A Castle of Glass: Leaky IoT Appliances in Modern Smart Homes

Andy Sun, Wei Gong, Ryan Shea, and Jiangchuan Liu

Abstract

The ubiquity of 802.11 WiFi and the miniaturization as a result of Moore's law has recently enabled the success of IoT. From smart lightbulbs to smart toasters, many home appliances are now becoming both Internet-enabled and interconnected through WiFi. Soon, these futuristic smart homes will be able to run themselves, allowing the human operators to be fully in control of their homes — or will they? Despite the physical advancements made since the '90s, the same cannot be said of the vulnerabilities of these smart devices. We analyze a set of common smart home appliances — a lightbulb, power switch, motion sensor, security camera, and home assistant — putting their vulnerabilities to the test to see what a 21st century home intruder could discover.

Introduction

Moore's law states that the number of transistors on a microchip doubles approximately every two years. Reinterpreted, it can also be stated that as the number of transistors on a chip are held constant, the area of the chip halves approximately every two years. Over the past two decades, this increase in circuit density has directly resulted in today's level of embedded computing devices and the resulting Internet of Things (IoT). From smart lightbulbs to smart toasters, many home appliances are now compute-enabled, becoming both interconnected and Internet-aware through the usage of WiFi. The long sought-after dream of complete home automation is no longer just a work of science fiction, but is slowly becoming a reality as 30 billion IoT devices are expected to be online by 2020 [1]. The idea is that these sensor laden "things" are able to detect and send information about their physical surroundings to other nearby devices, allowing them to optimally adjust for a given situation. The motivating scenario is that one day, as you come home from work, your smart home will turn on the lights, prepare the kitchen, and open the garage door as you enter — taking home amenities and luxury to the next level. Similarly, while you are away, the smart house provides protection in the form of wireless security cameras and motion sensors that alert you as potential intruders are detected, or as a reminder that the mailman has left a package on the porch — all neatly accessible only by you and only from your smartphone.

However, despite the idyllic scene presented, we often overlook the heavy implications of having a multitude of devices constantly broadcasting data over the air (OTA) and the illusion of security that is presented by these IoT devices. Mirai, a highly successfully IoT botnet trojan first detected in 2016, preys on weakly configured IoT devices to use as zombies for carrying out distributed denial of service (DDoS) attacks that at its peak reached 1.1 Tb/s [2, 3]. There has been plenty of recent discussion around the security of these devices and how IoT manufacturers should be hardening them [4]. One method recently proposed in [5] recommends utilizing a hypervisor-level memory inspector to detect foreign processes. Another proposes using contextual text extraction and analysis to detect phishing attacks [6]. However, there has been little attention on the physical OTA aspect of their vulnerabilities despite concerns about practical security issues in Internet-enabled appliances being raised as early as 2001 [7]. As a result, these modern day smart homes are not much different than a castle of glass: an intimidating, towering paragon of defense, but indefensible with its inner workings exposed and fragile walls easily cracked.

In this article, we conduct a network- and data-link-level traffic analysis on IoT devices that one could reasonably consider common in a modern smart home: a lightbulb, a power socket, a motion sensor, a security camera, and a central hub. We then present a method that allows for localization of these devices and develop various strategies that an attacker could employ depending on the desired outcome. Finally, we discuss the security implications of IoT devices and issues that have yet to be addressed.

Methodology

We assume an active network threat model where an adversary is capable of both network sniffing and injection, but not capable of physical access to any device. Conversely, we model the target to try to best match what could be considered as a typical home setup: an 802.11 network using pre-shared keys with a number of IoT devices connected. We also assume that they are able to obtain and analyze similar devices beforehand, either through their own IoT device laboratory or through medium access control (MAC) address identification.

Since our attacker is not assumed to have the capabilities to decrypt the 802.11 frames, any form of network data injection is limited to control frames or random-content data frames. Furthermore, we are more focused on a side-channel attack using the data leaked out in WiFi frames rather than a network attack based on the typical data path of these devices (Fig. 1). As a result, the

Digital Object Identifier:
10.1109/MWC.2017.1800120

The authors are with Simon Fraser University; Wei Gong is also with University of Science and Technology of China.

