Pitfalls of MUD file generation and design flaws of smart home IoT devices

Zizheng Liu, Gaston Ormazabal, Henning Schulzrinne
Department of Computer Science
Columbia University
New York City, United States
Email: zl2694@columbia.edu, {gso, hgs}@cs.columbia.edu

Aman Singh
Palindrome Technologies
Hazlet, United States
Email: aman.singh@palindrometech.com

***Abstract*—Internet of Things (IoT), which connects devices equipped with sensors and actuators to create smart environments, pervades our daily lives. However, due to the heterogeneity of IoT devices, there is neither a standard for designing IoT device network communication patterns nor an approach to analyze them. This paper studies the network traces of five popular smart home consumer IoT devices, two of which are installed in two test beds located in different continents, analyzing the network topology as well as the communication pattern between the device and its end-point servers. Our contributions are two-fold: first, by analyzing IoT devices' network traffic data collected in our lab and another test bed, we found that installed location, software version and the usage of the device are three main factors affecting a device's endpoints. While these three factors are likely to remain unchanged through a capture period performed in a single test bed, the manufacturer usage description (MUD) file generated from the captured data would be too specific to fit devices installed and used in other environments. We also observed patterns from network traffic captured when devices are powered on but not actively used, which contains representative features for IoT device fingerprinting. Second, we present design flaws caused by wrongly made assumptions when designing IoT device's communication system observed in our study. These design flaws have the potential to harm the performance of both network infrastructures such as an NTP server and the IoT device itself.**

***Index Terms*—Internet of Things; Manufacturer Usage Description; Internet Measurement; Cyber Security; Smart Home**

I. Introduction

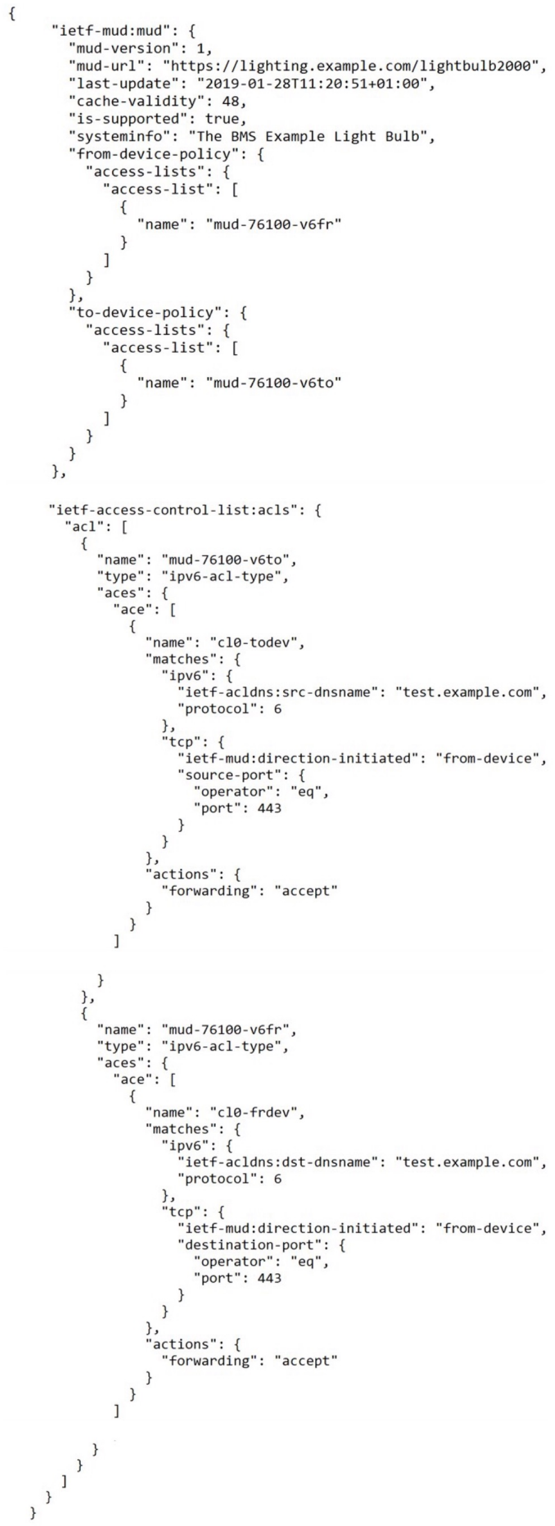
The growing popularity of Internet of Things (IoT) devices introduces security issues to the Internet. Given that hosts in the Internet could easily reach each other [1], IoT devices such as IP enabled home appliances and gadgets also have the capability to communicate with any endpoint host on the Internet. This behavior, uncontrolled, makes smart home IoT devices suitable candidates for distributed denial of service attack (DDoS attack) botnet members, due to their low computational power to protect themselves from malware. In [2], the authors documented the largest and the most well-known DDoS attack launched by IoT botnets to date, Mirai.

