Systematically Evaluating Security and Privacy for Consumer IoT Devices

Franco Loi[†], Arunan Sivanathan[†], Hassan Habibi Gharakheili[†], Adam Radford[⋆], Vijay Sivaraman[†]

[†]Electrical Engineering and Telecommunications, University of New South Wales, [⋆]Cisco Systems
Sydney, Australia
{f.loi,a.sivanathan}@student.unsw.edu.au,h.habibi@unsw.edu.au,aradford@cisco.com,vijay@unsw.edu.au

ABSTRACT

Internet-of-Things (IoT) devices such as smart bulbs, cameras, and health monitors are being enthusiastically adopted by consumers, with numbers projected to rise to the billions. However, such devices are also easily attacked, or used for launching attacks, at large scale and at increasing frequency. This paper is an attempt at developing a systematic method to identify the security and privacy shortcomings of various IoT devices, with a view towards alerting consumers, manufacturers, and regulators to the associated risks. We categorize the threats along four dimensions: *confidentiality* of private data sent to/from the IoT device; *integrity* of data from the IoT device to internal/external entities; *access control* of the IoT device; and *reflective attacks* that can be launched from an IoT device. We develop scripts to automate the security testing along each of these dimensions, subject twenty market-ready consumer IoT devices to our test suite, and reveal findings that give a fairly comprehensive picture of the security/privacy posture of these devices. Our methodology can be used as a basis for a star-based security ratings system for IoT devices being brought to market.

1 INTRODUCTION

The next wave of growth in Internet connections is slated to come from IoT devices, including household appliances and wearable health monitoring sensors. These devices are being rapidly adopted as they bring benefits to our everyday lives, but their security vulnerabilities are fueling an escalation in the frequency and severity of cyber-attacks. In November 2015 hackers compromised the Hello Barbie doll and violated confidentiality by gaining access to user accounts and encrypted audio [13]. In 2016 a large-scale attack used the Zigbee protocol in the Phillips Hue lightbulb to spread a worm to control other lightbulbs, thereby violating the integrity of the device [12]. In 2017 a hacker with the name "Stackoverflowin" gained illegitimate access to 150,000 printers exploiting the Internet printing protocol (IPP), and was able to send out rogue print jobs [9]. 2016 recorded one of the largest attacks to-date wherein a distributed denial of service (DDoS) attack used an army of compromised IoT devices to bring down the Dyn DNS sever, affecting many popular websites [11]. As each month brings new consumer IoT devices to the market and millions of deployments in households worldwide, new security and privacy attack vectors open up that can be exploited at a scale never seen before.

IoT manufacturers are increasingly warned [8] to embrace and abide by additional security practices to prevent harm to users and businesses. Meanwhile, consumers lack awareness about the potential risks associated with emerging connected devices, and regulatory bodies are being urged to give devices a security score, similar to an energy rating [14]. This can help users make informed decisions while buying IoT products, and also be beneficial to insurance companies in evaluating cyber-insurance claims.

Emerging research work [2, 3, 5, 15] has focused on understanding and identifying potential security and privacy threats for IoT. However there is little research into a systematic way for identifying security flaws in existing and emerging IoT devices. We believe our work is the first to develop a systematic methodology for profiling the security posture of consumer IoT devices, which can lead to a security-star rating that can inform consumers, regulators, and insurance bodies of the associated risks.

In this paper[1] we first develop a suite of security tests categorized under four criteria – *confidentiality* of data sent/received by the IoT device; *integrity and authentication* of connections the IoT device establishes with other (local or external) entities; the *access control and availability* of the IoT device to connection requests; and the capability of the IoT device to participate in *reflective attacks*. Next, we apply our automated security test suite to 20 IoT devices available in the market today, chosen to cover a range of applications including home security (cameras and motion sensor), health (weighing scale, blood-pressure monitor and air-quality sensors), energy management (light-bulbs and power-switch), and entertainment (photo frame, printer and speaker). Finally, using the outputs of our automated test suite, we assign a color-coded security score to each of the devices under each of the four criteria, thereby giving an intuitive visual representation of the device's security posture.

The remainder of this paper is organized as follows: In §2 we present our security test suite under the four criteria listed above, and apply it in §3 to evaluate 20 IoT devices. The resulting security posture is discussed in §4, and the paper is concluded in §5.

IoTS&P'17, Dallas, TX, USA

DOI: 10.1145/3139937.3139938

[1]Funding for this project was provided by the Australian Research Council (ARC) Linkage Grant LP150100666

2 SECURITY TEST SUITE

In this section, we develop a suite of security tests to categorize threats that exploit security/privacy vulnerabilities in IoT devices under four dimensions namely confidentiality, integrity, access control, and reflection.