1536-1284/18/$25.00 © 2018 IEEE

Device	OS	Ports	Service
Central hub	Linux 2.6.32 – 3.10	8008 8009 9000 10001	HTTP?
Lightbulb	Linux 2.4.X|2.6.X	9999	HTTP
Motion sensor	Linux 2.6	80	HTTP
Power switch	Unknown Linux	6668	MQTT?
Security cam	Linux 2.6.32 – 3.10	80 554 1935 8080	HTTP RTSP gSOAP 2.8

TABLE 1. Nmap scan results.

adversary's network injection attacks will be limited to a deauthentication attack, either to disable Internet connectivity of the IoT device or to coerce it into connecting to an attacker-controlled spoofed WiFi access point (AP).

A deauthentication attack can be carried out, in general, without knowledge of the encryption keys. This is due to the disassociation and deauthentication management frames being sent primarily in cleartext until a recent amendment to 802.11 included protected management frames [8]. However, for reasons of backward compatibility, these control frames can still be sent over plaintext if the client does not support the updated specifications. Additionally, we have observed that many devices on 802.11n which have been manufactured after the ratification of [8] still do not support these protected management frames.

Unfortunately, for the purposes of analysis, having only encrypted WiFi frames is not particularly helpful as only metadata can be extracted and only loose correlations observed. Since we have assumed that our adversary is able to conduct device and traffic analysis in their own device lab, we have collected both the encrypted and unencrypted traffic to better correlate the observed data against the device's operation and conjecture causality. Additionally, we want to stress that none of the outlined attacker methods uses the unencrypted data.

Network Traffic Analysis

Using a Raspberry Pi and two antennas (Alfa AWUS036NH, Asus USB-N13), we deployed a WiFi AP on the Asus, dumping the traffic using tcpdump, and sniffed the raw WiFi traffic using the Alfa. The aforementioned smart devices were connected to this AP, and unless specified otherwise, were turned on throughout the experiments below. Each device was then set up to connect to a Samsung Galaxy S5 smartphone, and the central hub if the device supported it. We also ran an nmap scan against all the devices to identify open ports and services (Table 1). With this data, we demonstrate a technique of fingerprinting each device based on its unique properties and operational domain that can be used for other similar devices and is not strictly limited to this representative testbed. Figure 2 depicts our IoT device lab setup, with each of the analyzed devices. In the future, we would like to explore the possibilities of leveraging machine learning to automate this process.

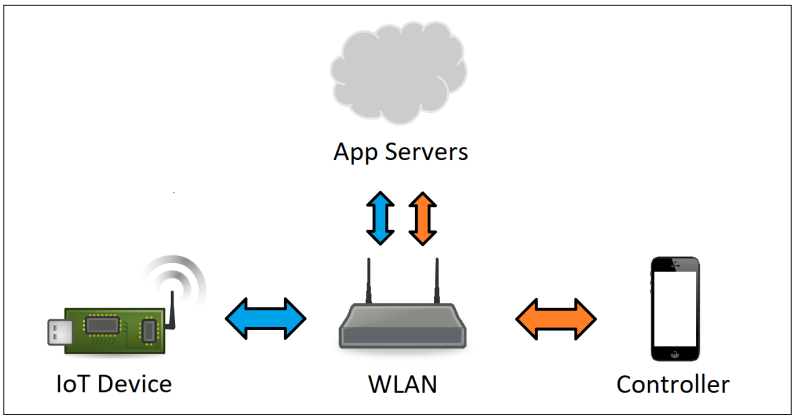

FIGURE 1. Typical IoT device data path.

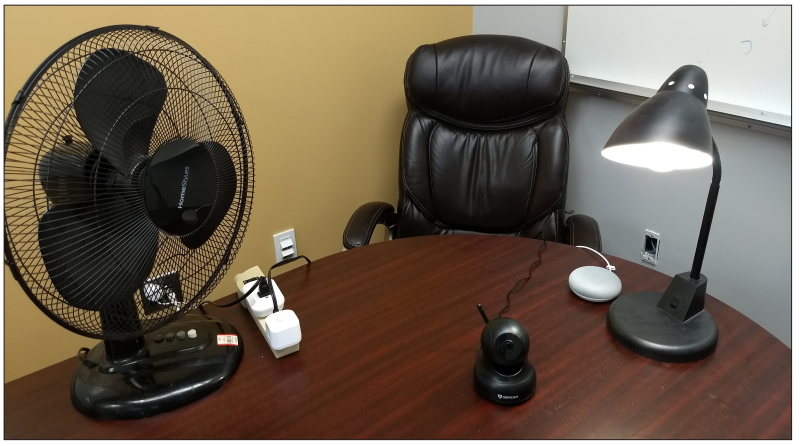

FIGURE 2. Left to right: smart plug, motion sensor, camera, central hub, smart lightbulb.