The idea of manufacturer usage description (MUD) came up by engineers from Cisco and Google in 2016. To enumerate the endpoints that the device is allowed to connect by its manufacturer, an access control list (ACL) is defined in MUD file as the white-list of end points. Connections to all other end points should be blocked. The ACL in a MUD file could be translated into a set of firewall rules [3] on the local area network (LAN) gateway and any communication to and from unlisted endpoints should be dropped by the gateway. Thus, MUD files help network operators create a firewall for IoT devices in LAN by restricting devices to communicate only with endpoints required for them to function fully. Fig. 1 shows an example of a MUD file. In a MUD architecture, once the gateway of the LAN detects a new IoT device joined the LAN such as by detecting a Dynamic Host Configuration Protocol [4] (DHCP) discover packet from the device, it sends the `MUD URL` in DHCP option 161 field to a MUD server, downloads the MUD file for the device and implements it on the gateway.

In March 2019, MUD became a RFC standard [5]. The standard suggests IoT manufacturers to publish behavioral files, namely MUD files, for their IoT products as they are the ones with the best knowledge of how the devices behave when connected to the Internet. However, manufacturers have not started publishing MUD files for their IoT products and manufacturer-defined MUD files do not exist at the moment to our knowledge. Even after manufacturers start publishing MUD files for their IoT products, there are still billions of IoT devices in people's homes without manufacturer-defined MUD files' protection. Thus, creating MUD files for IoT devices by analyzing their network traffic becomes an important field of Internet security.

In this paper, we report and list factors that might be ignored when generating MUD files using network measurement approaches, which would results in generating inaccurate MUD files. We also analyzed that, with the rapid increase of IoT device's quantity and emergence of new IoT device models, how is the stability of the Internet infrastructure such as NTP servers being influenced.

After the introduction in Section I, we briefly present the background of our work in Section II. In Section III, we report our method used for data collection and analysis, followed by the observations we made and how they influence MUD file generation in Section IV. We point out some design flaws of smart home IoT devices we found in our test bed in Section V. Related work and conclusions are Section VI and Section



```
{
    "ietf-mud:mud": {
      "mud-version": 1,
      "mud-url": "https://lighting.example.com/lightbulb2000",
      "last-update": "2019-01-28T11:20:51+01:00",
      "cache-validity": 48,
      "is-supported": true,
      "systeminfo": "The BMS Example Light Bulb",
      "from-device-policy": {
        "access-lists": {
          "access-list": [
            {
              "name": "mud-76100-v6fr"
            }
          ]
        }
      },
      "to-device-policy": {
        "access-lists": {
          "access-list": [
            {
              "name": "mud-76100-v6to"
            }
          ]
        }
      }
    },

    "ietf-access-control-list:acls": {
      "acl": [
        {
          "name": "mud-76100-v6to",
          "type": "ipv6-acl-type",
          "aces": {
            "ace": [
              {
                "name": "cl0-todev",
                "matches": {
                  "ipv6": {
                    "ietf-acldns:src-dnsname": "test.example.com",
                    "protocol": 6
                  },
                  "tcp": {
                    "ietf-mud:direction-initiated": "from-device",
                    "source-port": {
                      "operator": "eq",
                      "port": 443
                    }
                  }
                },
                "actions": {
                  "forwarding": "accept"
                }
              }
            ]

          }
        },
        {
          "name": "mud-76100-v6fr",
          "type": "ipv6-acl-type",
          "aces": {
            "ace": [
              {
                "name": "cl0-frdev",
                "matches": {
                  "ipv6": {
                    "ietf-acldns:dst-dnsname": "test.example.com",
                    "protocol": 6
                  },
                  "tcp": {
                    "ietf-mud:direction-initiated": "from-device",
                    "destination-port": {
                      "operator": "eq",
                      "port": 443
                    }
                  }
                },
                "actions": {
                  "forwarding": "accept"
                }
              }
            ]

          }
        }
      ]
    }
  }
```

Fig. 1. An example of MUD file [5]

VII, respectively.

II. Background

According to [6], 51% of American families are willing to pay in excess $500 for a smart home. For this research, we spent around $500 to built a lightweight IoT test bed to mimic the smart home environment of an average family. We chose devices among the most popular IoT categories and most of the devices we studied could be found in the first result page when searching its category name on *amazon.com*. Since there is no standard for IoT device-category mapping, we classify IoT devices into categories based on their most salient feature. For example, although a Triby speaker has Internet call functionality, we classify it into multimedia category because its most salient function is playing Internet radios and Spotify music streaming. We try to maximize the coverage of IoT vendors when selecting devices. This is based on the assumptions that 1) different vendors might have business relationships with different cloud service providers and 2) engineers from different vendors design their products differently. Despite that Triby speaker is not a best-selling device, it was studied previously by a research team from Australia [7] and the network traffic data for that study is publicly available. Thus, we bought the same model and installed it in our lab in the United States in order to do an apple-to-apple comparison. A common attribute shared by all devices we studied is that every IoT device needs a companion application from its vendor installed on a smart phone for Wi-Fi connection and bootstrapping. The applications also exist as interfaces for users to control the devices from the smart phone. We refer to such applications as 'companion applications'. We briefly introduce the device we selected and studied in this section to provide readers a better understanding of background for our discussions in following sections.

