/01/19, but the latest update by Netgear CN was version 1.0.0.48

Table 5 Vulnerabilities and corresponding products where the patch released by the US subsidiary was at least 30 days behind a non-US subsidiary.

Vendor	CVE ID	Public Disclosure Date (t_d)	Affected Product	Patch Release Date (t_p)			$t_p - t_d$		Exploit Release Date (t_e)	$t_p - t_e$	
				US	DE	Δ	US	DE		US	DE
D-Link	CVE-2017-7852	2017-02-22	DCS-2132L RevA	2016-11-02	2016-07-21	104	112	216	2017-02-22	112	216
			DCS-2136L RevA	2016-11-04	2016-07-21	106	110	216		110	216
			DCS-2310L RevA	2016-11-04	2016-07-21	106	110	216		110	216
			DCS-2332L RevA	2016-11-04	2016-07-21	106	110	216		110	216
			DCS-5009L RevA	2016-01-05	2015-11-18	48	414	462		414	462
			DCS-5222L RevB	2016-01-14	2015-12-11	34	405	439		405	439
			DCS-6010L RevA	2016-11-04	2016-07-21	106	110	216		110	216
			DCS-932L RevA	2016-07-19	2015-11-18	244	218	462		218	462
			DCS-933L RevA	2016-01-18	2015-11-18	61	401	462		401	462
	CVE-2016-6563	2016-11-07	DIR-890L RevA	2016-11-09	2016-09-07	63	-2	61	2016-11-21	12	75
			DIR-868L RevA	2016-12-29	2016-09-22	98	-52	46		-38	60
			DIR-869 RevA	2016-12-07	2016-06-22	168	-30	138		-16	152
	CVE-2016-5681	2016-08-11	DIR-868L RevB1	2016-04-29	2015-04-29	366	104	470	-	-	-
			DIR-868L RevC1	2016-07-01	2015-07-01	366	41	407			
	CVE-2016-1558	2016-03-16	DAP-3662 RevA	2016-09-29	2017-01-26	-119	-197	-316	-	-	-
	CVE-2015-1187	2015-03-02	DIR-626L RevA	2015-03-09	2015-04-16	-38	-7	-45	2015-02-26	-11	-49
	CVE-2014-100005	2014-03-07	DIR-600 RevB	2014-03-17	2013-06-25	265	-10	255	-	-	-
	CVE-2013-6027	2013-10-17	DIR-100	2013-12-02	2013-11-01	31	-46	-15	2013-10-14	-49	-18
	CVE-2006-6055	2006-11-21	DWL-G132	2006-11-16	2006-07-07	132	5	137	2006-11-13	-3	129
				US	**AU**	**Δ**	**US**	**AU**		**US**	**AU**
	CVE-2016-6563	2016-11-07	DIR-895L	2016-11-10	2016-09-22	49	-3	46	2016-11-21	11	60
	CVE-2016-5681	2016-08-11	DIR-850L RevB1	2016-05-17	2016-06-21	-35	86	51	-	-	-
			DIR-868L RevB1	2016-04-29	2015-09-24	218	104	322			
				US	**CN**	**Δ**	**US**	**CN**		**US**	**CN**
Netgear	CVE-2017-5521	2017-01-16	R6300v2	2016-09-07	2016-07-04	65	131	196	2017-01-30	145	210
	CVE-2013-3069	2013-07-16	R6200	2013-12-26	2013-01-14	346	-163	183	-	-	-
			WNDR3400v3	2014-05-13	2013-08-07	279	-301	-22			
			WNR1000v3	2014-06-19	2014-02-26	113	-338	-225			
			WNDR3700v2	2014-01-20	2013-12-01	50	-188	-138			
			WNDR37AVv2	2014-01-20	2013-12-01	50	-188	-138			
			WNDR4500v2	2014-02-03	2013-11-21	74	-202	-128			
	CVE-2011-1674	2011-04-09	WNAP210v1	2012-05-06	2011-11-30	158	-393	-235	-	-	-
	CVE-2011-1673	2011-04-09	WNAP210v1	2012-05-06	2011-11-30	158	-393	-235	-	-	-
	CVE-2006-6125	2004-01-30	WG311v1	2004-01-30	2003-12-19	42	0	42	2006-11-22	1027	1069

released on 2013/06/13—a whooping 3+ years before. Although we believe this is a strong indication that Netgear CN had already stopped supporting this product in its region, we were unable to discover any documentation by Netgear CN to this effect. We call this phenomenon *implicit End-of-Support* (implicit EoS), meaning a vendor effectively ends the support of a product without publishing this information.