Central Hub

This device specifically pings the manufacturer's DNS server at 8.8.8.8 twice every 63 s prior to sending the server a DNS query for the A and AAAA records of www.google.com. This results in a deterministic and predictable sequence of 2 WiFi frames with a data payload of 100 B (98-byte ping request) followed immediately by 2 frames with a data payload of 76 B (74-byte DNS query). Additionally, the device sporadically uses multicast DNS to query for other manufacturer devices on the network, sending to a destination MAC of 01:00:5e:00:00:fb (mDNS). These properties can all be determined by analyzing the encrypted 802.11 frames and strongly indicate the presence of this device.

Since this device only has two states, idle or active (being asked or answering a question), it is also possible to determine the state of the device by observing the size of the transmitted frames. In an idle state, the device mostly transmits small TCP maintenance frames of less than 600 B/frame. When a question is asked, the device begins to stream the audio data over TCP, and there is a sustained burst of TCP frames of 1514 B. Once the question is finished, the reverse happens as the audio is streamed back to the device. This can be easily picked up as a burst of WiFi frames of size 1580 transmitted by the device followed by a similar burst going the other way.

When disconnected from the home WiFi network, the device is rendered non-functional as it is a service-based device rather than an appli-

One of the most surprising results was that the motion sensor could not be fingerprinted at all. It had no idle network traffic aside from ARP requests and TCP keep-alives, and only sent data once the motion sensor was triggered. Ironically, in a data-laden world, it is this lack of data that may give it away.

ance-based device. The disabling of the central hub has a minimal impact on the usability of the smart home, because the IoT devices could alternatively be accessed via a smartphone.

Lightbulb

The lightbulb was the easiest to fingerprint and identify as it had an application layer heartbeat to a hub server. Every 54 s, the lightbulb sends a 117-byte frame and receives an 111-byte response (+/- 1 byte). This can be observed over WiFi as two data frames of size 183 and 177 respectively. This hub server is responsible for relaying the control server location to the device, which then establishes a connection and listens for commands.

As a lightbulb, the device fundamentally only has two states, on or off. However, this particular device is capable of dimming the LED as a percentage, thus giving it 101 states: off and 100 variations of on. We observed varying payload sizes of 1082–1090 B with no discernible mapping aside from a rough idea of the level of luminescence, that is, in general, the larger the payload, the brighter the light.

Interestingly, the lightbulb did not have a physical switch to control whether light was being emitted. After performing a deauthentication attack on the device, it was observed that the lightbulb maintained whatever state it had prior to disconnecting without any physical control mechanisms. This has a severe implication as it would be fairly easy for any attacker to externally control the function of the lightbulb.

Motion Sensor

One of the most surprising results was that the motion sensor could not be fingerprinted at all. It had no idle network traffic aside from ARP requests and TCP keep-alives, and only sent data once the motion sensor was triggered. Ironically, in a data-laden world, it is this lack of data that may give it away. While it is not possible to form a generative fingerprint, it is still possible to form a discriminative identity based on other devices on the WiFi network. For example, one can isolate, based on MAC address, the manufacturer's products and from there prune MAC addresses that send WiFi frames other than a 60-second TCP keep-alive and ARP requests (data payloads of 68 and 44 bytes, respectively). However, this method is not guaranteed as we also observed, over a 24-hour period, one NTP synchronization to the manufacturer's servers.

Due to motion sensors having only a triggered state, it is fairly easy to determine whether the device has been activated. The specific motion sensor we tested was trivially easy to determine as it did not send any sizable network traffic until the detection occurred. When testing deauthentications, some detections were buffered until the device managed to reconnect to the WiFi network; however, the parameters of how many and the order of which detections were dropped were not discernible.

Power Switch

This smart plug had a peculiar, extremely unique identifying aspect. By default, this device will broadcast data containing a JSON string over UDP every 3 s. The JSON data appears to contain a device state dump listing its IP, a flag regarding encryption, state, and other miscellaneous data. However, a smartphone running the corresponding application will pick this up and respond to the device; from there, the UDP broadcasts cease, and communication is switched to a TCP connection between the device and receiver. The TCP packets contain the same UDP data, except the JSON portion has been encrypted with non-standard base64 encoding. Despite this strange setup, the device also connects to a Message Queuing Telemetry Transport (MQTT) server to allow it to be controlled externally from outside the same WiFi network, and when idle, pings the server once every 30 s.