LIFX light-bulb is a smart light-bulb that could be turned on and off immediately or by schedule from its companion application. Light color and brightness of the light-bulb could also be changed from the companion application. We found that it communicates with only one end point during our experiment. **iHome** is another device with only one end-point. It senses real-time temperature, humidity, sunlight, sound and motion of the environment. The data collected is then sent to the end-point using a HTTP POST message and could be observed at any time from its companion application by HTTP GET from client side (companion application) to the remote server (the device's only end-point). **Kasa Spot camera** is a device with more end-points due to its more complex functionality. The camera is activated only when 'privacy mode' is set to be off. When the camera is activated, the user could choose to watch live video streaming captured by the camera and 'talk' through the microphone embedded in the camera by sending the voice data from companion application to the device via cloud. Even when the user is not watching the video streaming, the camera actively detect the presence of motion as long as it is activated. The camera records a video clip for tens of seconds after a motion is detected and sends it to the cloud. The user would receive a notification once a motion is detected and the short video clip could be replayed on the companion application after downloaded from the cloud. A **Triby speaker** is an IoT device that has Internet radio and Spotify music streaming as its entertainment functions and Internet call and text message as its communication functions. It is the only IoT device we studied that connects to content provider servers (Spotify and Internet radio sites). The last IoT device we studied in our test bed is a semi-Internet-connected device, namely **Voco Link outlet**. The device monitors power consumption data such as active power of the specific electric appliance connected to it. Although it is an IP-enabled device, the power consumption information is available and the outlet could be turned on and off only when the smart phone with the companion application and the outlet are connected to the same LAN. Otherwise, it is identical to a traditional outlet.

III. Data collection and analysis

In this section we report our test bed setting, the method and tools we used for data collection and analysis.

A. Locally collected data

As mentioned in Section II, devices from the same vendor share a companion application installed on a smart phone. We use an Apple iPhone X as the host of all vendors' companion applications in the experiment. To install the devices and connect them to the Internet, we first connect the smart phone to the gateway and download the companion applications of the devices. All devices could be connected to the Internet by receiving the gateway's information (Wi-Fi name and password) shared by their companion applications. To capture the network data sent from IoT devices, we run `tcpdump` [8] on `eth1` of our gateway, a DD-Wrt [9] powered Netgear R7000 router. To prevent the network traffic data from overflowing the router, we `mount` a 29 Gigabytes USB to the router for data storage. We separated our data collection time into two periods: activated period and idle period. For activated period data, we captured two hours of network traffic from 21:00 - 23:00 on July 26th, 2019. In this period, we manually executed all the functions of every IoT devices in our test bed and kept a log of all activities as the ground truth. The granularity of the activity is one minute. For example, if we start an Internet call from Triby application on smart phone to the Triby speaker at 21:01:45, and end the call at 21:03:02, we add an entry '21:01 - 21:03 Internet call from phone to speaker' to the activity log of the Triby speaker. We wait at least one minute to make another activation to prevent the intersection of activities, which means the soonest start time of next activity of Triby speaker is 21:05 in above example. We interact with IoT devices in two ways. The first is physical interaction such as making a movement in front of the Kasa Spot camera to let it detect the motion and send the video stream. The second is remote interaction such as changing the light color of the LIFX smart light-bulb from its companion application. We remotely interacted with IoT devices in both cases that when the smart phone is in the same LAN with the device and when the smart phone is not in the same LAN with the device. The only exception, Voco Linc plug, could not be activated by its companion application when the smart phone and the plug are not in the same LAN. We report and discuss this case in Section IV. By interacting with every IoT device, we believe that we have executed all the functions, captured corresponding network traffic data and created complete activity logs for every device. For idle period data, we captured the network trace of IoT devices from July 26th 23:50, 2019 to July 29th 14:10, 2019. Devices are neither physically nor remotely activated in this period, which means the features extracted from the captured traffic data are irrelevant to IoT device's usage. Both activated period data and idle period data are stored on the mounted USB in pcap file format.