Unfortunately, our investigation shows that implicit EoS appeared to occur in many markets. Among the seven regional subsidiaries discussed in this paper, we were unable to discover EoS information from Buffalo US, D-Link AU, Netgear US, and Netgear CN. We believe implicit EoS poses a serious security risk to the end users in a manner similar to geographical arbitrage. In Appendix C, we will demonstrate this risk is also very real with a simple vulnerability analysis of **CVE-2016-1555**, which is a case where Netgear CN had seemed to stop releasing patches for a product when Netgear US had continued.

6. Related Work

Many previous studies have leveraged information on vulnerabilities and patches to study various aspects of the vulnerability lifecycle and its management. Here we will discuss a few lines of these related works, grouped by the target of the studies.

The first and most-related line of work aims to study the vendor patch release behavior. In their landmark study, Frei et al. [7] performed a large-scale study using over 80,000 security advisories published between 1995 and 2006 to analyze the temporal relations among vulnerability discoveries, disclosures, and patch releases. A follow-up study was published in 2008 [8], which specifically focused on contrasting between Apple and Microsoft. In 2010, a large-scale study leveraging 420 vulnerabilities was published by Arora et al. [11]. However, the vulnerabilities were sampled from the period of 2000/09 to 2003/08, which was ∼7 years old at the time of its publication. In 2012, Shahzad et al. [13] published another large-scale study using 46,310 vulnerabilities disclosed between 1988 and 2011. Their focus included seven major software vendors and eleven of their products. In comparison, our study focused on consumer IoT vendors and our dataset has the natural benefit of being able to cover more recent vulnerabilities. A notable finding of our work is that we do *not* observe any improvement by our included vendors in terms of patch avail-

Table 6 Vulnerable products where the indicated regional subsidiary had seemingly stopped posting patches without any EoS announcement even though at least one other subsidiary had continued.

CVE ID	Affected Product	Patched Version (Global)		Latest Version at Subsidiary Shown	
		version	release date	version	release date
Buffalo US					
CVE-2016-4816	WHR-300HP	1.98	2016-01-25	1.93	2013-03-07
D-Link Australia					
CVE-2017-7852	DCS-2310L RevA	1.08.03	2016-11-04	1.07.00	2015-10-01
	DCS-2332L RevA	1.08.03	2016-11-04	1.07.00	2015-10-01
	DCS-6010L RevA	1.15.03	2016-11-04	1.14.00	2015-10-01
CVE-2016-6563	DIR-880L	1.08b04	2016-11-10	1.07b08	2016-03-20
	DIR-868L RevB1	2.05b02	2016-12-07	2.03b01	2015-09-24
CVE-2015-2052	DIR-645 RevA	1.05b01	2015-04-24	1.03b11	2012-10-12
CVE-2015-2051	DIR-645 RevA	1.05b01	2015-04-24	1.03b11	2012-10-12
CVE-2014-100005	DIR-600 RevB	2.17b01	2014-03-17	2.15b02	2013-03-11
CVE-2014-9518	DIR-655 RevB	2.12b01	2014-11-01	2.05b06	2012-01-17
CVE-2013-7389	DIR-645 RevA1	1.04b11	2013-12-19	1.03b11	2012-10-12
Netgear China					
CVE-2017-6862	WNR2000v4	1.0.0.66	2017-01-17	1.0.0.44	2015-03-20
CVE-2017-5521	R6700	1.0.1.16	2017-01-16	1.0.0.26	2016-03-31
	WNDR3400v2	1.0.0.54	2017-01-19	1.0.0.48	2013-06-13
CVE-2016-10176	WNR2000v4	1.0.0.66	2017-01-17	1.0.0.44	2015-03-20
CVE-2016-10175	WNR2000v4	1.0.0.66	2017-01-17	1.0.0.44	2015-03-20
CVE-2016-10174	WNR2000v4	1.0.0.66	2017-01-17	1.0.0.44	2015-03-20
CVE-2016-10106	FVS336Gv3	4.3.3-8	2016-05-27	4.3.3-6	2015-10-29
	FVS318Gv2	4.3.3-8	2016-05-27	4.3.3-6	2015-10-29
	SRX5308	4.3.3-8	2016-05-27	4.3.3-6	2015-10-29
CVE-2016-6277	R6700	1.0.1.16	2017-01-16	1.0.0.26	2016-03-31
CVE-2016-1556	WN604	3.3.3	2016-03-03	3.0.2	2012-12-19
CVE-2016-1555	WN604	3.3.3	2016-03-03	3.0.2	2012-12-19
CVE-2013-4775	GS724Tv3	5.4.2.27	2017-01-09	5.4.2.19	2015-06-25
	GS716Tv2	5.4.2.27	2017-01-09	5.4.2.19	2015-06-25
	GS108Tv2	5.4.2.27	2016-12-26	5.4.2.19	2015-07-01
	GS110TP	5.4.2.27	2017-01-09	5.4.2.19	2015-07-01
	GS510TP	5.4.2.27	2017-01-09	5.4.2.19	2015-07-01
	GS752TPS	5.3.0.29	2016-12-26	5.3.0.26	2015-01-20
	GS728TPS	5.3.0.29	2016-12-26	5.3.0.26	2015-01-20
	GS728TS	5.3.0.29	2016-12-26	5.3.0.26	2015-01-20
	GS752TS	5.3.0.29	2016-12-26	5.3.0.26	2015-01-20