It also results in a rather interesting fingerprinting decision tree: if the paired smartphone is not active (e.g., screen off), one would expect to see multiple broadcast frames sent from the device with a payload size of 217 B; otherwise, one would expect to see WiFi frames with a payload size of 84 B sent with a period between 25 and 30 s. Additionally, in both cases, one would expect to see a separate set of frames with a payload size of 58 B/30 s. If the smartphone is on the same network as the smart socket, the command goes from smartphone to device, which sends the result to the remote MQTT servers and back to the smartphone. Otherwise, the command goes from the smartphone to the MQTT servers, relaying the command back to the device.

Regardless, this device does have a physical switch that can control the state of the device, and as a result, deauthentication attacks cannot override the device controls. However, the attacks are effective at disrupting the timezone localized scheduling as it is based on the smartphone application.

Security Camera

Our security camera was also an easy device to identify as it had fairly unique idle network activity with two parts: one loop multicasts three different Simple Service Discovery Protocol (SSDP) messages every 75 s with consistent payloads, while the other is a slightly more complex UDP data stream heartbeat. This UDP stream is established every ~60 s to a remote server, incrementing the device port by 1 every time. The stream is initiated by the device sending a 4-byte payload, beginning a nested loop that sends a 48-byte payload every 44 s until the stream is terminated after 16 iterations. The data being sent was itself unchanging and consequently could be identified easily using its 802.11 frame size.

Furthermore, its activity could be trivially identified by observing any variation in frame sizes in addition to large frames being sent due to the video data. The camera also appears to be using a constant bit rate for transmission as there was no variation in data size unless a rotation order was received by the device.

There are also no physical controls on the device, meaning that like some of the other devices, its correct operation is completely dependent on its ability to connect to a WiFi network. Deauthentication attacks interrupt the video stream, which operates in fire-and-forget mode unless an SD card is inserted into the camera. Consequently, it is possible to render an entire WiFi surveillance network defunct by spamming the airwaves with dissociation frames.

Device Localization

As wireless cameras become more popular for security and surveillance systems, the leaking of their locations poses huge risks for homes and businesses relying on their innocuous operation. Intruders are highly motivated in localizing these devices as

it enables them to make a plan to avoid detection. This is usually called malicious localization. While WiFi localization has been extensively studied in the past, for example, using received signal strength (RSS) or channel state information, we demonstrate that malicious attacks are easy to perform using a passive RSS-based localization scheme. Its novelty lies in clustering analysis and trimmed-mean-based denoising.

In our attack model (Fig. 3), the attacker only needs to carry a mobile device (e.g., smartphone/tablet/laptop) to passively sniff RSS data off of all WiFi packets in the environment by walking around the area. Such an attack is simple to execute and hard to detect due to its stealthiness: the attacker does not need to communicate with the camera, and data collection is easy to carry out by only traveling around the target area. Furthermore, since it only uses RSS, it does not require more complicated channel state information/time of arrival/angle of arrival based methods (e.g., [9–12]), which always involves multiple antennas or specialized hardware that cannot always be discreetly hidden in view. Therefore, due to the simplicity of the above model, we want to examine how much an attacker can do to localize the target.

For the attacker, the first step is to distinguish the wireless camera from other WiFi devices, which can be done via the proposed traffic pattern analysis. Then the attacker can filter packets based on the MAC address and extract RSS data of the target camera. The second step is to collect RSS traces along different walking trajectories and obtain a series of tuples, for example, (attacker_position, camera_RSS, time). The third step is to locate the camera using our unsupervised clustering-based log-distance model. Our scheme is based on the log-distance model because of its simple, widely adopted, and proven robustness. Following the traditional log-distance model, we can estimate the range between the receiver and sender as follows:

$$E_d = E_{d'} - 10\alpha\log_{10}(d/d'),$$

where E_d is the RSS at distance d, $E_{d'}$ is the RSS at reference distance d', and α is a loss factor that indicates how fast the signal decays. Usually, α depends on the environment. In ideal free space, $\alpha = 2$, while for typical outdoor or indoor environments, α may vary between 2.7 and 4.3.

