Poster: Unveiling IoT Devices Provisioning Process

Rostand A. K. Fezeu, Timothy J. Salo, Amy Zhang, Zhi-Li Zhang
Department of Computer Science & Engineering, University of Minnesota – Twin Cities, U.S.A.
{fezeu001,salox049,zhan7007,zhang089}@umn.edu

ABSTRACT

In this paper, we conduct a first measurement study to examine the provisioning process used by smart home IoT devices. Through reverse-engineering techniques, we capture, decrypt and analyze the message exchanges among IoT devices, the vendor-specific mobile app and vendor cloud services, and investigate their security and privacy implications. Furthermore, we carry out a series of experiments and demonstrate the feasibility of developing an *open, edge-centric* framework for device isolation and device/network/cloud segregation, with an eye towards automatically setting up, managing and securing IoT devices.

ACM Reference Format:
Rostand A. K. Fezeu, Timothy J. Salo, Amy Zhang, Zhi-Li Zhang. 2021. Poster: Unveiling IoT Devices Provisioning Process. In *Internet Measurement Conference (IMC '21), November 2–4, 2021, Virtual.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3446382.3448667

1 INTRODUCTION

Today's smart home IoT devices follow an *opaque*, *closed* "device-to-cloud" stovepipe model, i.e., *cloud-centric* [1]. A user's first interaction with a new smart home IoT device is to "provision" (connect) the device to their Wi-Fi for subsequent device management and control via the vendor-specific mobile app. This provisioning process is largely a *manual* and *one-device-at-a-time* task that is cumbersome, error-prone, and time-consuming. As the number of devices increase to 50 per household by 2025 [2], this process becomes unwieldy. We are also left with lingering concerns: How trustworthy are the devices/vendors? Do they store my Wi-Fi password and other personal or private information in the cloud? Might they offer an entry for hackers into my home network?

IMC '21, November 2–4, 2021, Virtual

ACM ISBN 978-1-4503-8323-3/21/02…$15.00
https://doi.org/10.1145/3446382.3448667

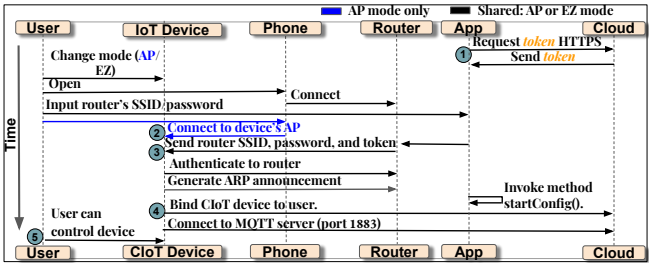

Figure 1: A summary of the data exchanged involved during the EZ and AP Modes device setup process.

To address these concerns, we reverse-engineer smart home IoT device provisioning processes. Our goal is two-fold: i) Gain a deeper understanding of the provisioning steps to examine their security and privacy implications and ii) Explore the feasibility of developing an *open, edge-centric* platform for automatically provisioning smart home IoT devices.

2 IOT PROVISIONING PROCESS

To uncover the provisioning process, we design two toolkits to capture and decrypt messages exchange amongst the mobile apps, the vendor cloud, and the internet for in-depth analysis.

IoT-Dissect I: Our testbed consists of several IoT devices under study, a Wi-Fi network using an OpenWRT access point instrumented with a packet-capture tool (Wireshark), and an Android phone running the vendors' mobile apps. A "man-in-the-middle" (MITM) SSL proxy intercepts traffic between the Wi-Fi network and the outside vendor cloud services.

IoT-Dissect II. For an in-depth analysis, we utilize reverse-engineering techniques: "Code injection" and "function tracing" techniques are used to perform code analysis on each vendor's mobile app (.apk files) and collect information about its operations. For example, function tracing passively monitors selected function calls within the .apk files as the HTTPS requests and responses are encrypted and decrypted. Code-injection is used to extract and log the plaintext `PostData` field of each request before it is encrypted by the mobile apps and each response after it is decrypted by the mobile apps.