2.1 Confidentiality

Confidentiality involves ensuring the exchanged data between endpoints cannot be understood by unwanted snoopers. We evaluate the confidentiality of exchanged data using three measures, whether it is *plaintext*, *encoded*, or *encrypted*. We assess all communication channels of a given IoT device – between: device and cloud server; device and user App; user App and cloud server. We therefore wrote a Python script that performs ARP spoofing inside the home network to intercept all traffic to/from the IoT device as well as the user's smartphone.

Encryption protocol: We use this test to determine the security protocol being used for a particular communication channel. The security protocol is obtained by checking the `protocol` field of the packet capture on Wireshark to see if it is identifiable.

Plaintext: After inspecting the protocol field, we analyze the `data` field (i.e. payload) to check if it contains any human-readable text. This test determines whether the data is in plaintext or not, but it does not differentiate between encoded data and encrypted data as both are not human-readable.

Entropy: Since above tests cannot always evaluate the confidentiality of data, we use the entropy test to verify whether a certain communication is encrypted, encoded or in plaintext. Entropy can not only be used to determine whether data is encrypted, but also to assess the strength of encryption. The better the level of encryption the higher the entropy as it will contain more information.

We wrote a Python script that is fed raw data from captured packets to compute the Shannon entropy of the data one byte at a time (i.e. a value between 0 and 8) – we look at the data in bytes. In order to have an accurate entropy value, we use at least 100 KB worth of packets. Our entropy test verifies whether the data is encrypted in conjunction with the encryption protocol test and confirms the plaintext test. We note that this entropy test can fail when processing video from cameras that are indeed unencrypted – it shows a high entropy value due to the video compression.

2.2 Integrity

Integrity assessment ensures a given IoT device performs its intended functions without any manipulation and no message to/from the device is modified without detection. We therefore test the following:

Replay attack: We feed captured packets sent from the user App to the IoT device (using the technique mentioned in §2.1) into our Python script which will then replay them to the IoT device. The attack is successful if the device performs a certain function specified in the packet. Further, if packets are in plaintext (or encoded), we modify certain fields inside the packets and replay them to check whether the device responds to tampered packets.

DNS security: We also test whether the device attempts to connect to an illegitimate server. Inspecting the DNS queries and responses, we assess whether it uses DNSSEC – if not, the device is vulnerable to DNS spoofing attacks. If the device is vulnerable to DNS spoofing we use a python script to perform DNS spoofing redirecting traffic to a fake server. If the device attempts to connect to this fake server, the system integrity is violated. Further, if it sends information to the fake server it indicates the device does not conduct any form of authentication.

2.3 Access Control and Availability

We consider the access control and availability of an IoT device to identify how easily an attacker can gain access/control to/of the device and determine whether it is susceptible to a denial of service (DoS) attack. We start our test by scanning for ports that are open on the device using command `nmap -sS -sU -p 0- 65535 [deviceIP]`. We then attempt to gain access via Telnet, SSH and HTTP using a list of known weak login credentials – these ports were exploited recently by the Mirai botnet that resulted in one of the largest DDoS attacks from IoTs over the Internet [6].

Denial of Service: We also assess the ease of launching a DoS attack. We determine how much incoming traffic the IoT device can handle before it completely loses its expected functionality. We flood the device with ICMP `ping` requests as well as UDP packets, and determine the amount of data that is required to stop the operation of the IoT. We conduct these two tests using the `hping3` tool by issuing the command: `hping3 -d 1000 -1 (1 for ICMP and 2 for UDP) -p (port) (deviceIP)`. We also use another python script to measure the maximum number of concurrent TCP connections the device can handle before it crashes – by flooding the device with TCP SYN packets to initiate connections to the list of open ports on the device.

2.4 Reflection

Following public announcement of the large DDoS attack fueled by IoTs in 2016 [6] many manufacturers have consequently closed their remote access ports, or strengthened their default login credentials. We have shown that IoT devices can still be employed to launch DDoS attacks by exploiting various protocols using source-spoofed traffic [4]. We write a python script that crafts malformed packets (with spoofed source IP address) and sends; (a) ICMP messages, (b) SSDP broadcasts, and (c) SNMP requests to a given IoT device. For the SNMP, we further check if the device supports the SNMP public community string that can potentially generate a larger volume of responses. If successful, we issue a `getBulk` SNMP request that sends multiple `getNext` requests at once. Responding to each of these protocols reveals that the device can be used to launch a reflection attack.

