Selective Forwarding Attack on IoT Home Security Kits

Ali Hariri, Nicolas Giannelos, and Budi Arief[0000−0002−1830−1587]

School of Computing, University of Kent, Canterbury, UK
{hariri.ali.93, nikognl}@gmail.com, b.arief@kent.ac.uk

Abstract. Efforts have been made to improve the security of the Internet of Things (IoT) devices, but there remain some vulnerabilities and misimplementations. This paper describes a new threat to home security devices in which an attacker can disable all functionality of a device, but to the device's owner, everything still appears to be operational. We targeted home security devices because their security is critical as people may rely on them to protect their homes. In particular, we exploited a feature called "heartbeat", which is exchanged between the devices and the cloud in order to check that the devices are still connected. Even though network traffic was encrypted, we successfully identified the heartbeats due to their fixed size and periodic nature. Thereafter, we established a man-in-the-middle attack between the device and the cloud and selectively forwarded heartbeats while filtering out other traffic. As a result, the device appears to be still connected (because the heartbeat traffic is being allowed through), while in reality the device's functionality is disabled (because non-heartbeat traffic is being filtered out). We applied this exploit on a set of six devices, and five were found to be vulnerable. Consequently, an intruder can use this exploit to disable a home security device and break into a house without the awareness of the owner. We carried out a responsible disclosure exercise with the manufacturers of the affected devices, but the response has been limited. This shows that IoT security is still not taken completely seriously and many threats are still undiscovered. Finally, we provide some recommendations on how to detect and prevent the threats posed by insecure IoT devices, which ironically include IoT home security kits.

Keywords: IoT · Security · Attack · Off-the-shelf Devices · Heartbeats · Selective Forwarding · SSL/TLS · WPA2.

1 Introduction

The Internet has considerably changed in the last decade. It has become more than just a platform for email exchanges, web browsing, instant messaging or media streaming. The connected devices are no longer just servers, computers and smartphones, but instead the Internet has become an Internet of Things (IoT), of connected wearables, home appliances, biomedical devices, cars, cities, and many more. IoT is gaining more and more popularity and experts predict that it will become an Internet of Everything in the near future [1]. It is estimated that the number of connected devices will reach up to 20 billion by 2020 [2].

Despite its convenience, IoT and its applications introduce major privacy threats and critical security risks. For instance, IoT devices may be compromised to access personal information, or to gain control over industries, cities and public organisations or disrupt their services. This has been shown by several incidents like the Mirai botnet [3,4]. Mirai is a computer worm that compromised hundreds of thousands of IoT devices, which were then used to mount a Distributed Denial of Service (DDoS) attack to disrupt well-known services like Netflix and Twitter [5]. IoT also imposes personal privacy threats through smart home devices like cameras, personal assistants and home automation kits. For example, Trendnet home security and monitoring cameras were found to be vulnerable, allowing an attacker to access live video feeds of the camera without any authentication [6]. Likewise, customers of Swann home cameras reported that they were able to access recordings from cameras of other customers [7]. A very recent report of security vulnerabilities in three specialist car alarm systems further illustrates the danger of connecting your device to the Internet without proper security testing [8]. These vulnerabilities allowed attackers to steal or hijack affected vehicles through the compromised alarm system. What is ironic here is that whoever bought these vulnerable car alarm systems did so out of a desire to improve the security of their vehicle. But inadvertently, they introduced security vulnerabilities that would allow attackers to take control of their vehicle. This irony resonates with the message we aim to convey in our paper.

Evidently, the security of IoT devices is a major issue that needs to be continuously evaluated and addressed due to its impact on the physical world. This motivated us to explore new and common vulnerabilities in a selected set of consumer IoT products. Particularly, we targeted home security devices because their security is critical as people may rely on them to protect their homes.

Contribution. The main contribution of this paper is the *exploitation of a vulnerability in the heartbeat exchange of IoT devices*. By exploiting this vulnerability, an attacker can disable IoT home security devices without the awareness of their owners. Particularly, the device will appear to be online and working normally, but in fact it will be completely disabled. This was due to wrong implementations of heartbeat messages exchanged between devices and their cloud infrastructure. Our second contribution relates to a *potential misimplementation of the WPA2 four-way handshake protocol in some IoT devices*. This misimplementation allows an attacker to carry out an evil twin access point attack, which would force the device to connect to the attacker's LAN. This would allow the attacker to exploit further vulnerabilities, or eavesdrop on the communications between the device and the cloud.