ability delay over the covered period, whereas [13, Fig. 9] showed a marked improvement by the software vendors in that study starting around 2006.

Another line of related work focused on characterizing patches and vulnerabilities. Leveraging the WINE dataset and public data, Bilge and Tudor [16] identified 18 zero-day vulnerabilities, of which 11 were not previously known to have been used in zero-day attacks. Similarly, Nappa et al. [26] characterized patch deployments of 10 popular client software packages and their vulnerabilities due to shared code. In contrast to these two studies, several previous works relied on only public data. In addition to the aforementioned study by Shahzad et al., we are also aware of [14], [15], [18], which all focused on individual open-source projects. Most recently, Li and Paxson [19] published a large-scale study on 682 open-source projects, characterizing over 4000 bug fixes for over 3000 vulnerabilities. Sharing a similarity with our study, their study also revealed a significant security-sensitive information leak. In particular, the authors observed that the patches to many vulnerabilities in open-source projects were publicly visible before disclosure, thus posing a significant risk of 1-day exploits.

Finally, we note that there has been a long line of work on notification, which is related to our study because a natural next step to our study would be to investigate the mechanisms and the effectiveness of notifying consumers in regards to vulnerable devices. For this direction, we refer the interested readers to five most recent works that we are aware of and the references therein: [35]–[39].

7. Discussions

Data Quality. First, we sincerely thank the JVN for providing curated public disclosure dates. Although some previous works used the entry creation dates in the NVD as their best-effort estimation of the public disclosure dates (e.g., [13] and [26]), we are aware that these two dates may differ greatly. For example, **CVE-2016-1558** is a buffer overflow vulnerability found in D-Link wireless routers. A vendor security advisory of it was published on 2016/03/16 and this date was correctly noted in the JVN. However, the NVD entry for this CVE ID was created on 2017/04/21, which was over a year later the former. Thus, we believe future studies should consider using the JVN feed as a more accurate source of public disclosure dates.

Aside from vulnerability databases such as the NVD and JVN, patch release notes are arguably the most important information source in vulnerability lifecycle management. Unfortunately, we encountered much irregularity in this source during our study. While some release notes do not contain a date (Sect. 3.4), some others do not mention the CVE ID of the vulnerability being fixed even when the patch is for a published CVE ID. For example, we identified the patched version for **CVE-2008-6122** as 1.2.6 because the release note in version 1.2.6 of the firmware contains similar keywords as the CVE entry. The high similarity between the two is shown in Table A·2 in Appendix B. We believe future studies may consider automating this matching process by leveraging natural language processing techniques.