3 SECURITY TESTING OF IOT DEVICES

We now validate our assessment methodology by applying it to twenty IoT devices that have been recently introduced to the consumer market, ranging from cameras and lightbullbs to power switches and health monitoring devices. We verify our methodology on some devices with known security flaws [2] and also evaluate the security and privacy posture of other IoT devices with security vulnerabilities that are unknown to us.

3.1 Confidentiality

Our confidentiality assessment results are shown in Table 1 by three measures over three communication channels (as discussed in §2.1). It can be seen that most of devices have fairly secure communication in two channels namely device-to-server and user-App-to-server (i.e. less plaintext, secure protocols of TLS/SSL, high

Table 1: Posture of confidentiality and integrity

	Confidentiality: Device to Server			Confidentiality: Device to User-app			Confidentiality: User-App to Server			Integrity and authentication		
Devices	**Plaintext**	**Protocol**	**Entropy**	**Plaintext**	**Protocol**	**Entropy**	**Plaintext**	**Protocol**	**Entropy**	**Replay Attack**	**DNS spoofing**	**Fake Server**
Phillip Hue lightbulb	No	AES	7.70	Yes	None	5.48				Yes	Yes	HTTP
Belkin Switch	Partially	Unknown	7.74	Yes	None	5.16				Yes	Yes	Fail SSL
Samsung Smart Cam	No	Unknown	7.99				No	Unknown	7.91		Yes	Fail SSL
Belkin Smart Cam	No	Unknown	7.06	No	SSL	7.95	No	SSL	7.48		Yes	Fail SSL
Awair air monitor	No	SSL	7.89				No	SSL	7.90		Yes	Fail SSL
HP Envy Printer				Yes	None	5.38				Yes	Yes	Fail SSL
LiFX lightbulb				No	Unknown	4.66	No	SSL	7.64	Yes	Yes	Plaintext
Canary Camera	No	TLSv1.2	7.96				No	TLSv1.2	7.46		Yes	Fail SSL
TP Link Switch	No	Unknown	7.95	No	Unknown	5.33	No	SSL	7.63	Yes	Yes	Fail SSL
Amazon Echo	No	TLSv1.2	7.98				No	TLSv1.2	7.91		Yes	Fail SSL
Samsung Smart Things	No	TLSv1.2	7.69				No	TLSv1.2	7.80		Yes	Fail SSL
Pixstar Photo Frame	No	TLSv1.2	7.87								Yes	Fail SSL
TP Link Camera	No	Unknown	7.97	Yes	None	7.51	No	TLSv1.2	7.73		Yes	Fail SSL
Belkin Motion Sensor				Yes	None	5.16				No		
Nest Smoke Alarm	No	Unknown	7.25				No	TLSv1.2	7.54		Yes	Fail SSL
Netamo Camera	No	IPsec	8.00	Partially	HTTP	7.97	No	TLSv1.2	7.98		Yes	Fail Ipsec
Dlink Camera	Yes	None	5.40							No		
Hello Barbie Companion	No	TLSv1.2	7.99								Yes	Fail SSL
Whithings Sleep Monitor	No	Unknown	7.84				No	TLSv1.2	7.63		Yes	Fail SSL
Nest Drop Camera	No	TLSv1.2	7.99				No	TLSv1.2	7.94		Yes	Fail SSL

entropy values). However, a majority of the vulnerabilities arise when the device communicates with the user App (i.e. five devices send in plaintext, only one device uses SSL, fairly lower entropy values). Note that for some devices (e.g. Belkin Switch, Samsung Smart Cam), the security protocol is not identified but together with plaintext and entropy tests, we can evaluate the confidentiality of a given channel. Considering the user privacy, we see quite a few devices such as Phillips Hue lightbulb, Belkin power switch, HP Envy printer, TPLink camera, and Belkin motion sensor, communicate in plaintext (some of them were discussed in [15]), – revealing private information, for example, whether the Belkin power switch is on/off, or when Phillips Hue lightbulb was last used.

Our results also enable us to discover new vulnerabilities in some devices such as TPLink camera. Fig. 2, in Appendix A, depicts a detailed insight into packets captured from the TPLink camera (i.e. a `POST` request packet payload in red text followed by the `HTTP` response packet in blue text). The video/audio stream is sent in plaintext (the video/audio header is human-readable even though its data doesn't seem human-readable). This data can be sniffed by an attacker and then used to reassemble the video/audio data. Surprisingly, it is revealing not only the video/audio data but also the authentication password required for logging-in to the device. This password is exposed in the basic authentication field of the packet shown in Fig. 2 (i.e. `YWRtaW46WvdSdGFND0=`) – this is a Base64 version of "admin". Given the password, we are able to log into the device by simply guessing the user-name as "admin" which is a common default credential used in many IoT devices.