The above traditional log-distance model suffers from many practical issues with WiFi measurements. One of the most important problems is noise, which comes from dynamic environments and hardware imperfection. Therefore, we adapt the unsupervised feature clustering method in [13]. Specifically, we primarily use six features extracted from the raw data. We have collected over 80 points, of which 50 points are for training and 30 points are for testing. *Frame* features denoting traffic statistics of aggregated frames, which is different from [13] that focuses on all kinds of frames. The reason is that we observe the behavior of aggregated frames can more accurately depict the real traffic pattern for wireless cameras. *Spatial* features denoting location features by trajectory analysis. RSS features denoting signal strength. *MSE* (mean-squared error) features denoting the fitting error of the localization model that involves RSS and spatial features. *Environmental* features denoting the smoothing factor α, which is not included in [13]. For the details of clustering process and feature selection, please refer to [13]. Our clustering result is four categories where the cluster center is the coordinate-wise mean of the vectors. The first category takes 59 percent of instances, which have moderate RSS errors and mean square error (MSE). The second category takes 26 percent of instances, which have low RSS errors and high MSE. The third category takes 12 percent of instances, which have the lowest RSS errors and moderate MSE. The fourth category takes 3 percent of instances, which have the largest RSS errors and the lowest MSE. Therefore, we only use data in the first category for further localization computation as the rest of the data is either noise or the outlier for accurate localization. The intuition behind this choice is that the first category can cover most of the instances with reasonable errors.

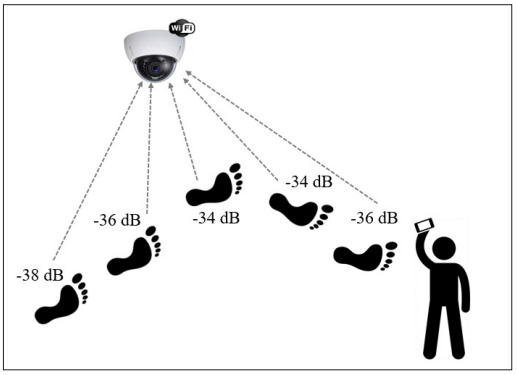

FIGURE 3. Localization model.

After clustering, we further apply a trimmed mean over a time window to increase the concentration of localization results, which are shown in Fig. 4. We quantify the localization accuracy by absolute localization error, which is the Euclidean distance between the estimated position and the true position. From experimental results, we observe that the localization accuracy is consistent across different random walks; for brevity, we only compare two representative samples. For our walking trajectory A, our method improves on localization accuracy from 12 m to 6 m, and a similar trend can be observed for B. In a multi-level building, one can use triangulation to reasonably approximate the floor where the device is. In the future, we would like to extend our localization attack model to other popular IoT devices (e.g., NFC, Bluetooth) and to different application scenarios, such as moving objects.

Attack Strategies

One of the most basic attacks is to use the localization technique covered in the previous section to map out where each IoT device is in the smart home with triangulation. This, combined with a MAC address lookup, grants an adversary working knowledge of roughly where each device is along with potentially what the device is, all by simply taking a lap around the neighborhood. This does not require access to the underlying cleartext and can be done quite easily using a simple promiscuous network controller. Unfortunately, this also means that it is extremely difficult to defend against as long as devices transmit traffic wirelessly. At best, this can be mitigated using a proprietary data-link protocol on a non-standard frequency to strongly discourage any opportunistic adversaries looking for an easy target.

Deauthentication attacks interrupt the video stream, which operates in fire-and-forget mode unless an SD card is inserted into the camera. Consequently, it is possible to render an entire WiFi surveillance network defunct by spamming the airwaves with dissociation frames.

For better or worse, we are beginning to enter an age where the consequences of failing cybersecurity now leads to potential physical harm to the human end-user instead of being isolated to hardware. The importance of this cannot be stressed enough as our observations indicate that some IoT devices lack even a physical control switch to override the embedded network-based controls.

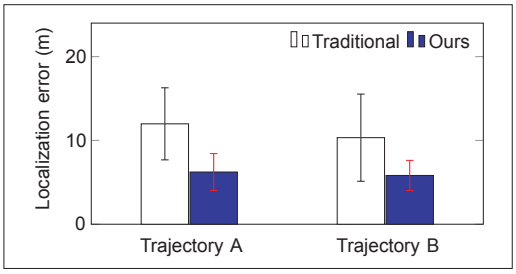

FIGURE 4. Localization performance of the traditional log-distance model and ours.