Data Size. During our study, we had on multiple occasions received feedback that the number of included CVE entries "seemed low". These were often accompanied by expressions of surprise and also suggestions to include more vendors. However, we believe we have already hit a diminish of return with our inclusion criteria and adding less-prominent vendors cannot substantially increase the number of included CVE entries, especially in non-US markets. Instead, we believe this perception may indicate a limitation of our approach—as we mentioned in Sect. 3.5, at present a vendor may silently patch a vulnerability that was discovered internally. This hinders any study in vulnerability lifecycle management that uses information about patch releases to measure vendor behaviors. We believe policy makers may consider requiring vendors to publicly and immediately disclose the dates, affected products, and affected versions of all discovered vulnerabilities in a way akin to data breach disclosures. Aside from allowing consumers to be informed about vulnerabilities that may affect them, this would also create incentives for vendors to create products that are more secure in the first place.

Vendor Behaviors. Our result shows that vendors who operate regional subsidiaries should consider to synchronize patch releases among their subsidiaries and to publish End-of-Support information in every region. We believe the former is particularly important because consumers may not be

able to simply install a firmware from another region onto their devices due to region-specific configurations such as the choice of wireless frequencies. Although these practices likely require additional investments in software engineering and business operations, we believe they are much-needed improvements—indeed, many mainstream software and smartphone vendors have long adopted these best practices and consumers would (rightly) expect them from every vendor.

Finally, we recommend vendors to consider maintaining permanent machine-readable security feeds, such as release notes with JSON/XML metadata and advisories in RSS. Together with adequate disclosures suggested above, we believe these feeds would create values for participating vendors by, e.g., enabling researchers to create better tools to help the customers of participating vendors to scan and defend their networks after a vulnerability has been discovered.

8. Conclusion

In this paper, we have studied the vulnerability disclosures and patch release behaviors of three prominent consumer IoT device vendors in Japan and three in the United States. Our dataset spans the 12 years from 2006 to 2017, covering 283 CVE entries and 450 products. To ensure the accuracy and completeness of our data, we cross-referenced a large set of data sources, including the NVD, JVN, security advisories mentioning an included device, and release notes in the patches to an included device after its first affected version.

Viewing our dataset as a whole, we are encouraged to see that five out of the six included vendors had been releasing patches in a timely manner (Fig. 4). However, our dataset also reveals several less-desirable behaviors of the included vendors, such as (i) their patch release delay did not show significant improvement over the 12 covered years, and (ii) they did not appear to prioritize releasing patches to vulnerabilities that had been classified as more severe.

One interesting investigation enabled by our bi-country inclusion criteria is to look for differences between markets. In particular, we noted major differences between our data on the two markets in our stratified analyses in Sect. 4.2. We explained these differences in Sect. 4.3, where we showed that 86.8% of the vulnerability disclosures involving the included JP vendors were coordinated, which is in stark contrast with the corresponding figure of 20.1% for the included US vendors. We have further identified that the finders of the majority of our JP dataset were affiliated with a Japanese IT security company and we have learned through one of the finders in this company that they would coordinate with IPA and JP/CERT when they publicly disclose a vulnerability.

Finally, our investigation has also uncovered three alarming vendor practices that may significantly increase the risk of 1-day exploits for customers. First, all six included vendors practiced *incremental patch release*. This practice means attackers may use a patch released for one device to discover and exploit the same vulnerability in other devices for which the vendor has yet to release patches. Second, several geographic subsidiaries of two of the included US vendors practiced *unsynchronized patch release*. This practice gives rise to ample opportunities of geographical arbitrage of 1-day exploits. Third, four of the included subsidiaries appeared to practice *implicit End-of-Support*. This practice also poses a significant security risk in a manner similar to geographical arbitrage.

Acknowledgements

We thank Allen Householder for insightful discussions and his suggestion of the term “geographical arbitrage”. We also thank the editor and the anonymous reviewers for their valuable feedback to our initial submission.