The efficacy of our entropy measure can be seen in the LiFX lighbulb. Our plaintext test for this device shows that the LiFX bulb is not communicating in a human-readable format, whereas its traffic data has a low entropy value of 4.56. When taking a closer look into the LiFX packets, we are able to discover that packets associated with certain commands (from the user App) are identical and certain bits represent specific functions of the device, meaning that the data is just encoded as shown in Fig. 3. Similarly in the TPLink power switch, we see that the data is not in plaintext but the entropy value is 5.33, suggesting that it could possibly be encoded or poorly encrypted. By guessing that the data is sent in JSON format (i.e. `{data}`), we attempt to XOR the first byte with the character "{" to obtain the single byte key. We then apply the key to the encrypted message and are able to extract the message in plaintext. This indicates a weak encryption is used in the TPLink power switch. Note that some devices employ stronger encryption protocols. For example, Amazon Echo uses TLSv1.2 for all traffic it communicates (shown in Fig. 4), or Netamo camera implements IPsec, protecting the IP address of endpoints from potential attackers (shown in Fig. 5 in Appendix A).

Lastly, we evaluate the confidentiality of devices' communication after their initial setup phase is complete. There are, however, some devices that communicate in an insecure manner when they initially pair with the user App. For example, Fig. 1 shows that Belkin camera exposes the password of the local WiFi network in plaintext (i.e. `ThisIsMyWiFiPassword` in Fig. 1) when responding to a `GET` request.

3.2 Integrity and Authentication

Our assessment results for the posture of integrity and authentication in twenty IoT devices are shown by last three columns in Table 1. Considering the test for replay attacks, five of our IoT devices are susceptible such as Philips Hue lightbub, Belkin power switch, HP Envy printer, LiFX lightbulb, and TPLink switch. Some of these exploits have been already reported. For example, the Belkin switch was evaluated to be insecure against replay attacks due to the lack of authentication [15] or the LiFX lightbulb that communicates encoded messages with the user App [1]. An attacker can turn on/off the Belkin switch with a well-crafted fresh packet, or change the color/brightness of the LiFX bulb using the control bit pattern shown in Fig. 3. On the other hand, those IoT devices

Table 2: Posture of access control and availability

Devices	Open Ports (TCP)	Open Ports (UDP)	Vulnerable Ports	Weak Passwords	ICMP DoS	UDP DoS	Number of TCP Connections	ICMP Reflection	SSDP Reflection	SNMP Reflection
Phillips Hue lightbulb	80, 8080	1900, 5353	80	No	Protected	Protected	112	Yes	Yes	No
Belkin Switch	53, 49155	53, 1900, 3111, 7638, 13965, 14675, 17143, 19422, 22894, 23835, 26011, 27047, 38849, 40014, 41970, 42518, 43403, 47836, 53121, 53330, 55353, 65484	None		23Mbps	6.3Mbps	97	Yes	Yes	No
Samsung Smart Cam	80, 443, 554, 943, 4520, 49152	161, 5353	80	No	90Mbps	4.1Mbps	17	Yes	No	v2c
Belkin Smart Cam	80, 81, 443, 9964, 49153	1900, 10000, 13105, 19827, 26854, 28971, 32596, 32435, 33435, 35042, 35316, 35056, 36500, 36943, 38587, 38606, 39632, 39714, 43588, 43834, 47709, 48190, 44179, 49156, 49201, 49360, 52042, 52144, 52603, 55254, 56284	80	No	7.7Mbps	74Kbps	256	Yes	Yes	No
Awair air monitor	Filtered	Filtered			36Mbps	7.2Mbps		Yes	No	No
HP Envy Printer	80, 443, 631, 3910, 3911, 8080, 9100, 9220, 53048	137, 161, 543, 3702, 5353, 5355, 7235, 53592, 56693, 56723	80, All ports allow telnet	No			1	Yes	No	v1
LiFX lightbulb	Closed	Filtered			6Mbps	82Kbps		No	No	No
Canary Camera	Closed	Closed			6.4Mbps			Yes	No	No
TPLink Switch	80, 9999	1040	80	No	5.5Mbps	25Mbps	15	Yes	No	No
Amazon Echo	4070	5353	None		Protected	9.2Mbps	258	Yes	No	No
Samsung Smart Things	23, 39500	Filtered	23	No	130Mbps	8.8Mbps	1	Yes	No	No
Pixstar Photo Frame	Closed	137			Protected	Protected		Yes	No	No
TPLink Camera	80, 554, 1935, 2020, 8080	1068, 3702, 5353, 42941	80	Yes	48Mbps	870Kbps	130	Yes	No	No
Belkin Motion Sensor	53, 49152	53, 1900, 3080, 3081, 3082, 3179, 3229, 3236, 3619, 4050, 4052, 4053, 4054, 4055, 4289, 4996, 4997, 4998, 14675	None		11.3Mbps	350Kbps	109	Yes	Yes	No
Nest Smoke Alarm	Closed and filtered	17395, 17466, 17471, 18184, 18234, 18455, 18721, 18916, 19090, 19112, 19217, 19458, 19581			Protected	Protected		Yes	No	No
Netamo Camera	80, 5555	654, 7242, 26082, 29110, 31574, 35826, 39408, 46721, 48080, 56943	80	No	8.2 Mbps	45Kbps	256	Yes	No	No
Dlink Camera	21, 23, 5001, 5004, 16119	1900, 5002, 5003, 10000	5004	No Password	49Mbps	292Kbps	20	Yes	Yes	No
Hello Barbie Companion	Closed	Closed			10Mbps			Yes	No	No
Whithings Sleep Monitor	22, 7685, 7888	5353	22	No	Protected	Protected	22	Yes	No	No
Nest Drop Camera	Closed	Closed of filtered			4Mbps			Yes	No	No