Once an attacker has obtained this map, the next side-channel vulnerability that can be exploited is the potential localization of the home's occupants (Fig. 5). By simply sniffing and observing the traffic emitted, it is possible for our adversary to discern whether or not a device is active and, based on the traffic pattern, get an idea of what is being sent. As an example, both the central hub and the security camera, when active, pump out an order of magnitude more data than when in an idle or inactive state. Likewise, the motion sensor does not output any data unless the sensor itself is triggered. Furthermore, the amount of data being transmitted along with the rate at which it is being sent/received allows for an easy means to identify the type of device. By identifying these devices and monitoring the traffic, the attacker can determine with reasonable accuracy how many occupants are in the home along with where they are, relative to where each device is.

Additionally, given a sufficiently long period of observation, the attacker could also eventually determine the pose of some devices — specifically those that have some cone of interest. As all security cameras transmit video data and, as a general principle, would like to reduce their data footprint, some manufacturers may see fit to use variable bit rate encoding rather than a constant bit rate. It would be reasonable to assume that for the majority of the time, these cameras will be observing a stationary scene. Thus, it stands to reason to lower the bit rate during these inactive periods and swap to a higher bit rate once a change in the scene is observed. Unfortunately, this creates entropy in the data that gives away two critical pieces of information: whether or not the camera is watching a fixed area (i.e., not sweeping) and the relative pose of the camera should its cone of view overlap with another IoT device's cone of interest. To be more concrete, suppose there was a motion sensor somewhere in the camera's field of view. A person walking through the motion sensor's cone of interest would consequently be walking through the camera's field of view. In terms of data, this would result in an increase in the camera's data rate along with a spike in traffic sent from the motion sensor. Since the attacker is able to triangulate the location of both the camera and the motion sensor, our adversary can slowly piece together the pose of those devices by observing and noting when each device was active. Should the sensor go off without a bit rate increase from the camera, we know that their cones do not completely overlap and can adjust accordingly. This triangulation of cones is greatly increased with each additional overlapping device.

Finally, as an adversary has effectively co-opted the smart home's device network, she can then selectively disable each device as needed remotely (Fig. 6). While it is not necessary to know where the devices are prior to remotely disabling them, it does make it easier to remain undetected. As all of these devices are connected OTA using WiFi, most of them are susceptible to a deauthentication attack — an injection of a deauthentication frame to the device from a rogue AP masquerading as the router. In our analysis, we noted that some devices lack a physical control switch which overrides the network-based switch. These devices are particularly susceptible to this attack as there is nothing the defender can do in this case to regain control of the IoT device.

Discussion and Conclusion

As we have shown, wireless IoT devices are extremely vulnerable to side-channel attacks. These devices unintentionally leak plenty of data that can then be used to reconstruct a layout of the house, as well as allowing an adversary to potentially co-opt the IoT sensor network for their own purposes. With smart devices becoming the norm, the digital and physical world begin to bleed into each other as their interactions become further intertwined, not to mention the involvement of cloud and edge computing [14]. For better or worse, we are beginning to enter an age where the consequences of failing cybersecurity now leads to potential physical harm to the human end user instead of being isolated to hardware. The importance of this cannot be stressed enough as our observations indicate that some IoT devices lack even a physical control switch to override the embedded network-based controls. This implies that once an attacker gains control of the device, short of physically disconnecting the device from its power source, there is nothing that can be done to regain control. In these cases, the device manufacturers must carefully consider what the default failure state should be. This is a key part that is often glossed over as a second thought as many people will assume an assurance of Internet connectivity given how ubiquitous it has become over the past decade.

In conclusion, we have demonstrated that it is possible to mount a side-channel attack on a smart home by exploiting the devices' liberal usage of data and that these devices are being targeted by malicious actors due to their prevalence and vulnerabilities. In what can only be considered an ironic twist of fate, these devices provide no more security than what they take away — much like a castle of glass, with its towering walls and buttresses intimidating and veiling, but offering no obstruction of its inner workings to any passerby. It is time for both device manufacturers and end users to seriously rethink IoT device security before this new technology becomes an ingrained aspect of a modern society. As our society's reliance on technology increases, both the risk and fallout of security breaches increase proportionally to the point where digital attacks may soon have an impact far greater than what conventional attacks can deliver. Consequently, IoT device manufacturers must be held to a higher standard with respect to the out- of-the-box vulnerabilities of their devices.

Acknowledgments

Ryan Shea's research is supported by a Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant. Jiangchuan Liu's research is supported by an Industrial Canada Technology Demonstration Program (TDP) grant and an NSERC Discovery Grant.