References

[1] 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,” ACM Asia Conference on Computer and Communications Security (AsiaCCS ’19), July 9–12, 2019, Auckland, New Zealand, ACM Press, pp.485–492, 2019.
[2] National Telecommunications and Information Administration, “Multistakeholder process—Internet of Things (IoT) security upgradability and patching,” https://www.ntia.doc.gov/other-publication/2016/multistakeholder-process-iot-security
[3] Ministry of Internal Affairs and Communications, the Government of Japan, “Conducting survey on vulnerable IoT devices,” JP: http://www.soumu.go.jp/menu_news/s-news/02ryutsu03_04000088.html, EN: http://www.soumu.go.jp/ main_sosiki/joho_tsusin/eng/ Releases/Telecommunications/ 170905_1.html
[4] B. Liu, L. Shi, Z. Cai, and M. Li, “Software vulnerability discovery techniques: A survey,” Proceedings of the 4th International Conference on Multimedia Information Networking and Security, IEEE, pp.152–156, 2012. [Online]. Available: http://ieeexplore.ieee.org/document/6405650/
[5] M. Felderer, M. Büchler, M. Johns, A.D. Brucker, R. Breu, and A. Pretschner, “Security testing,” Advances in Computers, vol.101, pp.1–51, 2016. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0065245815000649
[6] J. Li, B. Zhao, and C. Zhang, “Fuzzing: A survey,” Cybersecurity, vol.1, no.6, pp.1–13, 2018. [Online]. Available: https://cybersecurity.springeropen.com/articles/10.1186/s42400-018-0002-y
[7] S. Frei, M. May, U. Fiedler, and B. Plattner, “Large-scale vulnerability analysis,” Proceedings of the 2006 SIGCOMM Workshop on Large-Scale Attack Defense, ACM Press, pp.131–138, 2006.
[8] S. Frei, B. Tellenbach, and B. Plattner, “0-day patch exposing vendors (in)security performance,” Black Hat Europe 08, Black Hat, pp.1–15, 2008. [Online]. Available: https://www.blackhat.com/presentations/bh-europe-08/Frei/Whitepaper/bh-eu-08-frei-WP.pdf
[9] S. Frei, “Security econometrics—The dynamics of (in)security,” Ph.D. dissertation, ETH Zurich, 2009. [Online]. Available: https://www.research-collection.ethz.ch/handle/20.500.11850/151394
[10] G. Schryen, “A comprehensive and comparative analysis of the patching behavior of open source and closed source software vendors,” Proceedings of the Fifth International Conference on IT Security Incident Management and IT Forensics, IEEE, pp.153–168, 2009. [Online]. Available: http://ieeexplore.ieee.org/document/5277883/
[11] A. Arora, R. Krishnan, R. Telang, and Y. Yang, “An empirical analysis of software vendors’ patch release behavior: Impact of vulner-