that employ secure protocols (e.g. SSL) are protected against replay attacks such as Awair air monitor and Amazon Echo.

Our DNS security test results show that none of twenty IoT devices implements DNSSEC protocol that is primarily designed to prevent DNS spoofing attacks. This vulnerability enables attackers to hijack the DNS query and possibly impersonate the legitimate server to the IoT device. Even if DNS spoofing is successful, the victim IoT device may protect itself by some form of authentication. According to the last column of Table 1, some devices such as Phillips Hue lightbulb and LiFX bulb do communicate with the fake server, after a successful DNS spoofing. Phillips Hue lightbulb sends an HTTP message to the fake server that is listening on the same port as the real server, while the LiFX bulb sends data to our fake server which appears to be in its own unique data format (as shown in Fig. 3).

3.3 Access Control and Availability

Our access control evaluation results shown in Table 2 indicate that almost all devices have some form of vulnerabilities in terms of open ports which enable intruders to communicate with or access into the device. For example, Belkin smart camera exposes a large number of ports, 5 TCP and 31 UDP. Another vulnerable device is HP printer with 9 open TCP ports and 10 open UDP ports. Among all these open ports, we note that HP printer responds on a special TCP port 9100 that is used for printing with no authorization – this vulnerability was recently exploited to attack more than 150000 printers [10]. On the other hand, a device like Awair air monitor has all ports closed so is protected against common attacks such as SYN flooding.

We note that some IoT devices allow remote access via SSH (port 22), Telnet (port 23), or HTTP(port 80). Until recently, many IoT devices had weak credentials (from a list of about 60 common defaults) that Mirai malware [7] exploited to hijack hundreds of thousands of IoTs, launching a major DDoS attack on the Internet. None of these 60 defaults was valid when we used for our twenty IoT devices. Surprisingly, we have two devices with no protection for remote access: HP printer allows Telent without asking for a password, and DLink camera asks for no credentials during SSH access – some manufacturers seemingly open remote access ports for testing/debugging purposes.

From DoS attack test results shown in Table 2, it can be seen that most devices are susceptible to at least one form of DoS attacks, either of ICMP-, UDP- or TCP-based. We note that the required traffic rate to cause a device to stop functioning is not significant in many cases specially when UDP is used (i.e. less than 1 Mbps for Belkin SmartCam, LiFX lightbulb or TPLink camera). For Samsung Smart camera, it can handle ICMP traffic rate up to 90 Mbps, however it stops functioning (the camera will not be able to transmit live video stream to the user App), if it is bombarded by UDP-based traffic at a rate more than 4.1 Mbps.