The rest of this paper is organised as follows. Section 2 presents related work. Section 3 outlines our methodology, while Section 4 presents our findings and results. Section 5 describes the threat model to illustrate the feasibility of our attack. Section 6 dissects the risks and consequences of the discovered vulnerability. We present some insights from the responsible disclosure exercise we carried out with the affected manufacturers of the devices. We also propose some recommendations to how to fix the vulnerabilities uncovered by our research. Finally, Section 7 concludes the paper and provides ideas for future work.

2 Related Work

Visan et al. [9] assessed the security of Samsung Smart Things hub. They first attempted to extract credentials from the hub using various traffic sniffing methods. They also demonstrated that the hub is robust and secure. However, they discovered that a DoS attack against the hub is possible, if the attacker has access to the LAN. The attack would give an intruder an 8-minute window to break into a house before notifying the owner. Visan et al. argue that home security kits are not completely reliable.

Another good example is the work by Fernandes et al. [10], which demonstrated again that despite the effort put into security, there may remain some security issues due to the complex nature of the products. More precisely, they discovered vulnerabilities inside the architecture, in the capability model and the event subsystem of Samsung Smart Things, due to the numerous and complex functionalities exposed to the user. By exploiting them, they managed to insert backdoor pin-codes into a connected door lock, eavesdrop door lock pin-codes, cause a fake fire alarm and disable the vacation mode of the device. More importantly, those vulnerabilities were significant as they targeted the architecture of the application layer at its core, thus making them difficult to patch.

Apthrope et al. [11] proved that users privacy can be breached without compromising devices or network communications. They showed that any party having network access – e.g. Internet Service Providers (ISPs) – can infer sensitive information just by analysing network traffic. They particularly analysed DNS requests, IP and TCP headers and packet rates to identify device types and user interaction. For instance, they were able to determine if a user is sleeping by analysing a sleep sensors traffic, or if the user is moving inside a house by analysing a motion sensors traffic. This work highlights that privacy is a critical challenge in IoT security, and it cannot be achieved with cryptography only.

Jerry Gamblin discovered that Google home assistant can be controlled by any device that has network access to the LAN. The device can send commands to the assistant without any authentication and it can cause to reboot and even to disconnect from Wi-Fi. This is due to an undocumented API that can be exploited by sending rogue commands [12].

Very recently, OConnor et al. [13] uncovered a design flaw in the messaging protocols of 22 IoT devices. The design flaw they uncovered is very similar to the weakness we discuss in this paper. We independently carried out our research and developed a proof-of-concept automated tool to exploit this flaw.

3 Methodology

We chose smart home IoT devices because they provide a wider attack surface due to their numerous interconnected components such as cameras, alarms, motion detectors and many other sensors. We targeted a set of home security devices including Swann Smart Home Security Kit (SWO-HUB01K), D-Link Home Security Kit (DCH-G020), D-Link camera (DCS-935L), Panasonic Home Monitoring and Control Kit (KX-HN6012EW), Telldus Smart Home Start-up Kit and Samsung SmartThings (F-H-ETH-001). We performed DoS attacks on

the selected devices, mainly Wi-Fi deauthentication and blackhole attacks. We used `arpspoof` tool [14] for a blackhole attack and `aireplay-ng` tool [15] for Wi-Fi deauthentication attack. `arpspoof` tool was used to achieve a MITM position and IP forwarding was disabled to complete the blackhole attack. Moreover, we analysed network traffic thoroughly to find patterns that were then used in our selective forwarding attack. Traffic analysis was mainly conducted using `Wireshark` [16], in which several `Wireshark` filters were used to analyse and identify patterns in network traffic of each device. In addition, our selective forwarding attack requires achieving a MITM position on the Transport layer. Thus, it was necessary to run a TCP proxy and force the devices to connect to the proxy instead of connecting to their legitimate servers. DNS poisoning was the best option to force the devices to connect to the proxy. For that, we used `Wireshark` to view DNS requests and responses, and identify the domain names and the IP addresses of the servers. Then, we used `Bind9` and configured it to resolve the identified DNS requests to the IP address of the machine that is running the proxy. As a result, all devices connected to the proxy instead of connecting to their servers. The proxy was developed as a Python script, and it can be found with the Bind9 configurations on GitHub[1].