ability disclosure," Info. Sys. Research, vol.21, no.1, pp.115–132, March 2010.
[12] A. Arora, C. Forman, A. Nandkumar, and R. Telang, "Competition and patching of security vulnerabilities: An empirical analysis," Information Economics and Policy, vol.22, no.2, pp.164–177, May 2010. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167624509000651
[13] M. Shahzad, M.Z. Shafiq, and A.X. Liu, "A large scale exploratory analysis of software vulnerability life cycles," Proceedings of the 34th International Conference on Software Engineering, IEEE Press, pp.771–781, 2012.
[14] A. Ozment and S.E. Schechter, "Milk or wine: Does software security improve with age?," Proceedings of the 15th USENIX Security Symposium, USENIX Association, pp.93–104, 2006. [Online]. Available: https://www.usenix.org/legacy/event/sec06/tech/ozment.html
[15] S. Zaman, B. Adams, and A.E. Hassan, "Security versus performance bugs: A case study on Firefox," Proceeding of the 8th Working Conference on Mining Software Repositories, ACM Press, pp.93–102, 2011. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1985441.1985457
[16] L. Bilge and T. Dumitras, "Before we knew it—An empirical study of zero-day attacks in the real world," Proceedings of the 2012 ACM Conference on Computer and Communications Security, ACM, pp.833–844, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2382196.2382284
[17] Z. Durumeric, J. Kasten, D. Adrian, J.A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, and V. Paxson, "The matter of Heartbleed," Proceedings of the 2014 Conference on Internet Measurement Conference, ACM, pp.475–488, 2014. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2663716.2663755
[18] Z. Huang, M. DAngelo, D. Miyani, and D. Lie, "Talos: Neutralizing vulnerabilities with security workarounds for rapid response," Proceedings of the 2016 IEEE Symposium on Security and Privacy, IEEE, pp.618–635, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7546526/
[19] F. Li and V. Paxson, "A large-scale empirical study of security patches," Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, ACM Press, pp.2201–2215, 2017.
[20] MITRE Corporation, "CVE: Common Vulnerabilities and Exposures," https://cve.mitre.org/
[21] National Institute of Standards and Technology, "NVD: National Vulnerability Database," https://nvd.nist.gov/
[22] JPCERT Coordination Center and IPA Information-technology Promotion Agency, "JVN: JVN iPedia," https://jvndb.jvn.jp/
[23] A.D. Householder, G. Wassermann, A. Manion, and C. King, "The CERT guide to coordinated vulnerability disclosure," https://resources.sei.cmu.edu/asset_files/SpecialReport/2017_003_001_503340.pdf
[24] "CERT Coordination Center," https://sei.cmu.edu/about/divisions/cert/index.cfm
[25] D. Brumley, P. Poosankam, D. Song, and J. Zheng, "Automatic patch-based exploit generation is possible: Techniques and implications," Proceedings of the 2008 IEEE Symposium on Security and Privacy, IEEE Computer Society, pp.143–157, 2008.
[26] A. Nappa, R. Johnson, L. Bilge, J. Caballero, and T. Dumitras, "The attack of the clones: A study of the impact of shared code on vulnerability patching," Proceedings of the 2015 IEEE Symposium on Security and Privacy, IEEE Computer Society, pp.692–708, 2015.
[27] Internet Archive, "Internet archive," https://archive.org/
[28] JPCERT Coordination Center and Information-technology Promotion Agency, "How to use JVN iPedia," https://jvndb.jvn.jp/en/nav/jvndbhelp.html
[29] Offensive Security, "Exploit-db," https://www.exploit-db.com/
[30] Rapid 7, "Metasploit framework," https://www.metasploit.com/
[31] A. Fioraldi, "cve_searchsploit," https://github.com/andreafioraldi/cve_searchsploit
[32] Rapid 7, "Vulnerability & exploit database," https://www.rapid7.com/db
[33] devttys0, "binwalk," https://github.com/ReFirmLabs/binwalk
[34] P. Ribeiro, "Stack buffer overflow vulnerability in netgear wnr2000 router," https://raw.githubusercontent.com/pedrib/PoC/master/advisories/netgear-wnr2000.txt
[35] F. Li, G. Ho, E. Kuan, Y. Niu, L. Ballard, K. Thomas, E. Bursztein, and V. Paxson, "Remedying web hijacking," Proceedings of the 25th International Conference on World Wide Web, ACM Press, pp.1009–1019, 2016. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2872427.2883039
[36] F. Li, M. Bailey, Z. Durumeric, J. Czyz, M. Karami, D. Mccoy, S. Savage, M. Bailey, D. Mccoy, S. Savage, and V. Paxson, "You've got vulnerability: Exploring effective vulnerability notifications," Proceedings of the 25th USENIX Security Symposium, USENIX Association, pp.1033–1050, 2016. [Online]. Available: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/li
[37] B. Stock, G. Pellegrino, C. Rossow, M. Johns, and M. Backes, "Hey, you have a problem: On the feasibility of large-scale web vulnerability notification," Proceedings of the 25th USENIX Security Symposium, USENIX Association, pp.1015–1032, 2016. [Online]. Available: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/stock
[38] J. Zhang, H. Duan, W. Liu, and X. Yao, "How to notify a vulnerability to the right person? Case study: In an ISP scope," Proceedings of the 2017 IEEE Global Communications Conference, IEEE, pp.1–7, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/8253993/
[39] B. Stock, G. Pellegrino, F. Li, M. Backes, and C. Rossow, "Didn't you hear me?—Towards more successful web vulnerability notifications," Proceedings of the 2018 Network and Distributed System Security Symposium, Internet Society, 2018. [Online]. Available: https://www.ndss-symposium.org/wp-content/uploads/sites/25/2018/02/ndss2018_01B-1_Stock_paper.pdf
[40] Trustwave SpiderLabs, "Multiple vulnerabilities in NETGEAR routers," https://www.trustwave.com/Resources/Security-Advisories/Advisories/TWSL2017-003/?fid=8911

Appendix A: Vendor Name Variants

Table A·1 shows the orthographical variants of vendor names we have identified for each included vendor. It was referred to in Sect. 3.3.

Table A·1 Vendor name variants that were registered in the NVD.

Vendor	Name Variants
Buffalo	buffalo, buffalotech, buffalo_inc
IO-DATA	iodata, i-o_data, i-o_data_device
NEC (Aterm)	nec, aterm
D-Link	dlink, d-link
Linksys	linksys, cisco
Netgear	netgear

Appendix B: Example of Manual Matching between CVE Entries and Release Notes

Table A·2 shows an example where we were able to identify that **CVE-2008-6122** was fixed in firmware release 1.2.6 of

Table A·2 Two descriptions of the vulnerability in **CVE-2008-6122**: CVE entry vs. manually-matched release note.