B. Data form UNSW

Besides our locally collected data, we also obtained publicly available data captured by a research team from University of New South Wales [7]. From their web site, we downloaded the network trace data of 28 IoT devices which were monitored from September 22nd, 2016 to October 11th, 2016. The data is stored as 20 pcap files. Each pcap file contains the daily network trace of all 28 devices captured by `tcpdump` on the gateway of their test bed. They also provide device-MAC address mapping information on their website. Thus we could filter the traffic of the specific device by MAC address. Since we only care about the IoT device that both test beds have in common to study how device behaviour changes in different regions and probably with different software versions (we believe there are software version updates took place between 2016 and 2019 and we verified it in Section IV), we filtered the network trace of the Triby speaker and LIFX light bulb from their daily data sets by implementing MAC address filter on Wireshark [10]. Unfortunately, although the IoT devices are frequently activated in the monitoring period, there was no activity log like ours kept during the study. In spite of this, we could still guess the activities of these two devices based on knowledge of devices installed in our test bed and the similarity between the data sets.

C. Data analysis

We downloaded the data captured from our gateway and used Wireshark and Python's *scapy* [11] library for data analysis. We first drew a line chart of number of packets generated in each minute for every device and compare the line chart against the activity log of the device. We found that each interaction could be identified from a 'hump' in the line chart. This reveals that either a physical interaction or a remote interaction could cause an increase of the volume of data transmitted to one or some of the device's end points. We then analyzed the endpoints of IoT devices from different perspectives: 1) we analyzed the geographic information of the endpoints using the Maxmind [12] geolocation database. Although IP addresses to city mapping provided by such databases are well known to be inaccurate, the mappings on country level are reliable sources for IP address geolocation [13]–[15]. 2) we analyzed the owners of endpoints. According to [16], an endpoint of an IoT device could be a server from its vendor, a server from a supporting organization which provides cloud services to IoT vendors, or a server from an advertising company which is responsible for pushing customized advertisements to users of the device. We inferred the organization of the owners based on DNS reverse-resolutions and common knowledge about IoT vendors and cloud service providers. For example, a DNS reverse-resolution of LIFX lightbulb endpoint shows that the host name of the server is 'v2.broker.lifx.co' and the IP address belongs to Amazon AWS. Thus, we could infer that this server acts as a broker between the smart lightbulb and its companion application and LIFX is using cloud services from Amazon AWS. In fact, we found that four among five devices are renting cloud services provided by Amazon AWS.

IV. Observations

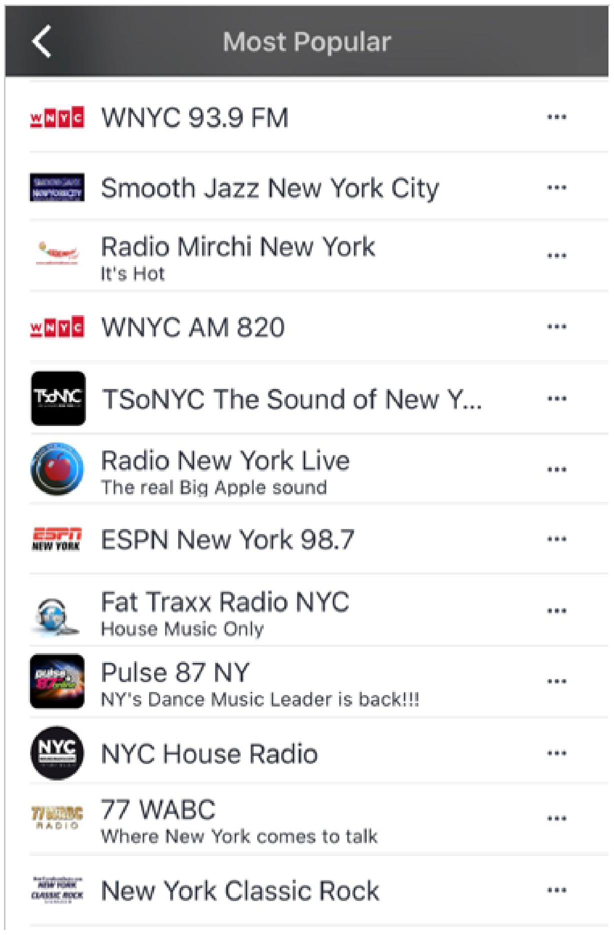
In this section, we report our observations on the network behaviors of IoT devices in our test bed. These observations characterize insights into how network behavior changes in different scenarios such as when devices are installed in different regions, whether or not the smart phone with companion application is in the same LAN with the device and when devices are studied in different time periods. All these differences would vary the endpoint hosts of the devices which further varies the devices' MUD files.