3.4 Reflective Attacks

Lastly, we consider ICMP, SSDP and SNMP protocols checking if a given device reflects traffic of these types. Our results are shown by right three columns in Table 2. We can see that all devices, except LiFX lightbulb, are reflecting ICMP traffic. We then test the SSDP protocol which is commonly enabled in many IoT devices for ease of discovery. When we use SSDP, the reflected traffic (i.e. response) is amplified by a large factor since it contains service and

Table 3: Security rating

	Confidentiality										Integrity and Authentication			Access Control							Reflection Attacks			
	Device to Server			Device to Application			Application to Server			All														
Devices	Plaintext	Protocol	Entropy	Plaintext	Protocol	Entropy	Plaintext	Protocol	Entropy	Privacy	Replay Attack	DNS Spoofing	Fake Server	Open Ports (TCP)	Open Ports (UDP)	Vulnerable Ports	Weak Passwords	ICMP DoS	UDP DoS	No. of TCP Connections	ICMP Reflection	SSDP Reflection	SNMP Reflection	SNMP Public Community String
Phillip Hue lightbulb	A	A	A	C	C	C	A	A	A	C	C	C	C	C	C	C	A	B	C	C	C	C	A	A
Belkin Switch	B		A	C	C	C	A	A	A	C	C	C	A	C	C	A	A	C	C	C	C	C	A	C
Samsung Smart Cam	A		A	A	A	A	A	A	A	A	A	C	A	C	C	C	A	C	C	C	C	A	C	C
Belkin Smart Cam	A		A	A	A	A	A	A	A	A	A	C	A	C	C	C	A	C	B	C	C	C	A	A
Awair air monitor	A	A	A	A	A	A	A	A	A	A	A	C	A	B	B	A	A	C	C	A	C	A	A	A
HP Envy Printer	A	A	A	C	C	C	A	A	A	C	C	C	A	C	C	C	A	A	A	C	C	A	C	A
LiFX lightbulb	A	A	A	A		C	A	A	A	A	C	C	C	A	B	A	A	C	B	A	A	A	A	A
Canary Camera	A	A	A	A	A	A	A	A	A	A	A	C	A	A	A	A	A	C	A	A	C	A	A	A
TPLink Switch	A		A	A		C	A	A	A	A	C	C	A	C	C	C	A	C	C	C	C	A	A	A
Amazon Echo	A	A	A	A	A	A	A	A	A	A	A	C	A	C	C	A	A	B	C	C	C	A	A	A
Samsung Smart Things	A	A	A	A	A	A	A	A	A	A	A	C	A	C	B	C	A	C	C	C	C	A	A	A
Pixstar Photo Frame	A	A	A	A	A	A	A	A	A	A	A	C	A	A	C	A	A			A	C	A	A	A
TPLink Camera	A		A	C	C	A	A	A	A	C	A	C	A	C	C	C	C	C	B	C	C	A	A	A
Belkin Motion Sensor	A	A	A	C	C	C	A	A	A	C	A			C	C	A	A	C	B	C	C	C	A	A
Nest Smoke Alarm	A		A	A	A	A	A	A	A	A	A	C	A	B	C	A	A			A	C	A	A	A
Netamo Camera	A	A	A	B	C	A	A	A	A	A	A	C	A	C	C	C	A	C	B	C	C	A	A	A
Dlink Camera	C	C	C	A	A	A	A	A	A	A	A			C	C	C	C	C	B	C	C	C	A	A
Hello Barbie Companion	A	A	A	A	A	A	A	A	A	A	A	C	A	A	A	A	A	C	A	A	C	A	A	A
Whithings Sleep Monitor	A		A	A	A	A	A	A	A	A	A	C	A	C	C	C	A			C	C	A	A	A
Nest Drop Camera	A	A	A	A	A	A	A	A	A	A	A	C	A	A	B	A	A	C	A	A	C	A	A	A

presence information of the IoT device – this makes it an attractive protocol for DDoS attackers. We observe that five of our devices are vulnerable to SDDP reflection attacks – rest of them do not use SSDP for discovery. Lastly, we examine SNMP protocol which is not widely used by IoT devices. Furthermore, with SNMP v2c (and v3), it is possible to use public community strings such that the amplification factor is significantly high. The SNMP v2c is only available in the Samsung Smart camera. Sending a `getBulk` request to the camera, it will iterate the `getNext` request multiple times, hence a larger amount of traffic is generated.

4 SECURITY RATING OF IOT DEVICES

Without doubt, hundreds of consumer IoT devices are going to emerge in the years ahead, and their security/privacy vulnerabilities are going to be diverse. Our results from evaluation of the twenty devices highlight the security posture of consumer IoTs, and reveal the problems that users have to deal with. In this section we discuss how our methodology can be used for a security ratings system that is beneficial to consumers or insurance companies. We propose a three-level rating: "A" being secure, "B" being moderately secure/insecure, and "C" being insecure. Table 4 shows our attempt to rate each of IoT devices that we assessed their security posture on the four dimensions – all ratings in this table are subjective and given based on authors perceptions. One may consolidate our table by giving weights to each dimension in the future.