Source	Description
NVD CVE-2008-6122	The web management interface in Netgear WGR614v9 allows remote attackers to cause a denial of service (crash) via a request that contains a question mark ("?").
Release Note WGR614v9 1.2.6	Fixed: Entering the question mark '?' to the web interface (like http://192.168.1.1/?) will crash the router; the router cannot work normally until a reboot.

the Netgear WGR614v9 wireless router. Even though the included release note did not mention the CVE ID, we were able to match it using the vulnerability description. The table was referred to in Sect. 3.4.

Appendix C: Vulnerability Analyses

In this appendix, we provide the vulnerability analyses of two CVE entries to demonstrate the relatively-low complexity to reify the 1-day risks presented in Sect. 5.1 and Sect. 5.2.

1. Unsynchronized Patch Release

CVE-2017-5521 is a vulnerability which affects many Netgear routers, potentially allowing an attacker to obtain the administrator password for the Web console. This is one of the cases where a patched firmware was released in the US later than in China. When an unauthorized user accesses the Web console, it leaks a token which can be reused as a parameter for another query to display the admin password [40]. In this vulnerability analysis, we examine the product R6300v2, which is one of the affected routers.

Comparing the unpatched (V1.0.4.2_10.0.74) and the patched (V1.0.4.6_10.0.76) versions of its firmware released for the US, we have identified some changes which we believe are relevant to the vulnerability. In what follows, we will analyze the `httpd` binary (ARM ELF) in the firmware, focusing on the function that generates the content of the HTTP body using the status code and other variables.

In the unpatched binary, the 401 content is generated using the format string shown in Listing 2 in order to redirect the user to `unauth.cgi`. As shown in the decompiled code snipped in Listing 4, the parameter `id` is appended to the redirect URL. This is the token that can be reused to leak the password.

```
HTTP/1.0 401 Unauthorized
WWW-Authenticate: Basic realm="%s"
Content-type: text/html
<html><head><meta http-equiv='Content-Type' content='text/html; charset
      =utf-8'>
<title>%s</title></head>
<body onload="document.aForm.submit()"><h1>%s</h1>
<p>%s</p><form method="post" action="unauth.cgi%s" name="aForm"></form
      ></body></html>
```

Listing 2: Format string leaking `id` (`FMTSTR_A`).

In the patched version, the control flow has been modified as shown in Listing 5. Instead of using the format string in Listing 2, it uses the one in Listing 3, which has been added to the patched binary and does not leak the parameter `id`. Although it still uses Listing 2 when the NVRAM config parameter `enable_password_recovery` is set, this should be safe because in that case the attacker will be asked to answer a security question first instead of seeing the password [40].

```
HTTP/1.0 401 Unauthorized
WWW-Authenticate: Basic realm="%s"
Content-type: text/html
<html><head><meta http-equiv='Content-Type' content='text/html; charset
      =utf-8'>
<title>%s</title></head>
<body><h1>%s</h1>
<p>%s</p></body></html>
```

Listing 3: Format string added in patched (`FMTSTR_B`).

```
if ( status_code == 401 )
    <snip>
    sprintf(v16, "?id=%lu", dword_TOKEN);
    v11 = (const char *)FMTSTR_A;
    <snip>
    return (void *)sprintf(buf, v11, v5, v12, v13, v14, v16);
```

Listing 4: Code in unpatched firmware.

```
if ( status_code == 401 ){
    if ( acosNvramConfig_match((int)"enable_password_recovery", (int)
          "1") ) {
        sprintf(v25, "?id=%lu", dword_TOKEN);
        v11 = (const char *)FMTSTR_A;
        <snip>
        result = (void *)sprintf(buf, v11, v5, v12, v13, v14, v25);
    } else {
        v15 = (const char *)FMTSTR_B;
        <snip>
        result = (void *)sprintf(buf, v15, v5, v16, v17, v18);
    }}
```

Listing 5: Code in patched firmware.