4 Results

We conducted blackhole and Wi-Fi de-authentication attacks on the selected devices to determine the required time to alert or display that the device went offline. The results showed that the devices took between 5 seconds and 2 minutes to alert the user depending on the vendor. Thereupon, we deduced that the devices must be exchanging periodic messages with the cloud to prove that they are still online. Once the cloud stops receiving those messages, it reports to the user application that the device was disconnected. Those messages are known as heartbeats and they are defined as "a periodic signal generated by hardware or software to indicate that it is still running" [17].

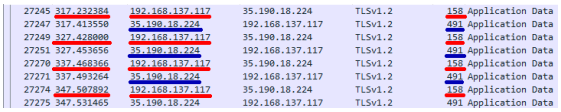
We realised that heartbeats will always have a fixed size during one session because the payload is always the same and it is always encrypted with the same algorithm over this session. Subsequently, we postulated that heartbeat messages can be identified in network traffic even if they were encrypted due to their periodic nature and constant payload size. Therefore, we presumed that we can exploit this pattern by selectively forwarding heartbeats and TLS handshakes, and blocking any other traffic. This will deceive the server into believing that the device is still online and sending heartbeats, when in fact its traffic is blocked. We designated this attack as the "heartbeat attack".

4.1 Heartbeats in Swann

We noticed that every 10 seconds, the hub sends 92 bytes of data and the server responds with 425 bytes of payload as shown in Figure 1. Subsequently, we wrote a Python script that opens two TCP streams with the hub and the server, and selectively forwards data between them. The proxy initially forwards all data for few seconds to ensure that the TLS handshake was completed, then it only

[1] https://github.com/HaririAli/IoTHeartbeatProxy.git

forwards 92-byte and 425-byte long data. Thus, the proxy will only forward heartbeats and block any other traffic.



27245	317.232384	192.168.137.117	35.190.18.224	TLSv1.2	158	Application Data
27247	317.413550	35.190.18.224	192.168.137.117	TLSv1.2	491	Application Data
27249	327.428000	192.168.137.117	35.190.18.224	TLSv1.2	158	Application Data
27251	327.453656	35.190.18.224	192.168.137.117	TLSv1.2	491	Application Data
27270	337.468366	192.168.137.117	35.190.18.224	TLSv1.2	158	Application Data
27271	337.493264	35.190.18.224	192.168.137.117	TLSv1.2	491	Application Data
27274	347.507892	192.168.137.117	35.190.18.224	TLSv1.2	158	Application Data
27275	347.531465	35.190.18.224	192.168.137.117	TLSv1.2	491	Application Data

Fig. 1. Wireshark capture of heartbeats in Swann's traffic

4.2 Heartbeats in D-Link Hub

Similarly, we analysed network traffic from D-Link hub looking for a periodic pattern. However, we did not find any periodic data packets exchanged between the server and the hub. We rather found that the server sends TCP keepalive packets every 5 seconds and the hub responds with a TCP keepalive ACK. Thus, D-Link relied on TCP keepalive packets to ensure that the hub is still online and connected. This, in fact, makes the exploit much simpler as the proxy does not have to forward any data but rather just acknowledge TCP keepalive packets once the connection is established. Thereupon, we edited the Python proxy to forward the TLS handshake and then terminate the TCP connection with the hub. Consequently, the proxy will establish a connection with the server because the TLS handshake was completed. The proxy will then acknowledge any traffic coming from the server including TCP keepalive packets. The behaviour of this exploit is illustrated in Figure 2.