We use color codes for ease of visualization, green for A rating, yellow for B rating, and red for C rating. We also use gray color for cells where the data is not available. For example, the encryption protocol of Belkin switch is not identified on Wireshark for the device-to-server communication; DNS query is not performed in Belkin motion sensor; normal functionality of the Pixtar photo frame is not affected by a DoS attack. Using our color-coded ratings table, consumers are able to quickly visualize the security posture of individual devices. All devices display some form of vulnerability in either of integrity, access control and reflection dimensions - this raises concerns for consumers as well as for the Internet ecosystem in general. Devices such as the Amazon Echo, Hello Barbie, Nest Dropcam, Whitings Sleep monitor seem relatively secure by the measure of confidentiality. Amazon Echo in particular is a top-rated device in security with encrypted communication channels and having almost all of its ports closed. On the other hand, devices such as Phillips Hue lightbulb and the Belkin switch seem fairly poor in security. The Phillips Hue in particular communicates in plaintext to the user App, is susceptible to replay attacks, has many open ports and can be used to launch various reflection attacks to victim servers.

We recognize that security is but one concern amongst many that manufacturers of IoT devices are dealing with. The surge in demand for IoT is leading many manufacturers to rush to market with their product, and increasing user appeal to gain market traction can become more paramount than ensuring fool-proof security. No matter how it evolves, consumers would eventually demand for a rating system (much like the energy rating system given to home appliances) that needs to be developed by standard bodies and tracked by regulation entities. This would protect consumers rights and incentivize manufacturers to improve the security of their device to receive an acceptable rating that can lead to a good share of the market.

5 CONCLUSION

The increasing uptake of consumer IoT devices poses security and privacy concerns at an unprecedented level. Unlike many prior works that have speculated on the risks or prescribed point-solutions, we have developed a systematic method to evaluate the security posture for a range of IoT devices available in the market today. We based our evaluation on four pillars of confidentiality, integrity, accessibility and reflection capability. We showed the types of security and privacy threats that are likely to be of most concern in each pillar. We believe our approach provides a useful starting point for evaluating any IoT device to come to market, particularly since this market is going to get more diverse and complex, and also informs both manufacturers and users on the likely threats for their range of IoT devices. As part of future work we are investigating more IoT devices in both the consumer and industrial space.

REFERENCES

[1] D. Hall. 2015. Building a LIFX packet. https://community.lifx.com/t/building-a-lifx-packet/59. (2015).

[2] N. Dhanjani. 2015. *Abusing the Internet of Things: Blackouts, Freakouts, and Stakeouts.* O'Reilly Media.

[3] Fernandes et al. 2016. Security Analysis of Emerging Smart Home Applications. In *Proc. IEEE Symposium on Security and Privacy (SP).*

[4] M. Lyu et al. 2017. Quantifying the Reflective DDoS Attack Capability of Household IoT Devices. In *Proc. ACM WiSec.* Boston, Massachusetts.

[5] T. Yu et al. 2015. Handling a Trillion (Unfixable) Flaws on a Billion Devices: Rethinking Network Security for the Internet-of-Things. In *Proc. ACM HotNets.*

[6] Herzberg et al. 2016. Breaking Down Mirai: An IoT DDoS Botnet Analysis. https://www.incapsula.com/blog/malware-analysis-mirai-ddos-botnet.html. (2016).

[7] J. Gamblin. 2016. Mirai Source Code. https://github.com/jgamblin/Mirai-Source-Code/tree/master/mirai/bot. (2016).

[8] M. Mimoso. 2017. Legislation Proposed to Secure Connected IoT Devices. https://goo.gl/fUQzaT. (2017).

[9] Ms. Smith . 2017. A Hacker just Pwned Over 150,000 Printers Left Exposed Online. https://goo.gl/1SZSUw. (2017).

[10] Ms. Smith. 2017. Hacker stackoverflowin pwning printers. https://goo.gl/4E4e71. (2017).

[11] N. Woolf. 2016. DDoS attack that disrupted internet was largest of its kind in history, experts say. https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet. (2016).

[12] R. Eyal et al. 2016. *IoT goes nuclear: Creating a ZigBee chain reaction.* Technical Report.

[13] S. Gibbs. 2015. Hackers can hijack Wi-Fi Hellow Barbie to spy on your children. https://goo.gl/p2Pfk9. (2015).