2.1 Preliminary Observations

Using IoT-Dissect I, we observed that the provisioning processes of all IoT devices (five devices, including four light bulbs and a smart plug) we examined used the same initial

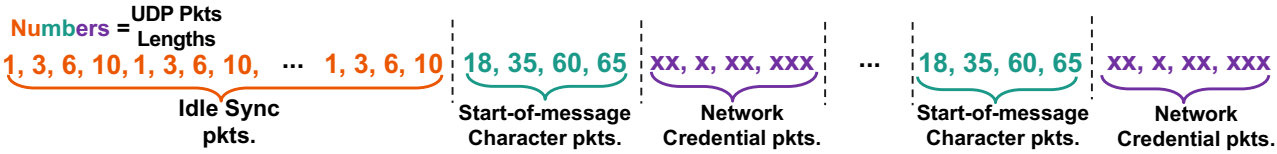

Figure 2: Broadcast UDP packets

sequence, as shown in Fig. 1. (1) The mobile app first connects to the vendor cloud services and request a provisioning `token`. (3) The mobile app gets the user's SSID & password and transmits that information along with the `token` encoded in a sequence of UDP packets broadcasted on port 30011 to the device. (4) The IoT device authenticates to the user's Wi-Fi and then binds itself to the vendor cloud services using the `token`. (5) The user can now control the device using the mobile app *via the vendor cloud services*[1].

Using IoT-Dissect II, we discover how the network credentials (SSID/password/`token`) are encoded and broadcast in a sequence of UDP packets to IoT devices. As shown in Fig. 2, the UDP packets consist of **Idle sync packets**: A sequence of four UDP packets with data lenghts of 1, 3, 6, and 10 bytes to provide "character" sync. **State-of-message character**: Four packets with data lengths of 18, 35, 60, and 65 bytes indicating the start of data content (the credentials). **Network credential packets**: A series of varying lengths UDP packets containing the SSID, password and `token`.

3 BREAKING THE OPAQUE, CLOSED "DEVICE-TO-CLOUD" STOVEPIPE

3.1 IoT Devices and `tokens`

In §2, we observed that the `token` is 32-characters. To further understand the role of this `token` in device-cloud identification and authentication, we conduct a series of experiments with encoded UDP packets containing the network credentials and randomly generated (i.e., arbitrary lengths and 32-characters) `tokens` and a recently received 32-character `token` from the cloud to communicate with the IoT device.

Our experimental results suggest that the IoT devices employ a rudimentary "authentication" process to check the validity of a vendor-generated `token` solely based on the length. The `token` is primarily used by the IoT device to authenticate itself to the vendor cloud services. The vendor-generated `token` is typically valid for about two hours [4]; hence any previously generated `token` older than two hours will be rejected by the vendor cloud services (as a way to defend against replay attacks).

[1], the mobile apps do not *directly* control the IoT device over the home Wi-Fi network. Hence if a user loses Internet connectivity, the user eventually loses control of the home IoT devices, even though the home network is up and running.

3.2 Isolating device-cloud communications

The above experiments illustrate a potential '`token` replay" attack vulnerability. This is partially possible because, IoT devices always listen on UDP port 6668 (MQTT) or UDP port 1883 (secure MQTT). We therefore set out to investigate whether it is possible to segment the "direct" device-cloud channel via a *trusted* intermediary (e.g., our envisaged "open, edge-centric" IoT platform residing at a user's home).

The key challenge is that all HTTPS post requests are signed by the mobile app using *secret keys*. Using IoT-Dissect II, we extract all the keys (including the steganograph-hidden secret key[3]), compute the HMAC_SHA256 hash and successfully sign post requests to the vendor clouds and obtain a `token` used to automate IoT devices provisioning.

4 CONCLUSIONS & FUTURE WORK

We have conducted – to the best of our knowledge – a "first-of-a-kind" in-depth measurement study of smart home IoT device provisioning processes. Our study demonstrates the feasibility of an *open, edge-centric* framework for managing and securing smart IoT devices that does not rely on the vendor cloud i.e., breaking the "device-to-cloud" stovepipe. With such an *open* framework, "management" apps developed by IoT device vendors/third-parties using the framework may be automatically downloaded, validated and verified before deploying on a user's smart home network for automatic provisioning based on user-specified security/privacy intents.

ACKNOWLEDGEMENT

The research was supported in part by NSF under Grants CNS-1618339, CNS-1814322, CNS-1831140, CNS-1836772, and CNS-2106771.

REFERENCES

[1] Udhaya Kumar Dayalan, Rostand AK Fezeu, Nitin Varyani, Timothy J Salo, and Zhi-Li Zhang. 2021. VeerEdge: Towards an Edge-Centric IoT Gateway. In *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. IEEE, 690–695.

[2] Nick G. 2019. How Many IoT Devices Are There In 2020? More Than Ever! (02 2019). https://techjury.net/blog/how-many-iot-devices-are-there/#gref

[3] Neil F Johnson and Sushil Jajodia. 1998. Exploring steganography: Seeing the unseen. *Computer* 31, 2 (1998), 26–34.

[4] Tuya Smart. 2021. Authorization Management-Documentation-Tuya Developer. (03 2021). https://developer.tuya.com/en/docs/iot/open-api/api-reference/authorization/oauth-management?id=K95ztzpoll7v5