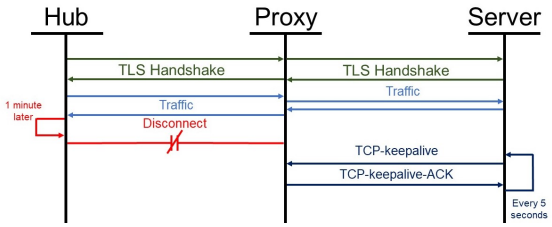
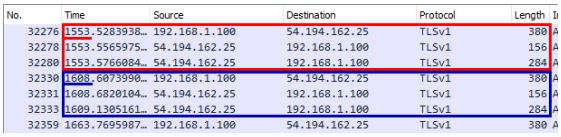


Fig. 2. Behaviour of heartbeat exploit against D-Link hub

4.3 Heartbeats in D-Link Camera

We also analysed D-Link cameras traffic and observed that it sends a 314-byte long heartbeat every 55 seconds and the server responds with two heartbeats of 90 and 218 bytes of data. This pattern is shown in Figure 3. Thereupon, we edited our Python script to selectively forward those 3 heartbeat messages and block everything else.



No.	Time	Source	Destination	Protocol	Length	I
32276	1553.5283938…	192.168.1.100	54.194.162.25	TLSv1	380	A
32278	1553.5565975…	54.194.162.25	192.168.1.100	TLSv1	156	A
32280	1553.5766084…	54.194.162.25	192.168.1.100	TLSv1	284	A
32330	1608.6073990…	192.168.1.100	54.194.162.25	TLSv1	380	A
32331	1608.6820104…	54.194.162.25	192.168.1.100	TLSv1	156	A
32333	1609.1305161…	54.194.162.25	192.168.1.100	TLSv1	284	A
32359	1663.7695987…	192.168.1.100	54.194.162.25	TLSv1	380	A

Fig. 3. Wireshark capture of heartbeats in D-Link camera's traffic

4.4 Heartbeats in Panasonic

We discovered that the heartbeats are sent as 48-byte long UDP datagrams every 15 seconds. Since the heartbeats are sent as UDP datagrams, there was no need

to run a TCP proxy and selective forwarding could be achieved using iptables only. Subsequently, we wrote iptables rules to allow only 48-byte long UDP packets and drop everything else. However, this exploit was only tested when both the hub and the smartphone are on the same LAN. Testing the exploit on WAN was not possible due to unknown technical issues in which the mobile application was not able to access the hub. The exploit succeeded on the LAN and the application displayed that the hub is online. However, the application was completely unresponsive because it could not receive anything apart from heartbeats.

4.5 Heartbeats in Telldus

Telldus TellStick was found to send its traffic in cleartext. Thus, it is possible to capture heartbeats and replay them with a custom HTTP client. This will display that the device is online without any selective forwarding and without even connecting it at all. To validate our assumption, we analysed traffic from Telldus to identify the cleartext heartbeat. Subsequently, we wrote a Python script that connects to the server and sends the same payload every two minutes.

4.6 Heartbeats in Samsung SmartThings

Visan et al. [9] proved that under a DoS attack, Samsung Smartthings provides a time-window of 8 minutes before notifying the user. To build on their work, we decided to apply the heartbeat exploit on the same device to check if it can provide an infinite time window. We analysed network traffic collected from the hub and noticed that both the server and the hub exchange heartbeats every 31 seconds; the hub sends 49 bytes and the server responds with 55 bytes of data.

Unlike the other devices, Samsung SmartThings proved that its heartbeat exchange is robust and secure. The hub disconnected immediately upon selectively forwarding its heartbeats. Subsequently, we analysed the behaviour of the device to understand how it detected the exploit. We discovered that upon blocking non-heartbeat messages, either the device or the server disconnects right after the next heartbeat. This means that the server was reporting in its heartbeat whether it received the last message or not. This allowed the device to detect that the last message was not received by the server, thus it disconnected. As such, we think that the heartbeat messages in this system have some sort of checking whether other (non-heartbeat) messages have passed through or not.

4.7 Summary

The sizes, types and periods of heartbeats of each device are summarised in Table 1. During the experiment, we developed custom proxies for each device and used them to conduct selective forwarding exploits. Once we confirmed our results, we developed a generic selective forwarding proxy (outlined in Section 3) that takes a set of arguments to customise its behaviour.