A. Lightweight devices vs. content retrieval devices

In contrast to general purpose devices that connect to unlimited number of hosts in the Internet, IoT devices only talk to a finite set of endpoints. Based on our study of IoT endpoints, we divide these endpoints into two sets. The first set contains endpoint hosts that serve as functional servers for IoT devices, it could be a web-server that holds device's configuration files or a SIP server for setting up and terminating Internet calls between devices. We call endpoints in this set functional endpoints. The other set contains endpoint hosts that provide media content to IoT devices; these endpoints are usually third-party content provider servers. An example from this set could be a server from Spotify that provides music streaming to devices. We call endpoints in this set content endpoints. Based on the endpoint type, we could also define IoT devices as different kinds. In our experiment, four out of five devices only connect to functional endpoints. Due to the limited number of functions each device has, all of these devices connect to fewer than 10 endpoints. Thus, we define these devices as *lightweight* devices. The only outlier, Triby speaker, connects not only functional endpoints but also content endpoints for its Spotify music streaming function and Internet radio function. With the Internet radio function, users could listen to hundreds of channels provided by different Internet radio sites by choosing from a channel list suggested in the companion application. Although such a list increases the number of endpoints by orders of magnitude compared to *lightweight* devices, we believe the length of the list is still finite. The evidence is the observation that Triby speaker connected to *airable.io*, a radio provider for hardware platforms. From the introduction of the *airable internet radio station catalogue* on *airable.api*'s web page [17], we learned that it is an API that support Internet radio for hardware and maintains a long but finite list of content provider domain names. We define devices like Triby speaker as *content retrieval* devices that typically have a larger number of endpoints compare to lightweight devices. Besides Triby speaker, a content retrieval device could be a smart hub with the function of 'reading' news from different media sites for users. A point worth noting is that both the Internet radio channel list and the media sites list are depend on the geographic region, which means the entries in the list might differ to serve local users better when devices are installed in different locations. We will discuss this in more detail in next subsection. In terms of MUD file generation, retrieving the catalogue of the content provider API and inserting its endpoint domain names into the ACL is an essential step to avoid blocking content provider endpoints by mistake. The list of endpoints should be extracted from the catalogue rather than from measurement approach because it is hard to exhaustively connect to all such endpoints in data capture period.

B. Multimedia IoT devices are more region dependent

When analyzing device endpoints' geographical information, we found that content retrieval IoT devices rely more on their installed location: Triby speaker, in its set up phase, sent a request to *akamai.net* for the external IP address of the gateway which reveals its location. By continuing to monitor its network traffic, we found its Internet radio functionality is location dependent. Fig.2 shows a screen shot of the 'popular radios' list shown in the companion application during our experiments. Indicated by channel names, most of channels are New York related. In contrast, while we study the network traffic generated by the same device installed in UNSW's test bed, although we do not have the counterpart of popular radio list from their test bed, we found DNS queries requested for *icecast.sbs.com.au* and the DNS server responded with IP addresses of servers located in Sydney. The explanation for this is that content retrieval IoT devices' functions are mostly content based, for the same device model, users from different regions are interested in different regional related contents, which makes device's endpoints rely heavily on device's installation location. From MUD aspect, a local version of MUD file is necessary for these region dependent devices. A MUD file for a content retrieval device installed in New York City would be significantly different from the one for the same device installed in Sydney. Additionally, a general MUD file which contains the union of endpoints from local MUDs' ACL would contain too many redundant endpoints to limit a device communication pattern which is opposed to MUD's original intention. Endpoints extracted from the catalogue of content providers should be the only difference between regional MUD files.


Most Popular

WNYC 93.9 FM

Smooth Jazz New York City

Radio Mirchi New York
It's Hot

WNYC AM 820

TSoNYC The Sound of New Y...

Radio New York Live
The real Big Apple sound

ESPN New York 98.7

Fat Traxx Radio NYC
House Music Only

Pulse 87 NY
NY's Dance Music Leader is back!!!

NYC House Radio

77 WABC
Where New York comes to talk

New York Classic Rock

Fig. 2. Part of the channel list of the Triby speaker installed in our test bed

C. Network behavior changes when smart phone is or not in LAN

For some IoT devices, their network behaviors vary depending on whether the smart phone with the companion application is in the same LAN with the device. Kasa spot camera is an example device that behaves differently in two scenarios. The device actively detects motions and passively waits for user's commands. When the smart phone is in the same LAN, the control messages such as watch video, stop video and talk through microphone, and video streaming data, exchange directly between camera and smart phone inside the LAN without travelling through the Internet. In this case data capturing tools installed on PC that listens on egress interface of router monitors IoT traffic by ARP spoofing are not able to capture traffic sent directly from IoT to the phone. However, when the smart phone is not in the same LAN, the control messages are sent from the smart phone to the cloud before being forwarded to the camera. Conversely, the video streaming data are sent from camera to the cloud before being forwarded to the smart phone. This difference should be noticed and access control entries of both local endpoints and external endpoints should be captured to generate a correct MUD file.