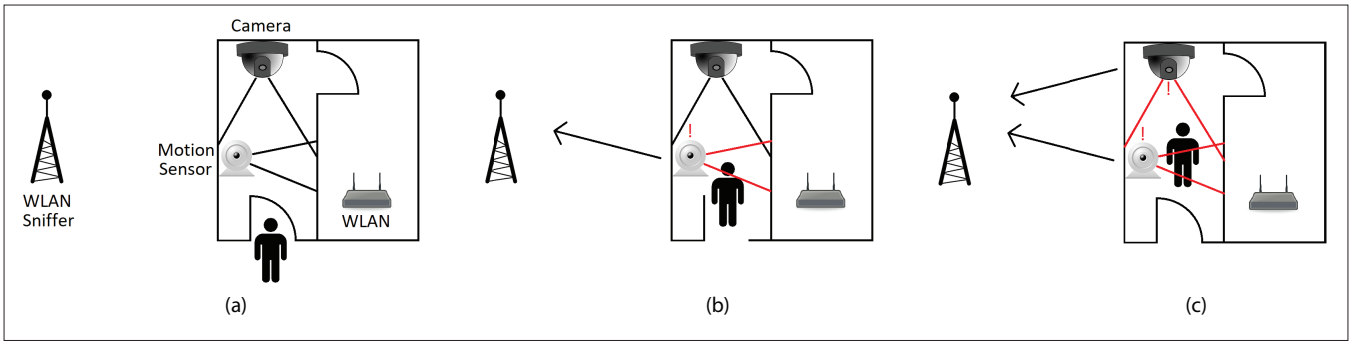

FIGURE 5. Figures demonstrating localization of a person inside a smart home: a) idle traffic; b) motion sensor emits detection; c) security camera emits video.

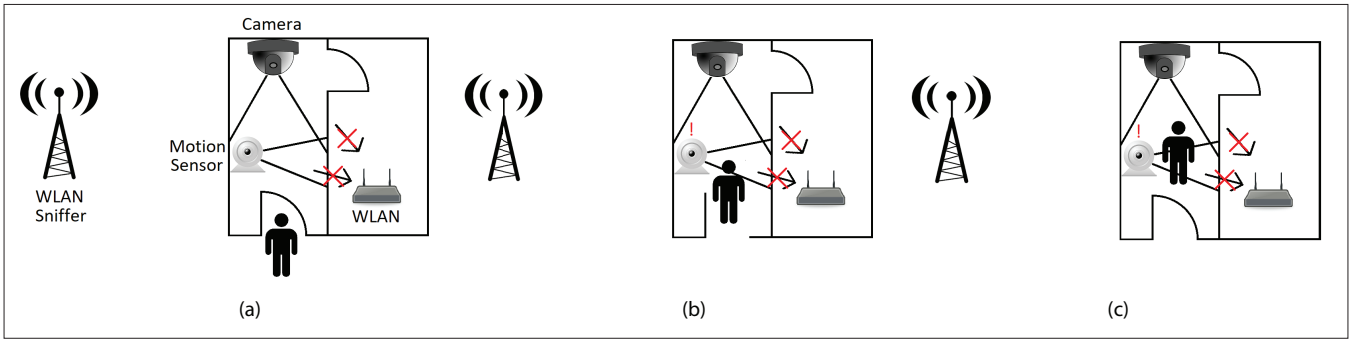

FIGURE 6. Figures demonstrating the effects of a deauthentication attack: a) IoT devices are disconnected; b) sensor detects, but cannot send alert; c) security camera is nullified.

References

[1] A. Nordrum, "Popular Internet of Things Forecast of 50 Billion Devices by 2020 Is Outdated," *IEEE Spectrum*, vol. 18, 2016.

[2] C. Kolias *et al.*, "DDOS in the IoT: Mirai and Other Botnets," *Computer*, vol. 50, no. 7, 2017, pp. 80–84.

[3] M. Antonakakis *et al.*, "Understanding the Mirai Botnet," *Proc. 26th USENIX Security Symp.*, 2017; https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[4] K. Zhao and L. Ge, "A Survey on the Internet of Things Security," *Proc. 2013 9th IEEE Int'l. Conf. Computational Intelligence and Security*, 2013, pp. 663–67.