The results of this exploit proved that the devices can be completely disabled while their corresponding applications still display them as online and operational. This exploit can be considered as an unnoticed DoS attack, because the system was not really available, but the application displayed that it was normally working. The results of the heartbeat exploit are summarised in Table

Table 1. Summary of heartbeat patterns in the analysed devices

Device	Heartbeat Type	Size of Server Heartbeat (bytes)	Size of device heartbeat (bytes)	Period (seconds)
Swann	TCP payload	425	92	10
D-Link	TCP keepalive	N/A	N/A	5
D-Link Camera	TCP payload	90; 218	314	55
Panasonic	UDP datagram	20	20	15
Telldus	TCP payload	"Ping" (cleartext message)	"Pong" (cleartext message)	120
Samsung	TCP payload	55	49	31

2, where the third column states if the device can be physically disconnected during the attack (which indicates a more severe security violation), and the last column presents relevant further details for each device.

Table 2. Summary of heartbeat exploit results

Device	Heartbeat Exploit	Can be physically disconnected	Notes
Swann	Succeeded	Yes	Swann has 2 heartbeats implemented on the Application layer. But the server is not checking for the one coming from the device.
D-Link	Succeeded	Yes	D-Link implements heartbeat on the Transport layer with TCP keepalives sent every 5 seconds.
D-Link Camera	Succeeded	No	D-Link camera employs three heartbeats on the Application layer. The camera sends a heartbeat and the server responds with two heartbeats.
Panasonic	Succeeded	No	On the LAN, both the hub and the mobile app are exchanging heartbeat messages every 15 seconds encapsulated in UDP.
Telldus	Succeeded	Yes	Hub and cloud exchange periodically messages at the application layer in cleartext.
Samsung SmartThings	Failed	No	The hub sends heartbeat messages every 31 seconds and the server responds. But it detects our attempt to attack the heartbeat, and on the next heartbeat, the server closes the connection.

5 Threat Model

In this section, we demonstrate how an attacker can theoretically and practically exploit the vulnerability in the heartbeat exchange. We then describe an automated attack that we have successfully applied to one of the devices.

5.1 Wi-Fi attacks

The easiest way for an attacker to exploit heartbeats is by gaining access to the LAN to which the device is connected. This is relatively straightforward for an attacker to achieve, as many users still use broken protocols like WEP and WPA; many do not even use Wi-Fi encryption at all. According to Wigle, around 4% of access points are unencrypted, 6.3% use WEP and 5.8% use WPA [18]. An attacker can also compromise Wi-Fi networks protected by WPA2 using dictionary attacks like the attack demonstrated in [19]. An attacker can compromise a Wi-Fi AP by exploiting the vulnerability of Wi-Fi Protected Setup (WPS) protocol, which allows association to an AP using an 8-digit PIN. [20].

5.2 DNS Server Hijacking

An attacker can hijack a DNS server and change the records directly. This may be a primary DNS server of the IoT device's vendor or a resolver or cache server of an ISP. In all cases the attacker would be able to exploit the vulnerability in a large group of devices that are served by the compromised DNS server. This attack is infeasible in most situations, but it is still possible and there

are several real-world examples, such as the DNSpionage attack, which targeted governmental and private organisations in Lebanon and the UAE. The attackers redirected email and VPN traffic to an IP address managed by them [21]. Similarly, attackers can redirect heartbeat traffic to a TCP proxy using this attack and eventually exploit the vulnerability in the heartbeat exchange.

5.3 Network Attacks

Theoretically, an attacker can gain access to one or more routers of an ISP by exploiting a vulnerability in the routers operating system. Subsequently, the attacker can use the compromised router to spoof DNS responses or to filter out traffic. This indeed is practically almost infeasible as it is very difficult to gain access to ISP's network devices. Nonetheless, there are some real-world examples that prove this kind of attacks possible in specific situations. For instance, in 2015, FireEye discovered that a group of attackers did manage to take over Cisco routers. [22] Similarly, Cisco has recently discovered and patched serious security holes that allow root access in their SD-WAN software [23]. This proves that such attacks although very difficult, are still possible and can allow attackers to eventually exploit the heartbeat vulnerability against not only one but a large group of devices.