2. Implicit End-of-Support

CVE-2016-1555 is a command injection vulnerability that was found in Netgear wireless access point models WN604, WN802Tv2, WNAP210, WNAP320, WNDAP350, and WNDAP360. This vulnerability resides in the device settings page, which allows the user to input a MAC address into a text field. This user-controlled string is then passed as the parameter `macAddress` without sanitization and used as one of the arguments to the `exec()` call in PHP. The vulnerable code in version 3.3.2 is shown in Listing 6, and the patched version is shown in Listing 7. We see that the vendor has fixed this vulnerability by applying the sanitization function `escapeshellcmd()`.

According to Netgear, this vulnerability has been fixed in version 3.3.3. However, the latest version listed by Netgear CN was 3.0.2 when we collected our data, which is far behind from the patched version that was already released by Netgear US at the time. To confirm the vulnerability, we have inspected version 3.0.2 downloaded from Netgear CN and the source code is vulnerable because it is the same as in version 3.3.2, which we show in Listing 8.

```
if (!empty($_REQUEST['macAddress']) &&
  array_search($_REQUEST['reginfo'],
  Array('WW'=>'0','NA'=>'1'))!==false &&
  ereg("[0-9a-fA-F]{12,12}",$_REQUEST['macAddress'],$regs)!==false) {
    exec("wr_mfg_data -m ".$_REQUEST['macAddress']." -c ".$_REQUEST['
        reginfo'],$dummy,$res);
```

Listing 6: `boardDataWW.php` in WN604 firmware version 3.3.2 released by Netgear US (before patch).

```
$macAddress = escapeshellcmd($_POST['macAddress']);
$reginfo = escapeshellcmd($_POST['reginfo']);
if (!empty($macAddress) && !empty($reginfo)) {
  if(validateCommandArg($macAddress,$reginfo))
      exec("wr_mfg_data -m ".$macAddress." -c ".$reginfo,$dummy,$res)
          ;
```

Listing 7: `boardDataWW.php` in WN604 firmware version 3.3.3 released by Netgear US (after patch).

```
if (!empty($_REQUEST['macAddress']) &&
  array_search($_REQUEST['reginfo'],
  Array('WW'=>'0','NA'=>'1'))!==false &&
  ereg("[0-9a-fA-F]{12,12}",$_REQUEST['macAddress'],$regs)!==false) {
    exec("wr_mfg_data -m ".$_RtEQUEST['macAddress']." -c ".$_REQUEST['
        reginfo'],$dummy,$res);
```

Listing 8: `boardDataWW.php` in WN604 firmware version 3.0.2 released by Netgear CN (latest version).

Asuka Nakajima received her B.A. degree in Environmental Information from Keio University, Japan in 2013. Since joining Nippon Telegraph and Telephone Corporation (NTT) in 2013, she has been engaged in research and development on software security, especially vulnerability discovery, reverse engineering, and IoT Security. She is currently affiliated with the Cyber Security Project of NTT Secure Platform Laboratories.

Takuya Watanabe received his M.E. degree in Computer Science and Engineering from Waseda University, Japan in 2016. Since joining Nippon Telegraph and Telephone Corporation (NTT) in 2016, he has been engaged in research of consumer security and privacy. He is currently affiliated with the Cyber Security Project of NTT Secure Platform Laboratories.

Eitaro Shioji received his B.E. degree in Computer Science and M.E. degree in Communications and Integrated Systems from Tokyo Institute of Technology in 2008 and 2010, respectively. Since joining Nippon Telegraph and Telephone Corporation (NTT) in 2010, he has been engaged in research and development on computer security. His research interests include software security, program analysis, and reverse engineering.

Mitsuaki Akiyama received his M.E. and Ph.D. degrees in Information Science from Nara Institute of Science and Technology, Japan in 2007 and 2013, respectively. Since joining Nippon Telegraph and Telephone Corporation (NTT) in 2007, he has been engaged in research and development on cybersecurity. He is currently a Senior Distinguished Researcher and is affiliated with the Cyber Security Project of NTT Secure Platform Laboratories. His research interests include cybersecurity measurement, offensive security, and usable security and privacy.

Maverick Woo received his Ph.D. degree in Computer Science from Carnegie Mellon University, USA in 2009. He joined the CyLab Security and Privacy Institute at Carnegie Mellon University in 2011 and he is currently a Systems Scientist at CyLab. His research interests include software security and program analysis, with a focus on algorithm design and budget optimization.