[5] Y. Cheng *et al.*, "A Lightweight Live Memory Forensic Approach Based on Hardware Virtualization," *Info. Sciences*, vol. 379, 2017, pp. 23–41.

[6] L. Wu, X. Du, and J. Wu, "Effective Defense Schemes for Phishing Attacks on Mobile Computing Platforms," *IEEE Trans. Vehic. Tech.*, vol. 65, no. 8, 2016, pp. 6678–91.

[7] J. Riihijarvi *et al.*, "Providing Network Connectivity for Small Appliances: A Functionally Minimized Embedded Web Server," *IEEE Commun. Mag.*, vol. 39, no. 10, Oct. 2001, pp. 74–79.

[8] IEEE Std. 802.11w, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 4: Protected Management Frames," 2009.

[9] J. Xiong, K. Sundaresan, and K. Jamieson, "ToneTrack: Leveraging Frequency-Agile Radios for Time-Based Indoor Wireless Localization," *Proc. ACM MobiCom*, 2015.

[10] D. Vasisht, S. Kumar, and D. Katabi, "Decimeter-Level Localization with a Single WiFi Access Point," *Proc. USENIX NSDI*, 2016.

[11] M. Kotaru *et al.*, "Spotfi: Decimeter Level Localization Using WiFi," *Proc. ACM SIGCOMM*, 2015.

[12] J. Gjengset *et al.*, "Phaser: Enabling Phased Array Signal Processing on Commodity WiFi Access Points," *Proc. ACM MobiCom*, 2014.

[13] Z. Li *et al.*, "Adversarial Localization Against Wireless Cameras," *Proc. ACM HotMobile*, 2018.

[14] Z. Guan *et al.*, "Achieving Efficient and Secure Data Acquisition for Cloud-Supported Internet of Things in Smart Grid," *IEEE Internet of Things J.*, vol. 4, no. 6, Dec. 2017, pp. 1934–44.

Biographies

Andy Sun received a B.Sc. degree with distinction in 2017 in computer science from Simon Fraser University, Burnaby, British Columbia, Canada. He is currently pursuing an M.Sc. at the same institution. His research interests include mobile computing, augmented reality, and color-based computer vision.

Wei Gong received his B.S. degree from the Department of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China, in 2003, and his M.S. and Ph.D. degrees from the Department of Computer Science and Technology, School of Software, Tsinghua University, Beijing, China, in 2007 and 2012, respectively. His research interests include wireless sensor networks, RFID applications, and mobile computing.

Ryan Shea [S'08, M'16] is an assistant professor in the School of Computing Science at Simon Fraser University. He currently teaches advanced courses on big data systems design for the SFU Professional Master's program. His research interests include virtualization, and performance of big data and cloud computing. While studying as a Ph.D. candidate at Simon Fraser University, he received the prestigious NSERC Alexander Graham Bell Canada Graduate fellowship. He has published over 30 peer-reviewed articles in IEEE and ACM journals, magazines, and conference proceedingss. Recent high profile publications include a point of view article in the *Proceedings of the IEEE* titled "The Future of Cloud Gaming."

Jiangchuan Liu [S'01, M'03, SM'08,F'17] is a professor in the School of Computing Science, Simon Fraser University. He is an NSERC E.W.R. Steacie Memorial Fellow. He is an EMC-Endowed Visiting Chair Professor of Tsinghua University and an adjunct professor of Tsinghua-Berkeley Shenzhen Institute. In the past he worked as an assistant professor at the Chinese University of Hong Kong and as a research fellow at Microsoft Research Asia. He received his B.Eng. degree (cum laude) from Tsinghua University in 1999 and his Ph.D. degree from Hong Kong University of Science and Technology in 2003, both in computer science. He is a co-recipient of the inaugural Test of Time Paper Award of IEEE INFOCOM (2015), ACM SIGMM TOMCCAP Nicolas D. Georganas Best Paper Award (2013), and ACM Multimedia Best Paper Award (2012). His research interests include multimedia systems and networks, cloud computing, social networking, online gaming, big data computing, RFID, and the Internet of Things. He has served on the Editorial Boards of *IEEE/ACM Transactions on Networking*, *IEEE Transactions on Big Data*, *IEEE Transactions on Multimedia*, *IEEE Communications Surveys & Tutorials*, and the *IEEE Internet of Things Journal*. He is a Steering Committee member of *IEEE Transactions on Mobile Computing* and was Steering Committee Chair of IEEE/ACM IWQoS (2015–2017).