D. Changes of IoT devices' endpoints over time

We compared the data collected from our test bed against the data from UNSW. For the two devices we have in common, namely LIFX light-bulb and Triby speaker, we found both of their network characters has changed over time. Three years ago, we found that LIFX light-bulb used NTP protocol for synchronization after DNS requesting for *ntp.pool.org* every 300 seconds. However, the NTP pool project [18] does not suggest programmers to request first layer domain name *ntp.pool.org* for NTP services. As a consequence, DNS responses to the light bulb contained IP addresses of NTP servers distributed globally. We found 594 different IP addresses mapping to servers around the world from DNS responses of *ntp.pool.org*. This is probably because that first layer NTP servers (ntp.pool.org) wanted to redirect NTP clients globally for load balancing. However, the responses from NTP servers far from the light bulb might suffer longer delay and reduce the accuracy for synchronization. pool.ntp.org suggests the users to sign up for a vendor specific domain name such as pool.vendor_name.ntp.org and the DNS would resolve the domain names to IP address of NTP servers near the requesting device. Different from three years ago, from the data collected in our test bed, we observed that LIFX light-bulb connects to only one endpoint server: 'v2.broker.lifx.co'. As its name suggests, this is a 'broker' server that handles the commands sent from companion application and forward them to the light-bulb. Since we found that the light-bulb and the broker server is communicating with HTTP protocol, we indicate that the light-bulb synchronizes itself with the server using the timestamp in HTTP headers. Instead of unwisely requesting time data from NTP servers around the world, synchronizing from HTTP timestamps is a better design because the light bulb is not required to have its clock be accurate in the order of millisecond for its scheduled turn on/off functionality. A proper way of requesting NTP service is given by Triby speaker's vendor - Invoxia. From the UNSW data, we observed the speaker used *pool.invoxia.ntp.org* to request NTP services. DNS responded the domain name with IP addresses of NTP servers located in New South Wales to minimize the delay for synchronization. Besides synchronization method, another type of difference we captured is the change of business relationship between IoT vendors and cloud service providers. For example, from UNSW's data, we found an endpoint with domain name *sip.invoxia.com* and it is resolved to IP addresses of SIP servers that are owned by OVH SAS and located in France. But from the data generated in our test bed, we captured *sip.aws.invoxia.io* as the domain name of one of Triby speaker's endpoints. The same differences are found for Web-socket severs. Thus, we conclude that Triby speaker used to use SIP server (for Internet call) and Web-Socket server (for doodle message) from OVH SAS three years ago but they are renting these services from Amazon AWS now.

E. Traffic features irrelevant to device usage

When analyzing the traffic data of idle period, we observed that each device has its own traffic character. Since the devices are not actively used in the idle period, the characters are irrelevant to device usage and could be used for device identification. For example, different devices use different port numbers from dynamic ports range (port numbers greater than 49151), combining with transport layer protocol, the port number and protocol information could be used for device identification. We also found different devices connect to functional servers such as DNS servers and NTP servers at different frequencies. For example, Kasa spot camera connects to NTP server at 0:00 a.m. daily while Triby speaker synchronizes itself via NTP every 12 hours. Those different patterns are another feature for device identification. Besides, domain names from DNS requests queried by the device are also representative signatures for device fingerprinting. A notable point is that all features we introduce above could be extracted from headers rather than inspecting the traffic's payload.

V. Design flaws

A. Constant IP

In our study, we found that the Triby speaker connects to an end-point whose IP address is not resolved via DNS. We ruled out the possibility that we missed the DNS packets resolve to this IP address because we captured all traffic starting at the device installation phase. We listened to two channels with the Internet radio, namely '011.FM Pure Piano' and 'Smooth Jazz 24/7'. Before the speaker connected to '011.FM Pure Piano', it sent a DNS query with *listen.011fm.com*. However, the speaker directly connected to 54.38.43.201 for 'Smooth Jazz 24/7' without DNS resolution. Although we tend to attribute this to the fault of the Internet radio API used by the device (Section IV.A), we see the risk that hard coded IP addresses would be ignored when generating MUD file because the endpoints are represented by domain names rather than IP addresses in MUD files. It might cause the device unable to connect to specific servers when such an incorrect MUD file is implemented in LAN. The hard coded IP address becomes useless and even could be a security issue if the IP address of that host is changed.

B. Periodic NTP queries

As mentioned in Section IV.E, devices query NTP servers with different patterns. We summarize devices into three different groups based on their query patterns. Devices in the first group synchronize themselves at a fixed time, for example at 0:00 a.m. daily. Devices in the second group query for time after every fixed time interval (e.g., every minute or five minutes). Devices in the third group connect to NTP servers after a randomized time interval within a range (e.g., 1 minutes to 2 minutes). In our opinion, the pattern of the third group is the best and the pattern of the first group is the worst. All devices in a time zone following the first pattern would connect to NTP servers at the same time. The traffic burst might reduce the performance of NTP servers and increase the delay between NTP servers and devices. The second pattern, might also cause NTP servers bombarded by devices that are synchronized with each other. Additionally, the authors of [19] pointed out that 'strictly periodic sampling carries the danger of phase locking with periodic network events'. Thus, the pattern followed by devices in the third group, which is referred to as 'noisily periodic' by them, is preferred.