5.4 Automated Attack against Swann using Evil-Twin Wi-Fi AP

For devices that support Wi-Fi connections, attackers can try Evil Twin attacks to force a device to connect to their own LAN on which they can spoof DNS records. We experimented this attack on the Swann hub and successfully managed to force it to connect to our LAN and eventually exploited its heartbeats. In WPA2, a four-way handshake needs to be completed before sending any data, but the handshake cannot be completed because the evil twin does not know the password. Surprisingly, Swann hub connected to the rogue AP anyway and sent DHCP, DNS and TCP traffic normally. After exchanging few packets, the hub then fell back to association request/responses and kept repeating the same behaviour in a loop. An illustration of this behavior is shown in Figure 4.

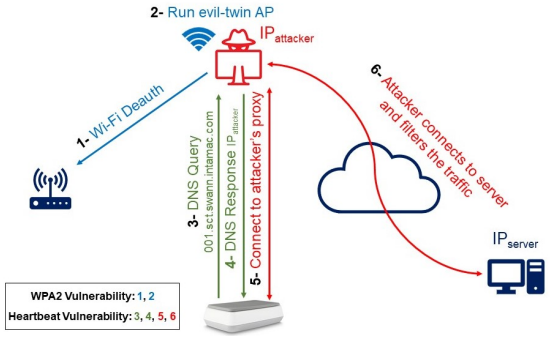


Fig. 4. Combination of the WPA2 and Heartbeat vulnerabilities in Swann.

The IEEE802.11i standard [24] states that an AP and a Station (STA) must exchange four EAPoL messages to complete the 4-way handshake for authentication before sending any data. Based on the IEEE802.11i standard we de-

duced that this is a misimplementation of the standard in the driver of the devices' wireless chipset. By analysing Swann hubs hardware, we discovered that it uses a Jorjin WG7831-D0 Wi-Fi chipset. According to its documentation, the chipset is based on the Texas Instruments WL1831MOD chipset and WiLink8 driver [25, 26].

By exploiting this misimplementation, we managed to achieve a MITM between the hub and the server, and perform the heartbeat attack. The attack was then automated using bash and the Python proxy script. The whole process can be found on GitHub[2]. Consequently, our exploit can disable any Swann device located nearby our attacking machine, without its owner's awareness.

6 Discussion

Exploiting the heartbeat pattern is not critical in all IoT applications like light bulbs for example. However, it can be remarkably critical in home security kits and in some health care applications which are used to remotely monitor patients. Furthermore, this attack can be extended to identify other patterns in encrypted traffic like sensor readings or controller commands and selectively forward and drop those patterns. For instance, attackers can block some control commands in water distribution systems to cause floods, or in electricity distribution systems to cause blackouts. Autonomous vehicles can also be affected by this attack in which attackers filter communications between the vehicles leading to deadly accidents. Therefore, detection and defence mechanisms must be studied and developed to prevent such attacks and to reach a more secure IoT. To exploit heartbeats, an attacker must redirect network traffic to a malicious proxy. Nevertheless, attackers can find many ways to gain access over the network. This was proven by Chapman [27] who compromised WLAN credentials by exploiting a vulnerability in LIFX light bulbs. Alternatively, attackers can compromise Wi-Fi credentials using social engineering techniques. Therefore, it is necessary to secure the heartbeat exchange in IoT, even though its exploit requires network access.

6.1 Recommendations and Countermeasures

Some "secure" implementations of the heartbeat pattern were proposed by different authors. For instance, IBM proposes an implementation that uses freshness and challenge-response mechanisms and provides security against spoofing and replay attacks [28]. However, this solution is not secure against our attack, because the proxy can simply detect the challenge-response pattern due to its periodic nature and selectively forward it, while blocking any other traffic. For that, we propose some recommendations and countermeasures that can make heartbeat exchanges more secure in IoT. Firstly, heartbeats must always be implemented on the application layer instead of the transport layer like TCP keepalive packets. Secondly, heartbeats must include information about the last message that was sent before the heartbeat. This would allow both endpoints to detect if any messages were filtered out. To understand this mechanism better, consider this scenario:

[2] https://github.com/SRJanel/SWO_exploit

- Device sends an encrypted heartbeat. It contains a sequence number of the last message sent, which was a heartbeat too.
- Server confirms that the previous message was indeed a heartbeat.
- Device sends an alert that also contains previous message's sequence number.
- Attacker selectively blocks the alert.
- Device sends a heartbeat message that contains a sequence number of the alert that was blocked.
- Server understands that the alert was not delivered due to the missing sequence number.
- Finally, server notifies the user that alerts are being filtered out.