[14] S. Gibbs. 2017. Smart fridges and TVs should carry security rating, police chief says. https://goo.gl/P9MsA6. (2017).

[15] S. Notra et al. 2014. An Experimental Study of Security and Privacy Risks with Emerging Household Appliances. In *Proc. First International Workshop on Security and Privacy in Machine-to-Machine Communications (M2MSec).*

APPENDIX A WIRESHARK CAPTURES

```
GET /apcam/for-android/aplist.asp HTTP/1.1
Host: 10.68.68.22
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11;
rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/
xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Authorization: Basic YWRtaW46YWRtaW4=
Connection: keep-alive
Upgrade-Insecure-Requests: 1

HTTP/1.0 200 OK
Date: Thu Jan  1 02:35:43 1970
Server: Camera Web Server
Pragma: no-cache
Cache-Control: no-cache
Content-type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
====================================================
      var currentbssid="f6:f2:6d:22:9d:00";
      var currentencrytype="AES";
      var currentmode="WPA2PSK";
      var currentkey="0";
      var currentwpapsk="ThisIsMyWiFiPassword";
var hasrj45="0";
var usewps="0";
```

Figure 1: Belkin camera is pairing with user app

```
POST /stream/video/h264?resolution=VGA HTTP/1.1
Connection: close
Authorization: Basic YWRtaW46WVdSdGFXND0=          Base64 encoded
User-Agent: Dalvik/2.1.0 (Linux; U; Android 5.0; SM-N9005
Build/LRX21V)
Host: 192.168.1.143:8080
Accept-Encoding: gzip
Content-Type: application/x-www-form-urlencoded
Content-Length: 0

HTTP/1.0 200 OK
Server: Streamd,F4F26D9351F1
Date: Fri, 09 Sep 2016 05:19:43 UTC
Content-Type: multipart/x-mixed-replace;boundary=video-
boundary--
Pragma: no-cache
Cache-Control: no-cache
Content-Length: -1
Connection: close                                  HTTP payload

--video-boundary--
Content-Type: video/x-h264
Content-Length: 18732                              video/audio header
Date: 2016-09-09 15:19:43 GMT
X-Timestamp: 617.673270
X-Audio: 0
X-Params: {"DayNight": 1}
                                                   video/audio data
....gM@..T........h.8.....e..@...N...`..E'.... ..
%...V..*^...R...H........v ..<.wY.!...7i{....f
+........A>.....3^.....\#....M.....Q)...H...
\|..g...gA.,....=z.o]..{2X4.'.R.-.Si..)Z..l ..b9`.
+r....`7...P.W.I.j.=.z...p     z?.#_..
7..O.Z...U...~&2.L.~..<.$.8..  )s...(.^.v{.XEG.....
1..8......(C ...a.V;.........+.........H.*E...m.8v..1
..b.,B..l=.J+..*....B......{.M5.A.
9...w.<.......p_C...x......T..f.............SS....go.....
c....'...a.O.F<..
```

Figure 2: TPLink camera POST message

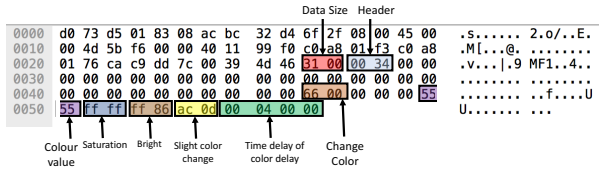

Figure 3: Control bit pattern of LiFX lightbulb

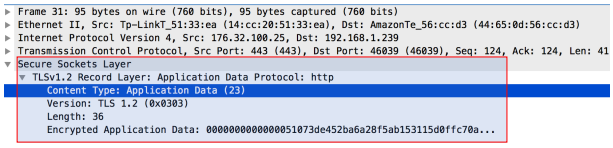

Figure 4: Wireshark capture of Amazon Echo

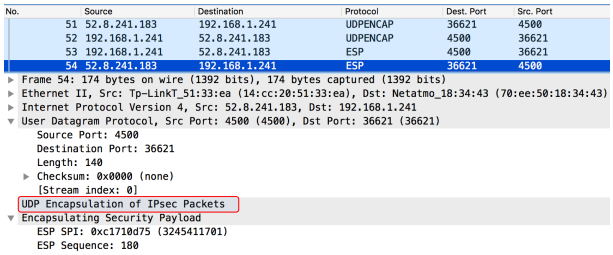

Figure 5: Wireshark capture of Netatmo camera