C. Single point of failure

During the installation and set up phase of the Triby speaker, it connected to *developer.invoxia.com* to download configuration files for initialization. Then it connected to this host daily to check for updates. In both of the data from UNSW and our data, we found only one server, located in France, map to the address resolved by this domain name. From the domain name, we indicate that the server is used for developers to develop the system of devices. Since the number of developer is relatively small comparing with the number of products, one server is enough for developing and testing. However, when the prototype became products and sold to all around the world, this server should no longer be used by the devices because a single point of failure might cause all products around the world unable to be set up and updated. The servers work for products should be distributed rather than being single and unchanged in three years.

VI. Related Work

Similar to our work that learns IoT devices from network traffic aspects, [20] dissect the security risks of three household IoT devices and highlight the ease with which security and privacy could be compromised. [16] studied the information exposure from tens of smart home IoT in terms of traffic destination, whether the traffic is encrypted and what activity caused the traffic.

[21], [22] and [23] present IoT security frameworks that could be implemented on the gateways of LANs. [24] introduces a distributed IoT intrusion detection system (IDS) that takes advantage of generative adversarial network (GAN). IoT device with the system installed could monitor itself as well as neighbour IoT devices to detect internal and external attacks. [25] develops a open-source tool that collects IoT traffic by ARP spoofing. The data are gathered in a crowd-sourcing manner under users' agreement. The data could be used for data-driven smart home research such as [7], [26], [27] and [28] that use machine learning approach for IoT device identification or to differentiate IoT devices from general purpose devices.

[29] provide methods for IoT vendors on MUD file generation and how to validate the quality of a MUD file. [30] discusses how MUD files could be implemented on SDN networks to detect volumetric attacks.

VII. Conclusion

By collecting and analyzing data generated from our test bed which is mimicking a normal smart home environment and comparing it with the data captured from a test bed in another location three years ago, we documents the pitfalls that might be ignored when generating MUD files in one test bed by measurement approaches. Our observation presents guidelines to be followed when generating MUD files. We also reveal design flaws of smart home IoT device from network aspect. Our work provides a closer viewpoint of IoT device traffic for IoT researchers and bad design patterns that should be avoided by IoT device designers to maintain the security and stability of the Internet.

References

[1] Bush, Randy, Olaf Maennel, Matthew Roughan, and Steve Uhlig. ”Internet optometry: assessing the broken glasses in internet reachability.” The 9th ACM SIGCOMM conference on Internet measurement, pp. 242-253. November 2009, Chicago, IL, USA

[2] Kolias, Constantinos, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. ”DDoS in the IoT: Mirai and other botnets.” Computer Volume: 50, Issue: 7, pp. 80-84, 2017.

[3] Russell, Rusty, and Harald Welte. ”Linux netfilter hacking howto.” [Online] Available: https://www.netfilter.org/documentation/HOWTO/netfilter-hacking-HOWTO.txt

[4] Alexander, Steve, and Ralph Droms. ”DHCP Options and BOOTP Vendor Extensions” RFC 2132, March 1997. [Online] Available: https://tools.ietf.org/html/rfc2132

[5] Lear, E., R. Droms, and D. Romascanu. ”Manufacturer Usage Description Specification,” RFC 8520, March 2019. [Online] Available: https://tools.ietf.org/html/rfc8520

[6] Icontrol Networks. ”2014 State of the Smart Home - Icontrol Networks.” 2014. [Online] Available: http://www.icontrol.com/insights/2014-state-smart-home/

[7] Sivanathan, Arunan, Hassan Habibi Gharakheili, Franco Loi, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. ”Classifying IoT devices in smart environments using network traffic characteristics.” IEEE Transactions on Mobile Computing, Volume: 18, Issue: 8, pp. 1745-1759. August 2018.

[8] The Tcpdump Group. ”TCPdump and Libpcap projects” 2010. [Online] Available: https://www.tcpdump.org/

[9] S. Gottschall, ”DD-WRT project,” July 2008. [Online] Available: http://www.dd-wrt.com

[10] Combs, Gerald. ”Wireshark-network protocol analyzer.” 2008. [Online] Available: https://www.wireshark.org/

[11] Biondi, Philippe, ”Python scapy program” 2008. [Online] Available: https://scapy.net/

[12] Maxmind GeoLite2 Database. 2002. [Online] Available: https://dev.maxmind.com/geoip/geoip2/geolite2/

[13] Gueye, Bamba, Artur Ziviani, Mark Crovella, and Serge Fdida. ”Constraint-based geolocation of internet hosts.” IEEE/ACM Transactions on Networking, Volume 14, Issue 6, pp. 1219-1232, June 2006.