A prevention mechanism would be to always pass the last message within the heartbeat. This would allow the server to receive the blocked message and detect the attack because it did not receive the message as a stand-alone. Another prevention would be using IPSec as it prevents redirecting traffic to a proxy.

The proposed countermeasure will affect the efficiency of the IoT system especially if the devices have limited resources. This is due to the additional metadata the must be sent in heartbeats or the overhead introduced by IPSec. However, security usually comes at the cost of performance.

6.2 Responsible Disclosure Exercise

We contacted most of the manufacturers of the affected devices. Furthermore, we conducted more in-depth conversation with Swann since its device has another serious flaw. We also contacted Texas Instruments, which manufactured the Wi-Fi chipset used by the Swann device under our investigation.

D-Link confirmed our findings but decided that the occurrence of such attack is uncommon and eventually lowered the development priority of a patch. Swann also confirmed our findings and reported that they will release a security patch for their products. Panasonic stated that the discovered pattern is not a heartbeat, but rather a mechanism to establish a peer-to-peer connection between the device and the mobile application. Thus, they think that there is no need to release a patch. Texas Instruments tried to reproduce our WPA2 results in Swann. They tested several versions of their driver and all proved to be secure. They also sent us an evaluation module to test it ourselves and we do confirm their results. After thorough investigations by Swann, they confirmed that the module was installed by a company that went out of operation, so although the root cause has not been found, the vulnerability most likely stems from the specific WPA supplicant and software used in the product. This means that the company that installed the module is probably responsible for the poor implementation

7 Conclusion

Although the security of IoT devices has substantially improved in the past few years, some vulnerabilities remain undiscovered in many IoT systems. Our paper describes a new threat to home security devices in which an attacker can disable a device while making it appear to be working normally to the user.

We demonstrated that heartbeats can be identified in network traffic even if they were encrypted. Subsequently, we proved that heartbeats can be exploited

using selective forwarding attack. In particular, we redirected network traffic from the devices to a TCP proxy that only forwards heartbeats between the hub and the server and discards any other data. As a result, the user application displays that the device is online because heartbeats are passed. However, the device is in fact disabled because the rest of its traffic was blocked. This allows an intruder to disable home security devices and break into houses without the awareness of their owners.

We applied this exploit on a set of home security devices and most of them were found to be affected by this vulnerability. We disclosed our findings to the affected companies to fix the issue and release the necessary patches. To mitigate this vulnerability, we propose some recommendations to implement a robust and secure heartbeat exchange. Our results confirm the findings of OConnor et al. [13], although our research was carried out independently and on a different set of devices.

Our work proves that further research is still needed for a more secure IoT. In addition, researchers should work closely with industry to ensure that security standards are implemented correctly. To build on this work, selective forwarding exploits should be extended to cover any type of traffic patterns rather than just heartbeats. This would help researchers to develop robust frameworks that can prevent or at least detect any type of selective forwarding.

References

1. D. Evans, "How the internet of everything will change the world...for the better ioe [infographic]," Apr 2013. [Online]. Available: https://blogs.cisco.com/digital/how-the-internet-of-everything-will-change-the-worldfor-the-better-infographic
2. R. v. d. Meulen, "Gartner says 8.4 billion connected "things" will be in use in 2017, up 31 percent from 2016," Feb 2017. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2017-02-07-gartner-says-8-billion-connected-things-will-be-in-use-in-2017-up-31-percent-from-2016
3. M. Antonakakis, T. April, M. Bailey, and et al., "Understanding the mirai botnet," in *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, 2017, pp. 1093–1110.
4. O. Cetin, C. Ganán, L. Altena, and et al., "Cleaning Up the Internet of Evil Things: Real-World Evidence on ISP and Consumer Efforts to Remove Mirai," in *Network and Distributed Systems Security (NDSS) Symposium*, 2019.
5. C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017. [Online]. Available: https://doi.org/10.1109/mc.2017.201
6. K. Zetter, "Flaw in home security cameras exposes live feeds to hackers," Jun 2012. [Online]. Available: https://www.wired.com/2012/02/home-cameras-exposed/
7. L. Kelion, "Swann's home security camera recordings could be hijacked," Jul 2018. [Online]. Available: https://www.bbc.co.uk/news/technology-44809152
8. D. Simmons, "Security holes found in big brand car alarms," Mar 2019. [Online]. Available: https://www.bbc.co.uk/news/technology-47485731
9. B. Visan, J. Lee, B. Yang, A. H. Smith, and E. T. Matson, "Vulnerabilities in hub architecture iot devices," in *2017 14th IEEE Annual Consumer Communications Networking Conference (CCNC)*, Jan 2017, pp. 83–88.

10. E. Fernandes, J. Jung, and A. Prakash, "Security analysis of emerging smart home applications," *IEEE Symposium on Security and Privacy (SP)*, pp. 636–654, 2016.
11. N. Apthorpe, D. Reisman, and N. Feamster, "A Smart Home is No Castle: Privacy Vulnerabilities of Encrypted IoT Traffic," *CoRR*, vol. abs/1705.06805, 2017. [Online]. Available: http://arxiv.org/abs/1705.06805
12. N. Shaun, "This one weird trick turns your google home hub into a doorstop," Nov 2018. [Online]. Available: https://www.theregister.co.uk/2018/10/31/google_home_api
13. T. OConnor, W. Enck, and B. Reaves, "Blinded and confused: uncovering systemic flaws in device telemetry for smart-home internet of things," in *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 2019, pp. 140–150.
14. S. Whalen, S. Engle, and D. Romeo, "An introduction to arp spoofing," Apr 2001. [Online]. Available: http://index-of.es/Misc/pdf/arpspoofing_slides.pdf
15. "Aircrack-ng tool." [Online]. Available: https://www.aircrack-ng.org/doku.php?id=aireplay-ng
16. Wireshark, "Wireshark Tool." [Online]. Available: https://www.wireshark.org/
17. PC Magazine Encyclopedia, "Heartbeat Definition." [Online]. Available: https://www.pcmag.com/encyclopedia/term/44190/heartbeat
18. "All the networks. found by everyone." [Online]. Available: https://wigle.net/
19. O. Nakhila, A. Attiah, Y. Jin, and C. Zou, "Parallel active dictionary attack on wpa2-psk wi-fi networks," in *MILCOM 2015-2015 IEEE Military Communications Conference*. IEEE, 2015, pp. 665–670.
20. S. Viehböck, "Brute forcing wi-fi protected setup," December 2011. [Online]. Available: https://sviehb.files.wordpress.com/2011/12/viehboeck_wps.pdf
21. B. Krebs, "A deep dive on the recent widespread dns hijacking attacks," Feb 2019. [Online]. Available: https://krebsonsecurity.com/2019/02/a-deep-dive-on-the-recent-widespread-dns-hijacking-attacks/
22. T. Greene, "Attackers can take over cisco routers; other routers at risk, too," Sep 2015. [Online]. Available: https://www.networkworld.com/article/2984124/attackers-can-take-over-cisco-routers-other-routers-at-risk-too.html
23. C. S. Advisory, "Cisco security threat and vulnerability intelligence," Jan 2019. [Online]. Available: https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20190123-sdwan-file-write
24. IEEE Computer Society LAN/MAN Standards Committee, "IEEE Standard for Information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11*, 2007.
25. Jorjin, "Jorjin WG7831-D0 Wi-Fi chipset." [Online]. Available: https://www.jorjin.com/product.php?id=79
26. ——, "WG78XX Serial Module - Support Note." [Online]. Available: https://www.jorjin.com/upload/1470892430.pdf
27. A. Chapman, "Hacking into internet connected light bulbs," Jul 2014. [Online]. Available: https://www.contextis.com/blog/hacking-into-internet-connected-light-bulbs
28. "Identifying and Preventing Threats to Your IoT Devices." [Online]. Available: https://developer.ibm.com/articles/iot-prevent-threats-iot-devices/#heartbeat