[14] Huffaker, Bradley, and Marina Fomenkov. ”Geocompare: a comparison of public and commercial geolocation databases-Technical Report.” Technical Report, Cooperative Association for Internet Data Analysis (CAIDA), May 2011.

[15] Poese, Ingmar, Steve Uhlig, Mohamed Ali Kaafar, Benoit Donnet, and Bamba Gueye. ”IP geolocation databases: Unreliable?.” ACM SIGCOMM Computer Communication Review Volume 41 Issue 2, pp. 53-56. April 2011.

[16] Ren, Jingjing, Daniel J. Dubois, David Choffnes, Anna Maria Mandalari, Roman Kolcun, and Hamed Haddadi. ”Information exposure from consumer iot devices: A multidimensional, network-informed measurement approach.” ACM The Internet Measurement Conference, pp. 267-279. October 2019, Amsterdam, Netherlands.

[17] airable.api: airable internet radio station catalogue [Online] Available: https://www.airablenow.com/airable/airable-api/

[18] pool.ntp.org: public NTP time server for everyone. 2003. [Online] Available: https://www.ntp.pool.org/en/

[19] Baccelli, Francois, Sridhar Machiraju, Darryl Veitch, and Jean C. Bolot. ”The role of PASTA in network measurement.” ACM SIGCOMM Computer Communication Review, Volume 36 Issue 4, pp. 231-242. September 2006, Pisa, Italy.

[20] Notra, Sukhvir, Muhammad Siddiqi, Hassan Habibi Gharakheili, Vijay Sivaraman, and Roksana Boreli. ”An experimental study of security and privacy risks with emerging household appliances.” 2014 IEEE Conference on Communications and Network Security, pp. 79-84. October 2014, San Francisco, CA, USA.

[21] Midi, Daniele, Antonino Rullo, Anand Mudgerikar, and Elisa Bertino. ”KalisA system for knowledge-driven adaptable intrusion detection for the Internet of Things.” IEEE 37th International Conference on Distributed Computing Systems (ICDCS), pp. 656-666. June 2017, Atlanta, GA, USA.

[22] Pacheco, Jesus, and Salim Hariri. ”IoT security framework for smart cyber infrastructures.” IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS* W), pp. 242-247. September 2016, Augsburg, Germany.

[23] Singh, Aman, Shashank Murali, Lalka Rieger, Ruoyu Li, Stefan Hommes, Radu State, Gaston Ormazabal, and Henning Schulzrinne. ”HANZO: Collaborative Network Defense for Connected Things.” Principles, Systems and Applications of IP Telecommunications (IPTComm). October 2018, Chicago, IL, USA

[24] Ferdowsi, Aidin, and Walid Saad. ”Generative Adversarial Networks for Distributed Intrusion Detection in the Internet of Things.” arXiv preprint arXiv:1906.00567 (2019).

[25] Huang, Danny Yuxing, Noah Apthorpe, Gunes Acar, Frank Li, and Nick Feamster. ”IoT Inspector: Crowdsourcing Labeled Network Traffic from Smart Home Devices at Scale.” arXiv preprint arXiv:1909.09848 (2019).

[26] Meidan, Yair, Michael Bohadana, Asaf Shabtai, Martin Ochoa, Nils Ole Tippenhauer, Juan Davis Guarnizo, and Yuval Elovici. ”Detection of unauthorized IoT devices using machine learning techniques.” arXiv preprint arXiv:1709.04647 (2017).

[27] Bremler-Barr, Anat, Haim Levy, and Zohar Yakhini. ”IoT or NoT: Identifying IoT Devices in a Short Time Scale.” arXiv preprint arXiv:1910.05647 (2019).

[28] Miettinen, Markus, Samuel Marchal, Ibbad Hafeez, N. Asokan, Ahmad-Reza Sadeghi, and Sasu Tarkoma. ”IoT Sentinel: Automated device-type identification for security enforcement in IoT.” IEEE 37th International Conference on Distributed Computing Systems (ICDCS), pp. 2177-2184. June 2017, Atlanta, GA, USA.

[29] Hamza, Ayyoob, Dinesha Ranathunga, Hassan Habibi Gharakheili, Matthew Roughan, and Vijay Sivaraman. ”Clear as MUD: generating, validating and applying IoT behavioral profiles.” ACM SIGCOMM 2018 Workshop on IoT Security and Privacy, pp. 8-14. August 2018, Budapest.

[30] Hamza, Ayyoob, Hassan Habibi Gharakheili, Theophilus A. Benson, and Vijay Sivaraman. ”Detecting Volumetric Attacks on loT Devices via SDN-Based Monitoring of MUD Activity.” ACM 2019 Symposium on SDN Research, pp. 36-48. April 2019, San Jose, CA, USA.