DFRWS 2018 IoT Forensic Challenge

Submission by SPAWAR Systems Center ATLANTIC

Team: Mark Gramajo, Joshua Lewis, Randall Sharo, Shawn Zwach

Submission Date: 21 Feb 2018

Revision Date: 27 Apr 2018

Version 1.2

Table of Contents

Table of Figures

Submission Contents

Our submission consists of the following elements:

Table 1- Contents of this submission

Element	Path	Description	Prerequisites
FINDINGS	./SSCLANT_FINDINGS.pdf	Summarizes the team's findings	-
Google OnHub Diagnostic Parser	./onhub_parsing	Extracts command output and static files from onhub diagnostic	Python 2.7 Google Protobuf Python Bindings (>=v3.5.1)
Docker scripts	./docker	Scripts to automate elasticsearch, kibana, and plaso interactions	Ubuntu 16.04, Docker 17.05, Elasticsearch 5.2.1, Kibana 5.2.1
Plaso IoT patch	./docker/plaso.p	Adds Google Hangouts, Kodi, and Smart Things parsers for Plaso	Plaso 20171231 (image from Docker Hub)

In the event of a Grand Prize monetary award, the check should be written to US Treasury.

Executive Summary

Through the DFRWS 2018 Forensic Challenge allotted period of time, Team SPAWAR Systems Center Atlantic (SSCA) performed a digital forensic examination of the following devices:

/001-SmartTV-RaspberryPi/E001SmartTVMMC/

SHA1: 9ac0de76eca7958bfed1bd5909bbf766409af180

/002-BettyNote2Black/SHV-E250L_Physical_20170717/

SHA1: cd494cf3097d8482100ce26dc8e35f0d87b67198

/003-SimonNote2White/SHV-E250S_Physical_20170718/

SHA1: fc28e415ee740531df86a2b227c4f514e9ed40ba

/004-Onhub-diagnostic-report/

SHA1: 20eb4825eaf6c303beadd090868110fb2de37066

/005-Amazon-Ech-Alexa-Web-Scrape/

SHA1: d1d126f47b565926dcc80fe6a4e7094f0281cb47

/006-SmartHome-Cloud-Provider-AcmeInc-NetworkDump.tcpdump

SHA1: 6ab6c522b070cde292a18645a19929998e009293

In conjunction with the IoT Forensic Challenge Case Details and forensic images, Team SSCA has concluded with a moderate level of confidence that the murder was perpetrated by **John Macron**, Betty Hallym's co-worker and associate, on 07/17/2017 at 15:13 GMT+9 and ending with Betty taking a final 41 steps and her movement ceasing at 15:15 GMT+9.

Findings Summary

We find that the most likely murder suspect is **John Macron**, Betty Hallym's co-worker and associate.

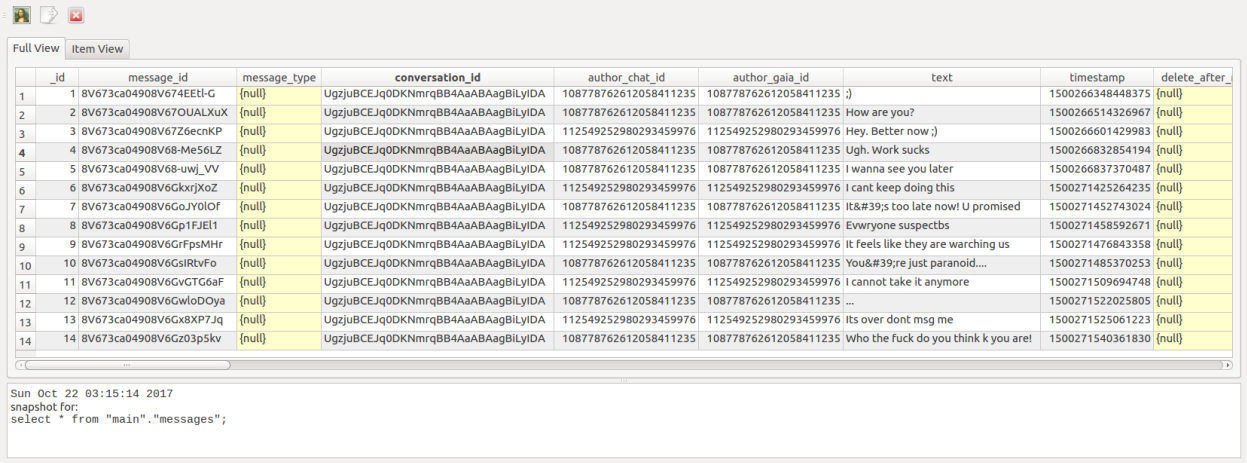
The timeline of significant events leading up to the murder on July 17, 2017 are as follows:

Table 2 - Timeline of Significant Events

Time (GMT+9)	Event	Source	File
13:39:08	John contacts Betty on Google Hangouts	Betty's Phone: Google Hangouts	/data/data/com.google.android.talk/databases/babel1.db
13:47:17	John asks to see Betty later	Betty's Phone: Google Hangouts	/data/data/com.google.android.talk/databases/babel1.db
15:03:45	Betty attempts to break off contact	Betty's Phone: Google Hangouts	/data/data/com.google.android.talk/databases/babel1.db
15:05:40	John issues hostile reply, Betty blocks his account	Betty's Phone: Google Hangouts	/data/data/com.google.android.talk/databases/babel1.db
15:06:06	Betty turns on Pandora	Alexa webscrape	9.json, 9.wav
15:07:30	Simon finishes "Jackie Park" video	SmartTV: Kodi MyVideos Database	/home/osmc/.kodi/userdata/Database/MyVideos107.db; YouTube Video ID: ibOskbTPZYE
15:12:13	Actuator ("deadbolt") engaged via smartthings	Cloud capture	006-SmartHome-Cloud-Provider-AcmeInc-NetworkDump.tcpdump, starting packet **20687**
15:12:39	Betty and John argue in-person	Alexa webscrape	8.json, 8.wav
15:12:58	Alexa hears Betty's voice: "Alexa"	Alexa webscrape	6.json, 6.wav
15:13:12	Alexa hears "stop"	Alexa webscrape	5.json
15:13	Betty takes her	Betty's Phone:	/data/data/com.xiaomi.hm.health/databases/origin

to 15:14	final 41 steps	MiFit	_db_ 8ddbe561538e754225d70a5a122dd1f5 data/com.xiaomi.hm.health/files/mili_log.txt
15:15	Betty's movement ceases	Betty's Phone: MiFit	/data/data/com.xiaomi.hm.health/databases/origin _db_ 8ddbe561538e754225d70a5a122dd1f5 data/com.xiaomi.hm.health/files/mili_log.txt
15:19:22	Smart outlet power command via smartthings ("tv off")	Cloud tcpdump	006-SmartHome-Cloud-Provider-AcmeInc-NetworkDump.tcpdump, starting packet **33045**
15:19:37	Simon stops watching "Top 15 Korean Drama" video	SmartTV: root filesystem	/home/osmc/.kodi/userdata/Database/MyVideos107.db, YouTube VideoID: VKfbVLmkQUs
15:20:07	Simon turns off tv	Alexa webscrape	3.json, 3.wav
15:20:33	Simon says "Alexa, call the ambulance!"	Alexa webscrape	2.json, 2.wav, 1.json, 1.wav
15:31	Police called	Case Details	IoT Forensic Challenge Case Details.pdf
15:34	Irregular movement observed on MiFit	Betty's Phone: MiFit	/data/data/com.xiaomi.hm.health/databases/origin _db_ 8ddbe561538e754225d70a5a122dd1f5 data/com.xiaomi.hm.health/files/mili_log.txt
15:40	Police arrive, secure scene	Case Details	IoT Forensic Challenge Case Details.pdf

Betty's Google Hangouts database identifies John Macron as a chat participant. John and Betty have a conversation that proceeds as follows:


Full View | Item View

	_id	message_id	message_type	conversation_id	author_chat_id	author_gaia_id	text	timestamp	delete_after_
1	1	8V673ca04908V674EEtl-G	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	;)	1500266348448375	{null}
2	2	8V673ca04908V67OUALXuX	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	How are you?	1500266514326967	{null}
3	3	8V673ca04908V67Z6ecnKP	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	Hey. Better now ;)	1500266601429983	{null}
4	4	8V673ca04908V68-Me56LZ	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	Ugh. Work sucks	1500266832854194	{null}
5	5	8V673ca04908V68-uwj_VV	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	I wanna see you later	1500266837370487	{null}
6	6	8V673ca04908V6GkxrjXoZ	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	I cant keep doing this	1500271425264235	{null}
7	7	8V673ca04908V6GoJY0lOf	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	It's too late now! U promised	1500271452743024	{null}
8	8	8V673ca04908V6Gp1FJEl1	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	Evwryone suspectbs	1500271458592671	{null}
9	9	8V673ca04908V6GrFpsMHr	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	It feels like they are warching us	1500271476843358	{null}
10	10	8V673ca04908V6GsIRtvFo	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	You're just paranoid....	1500271485370253	{null}
11	11	8V673ca04908V6GvGTG6aF	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	I cannot take it anymore	1500271509694748	{null}
12	12	8V673ca04908V6GwloDOya	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	...	1500271522025805	{null}
13	13	8V673ca04908V6Gx8XP7Jq	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	Its over dont msg me	1500271525061223	{null}
14	14	8V673ca04908V6Gz03p5kv	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	Who the fuck do you think k you are!	1500271540361830	{null}

```
Sun Oct 22 03:15:14 2017
snapshot for:
select * from "main"."messages";
```

Figure 1 - Google Hangouts argument: Betty Hallym and John Macron

Betty does not appear to reply to the final message from John Macron. Based on the fact that John Macron is on the `blocked_people` database table it can only assumed that she blocked his account after receiving the final message.

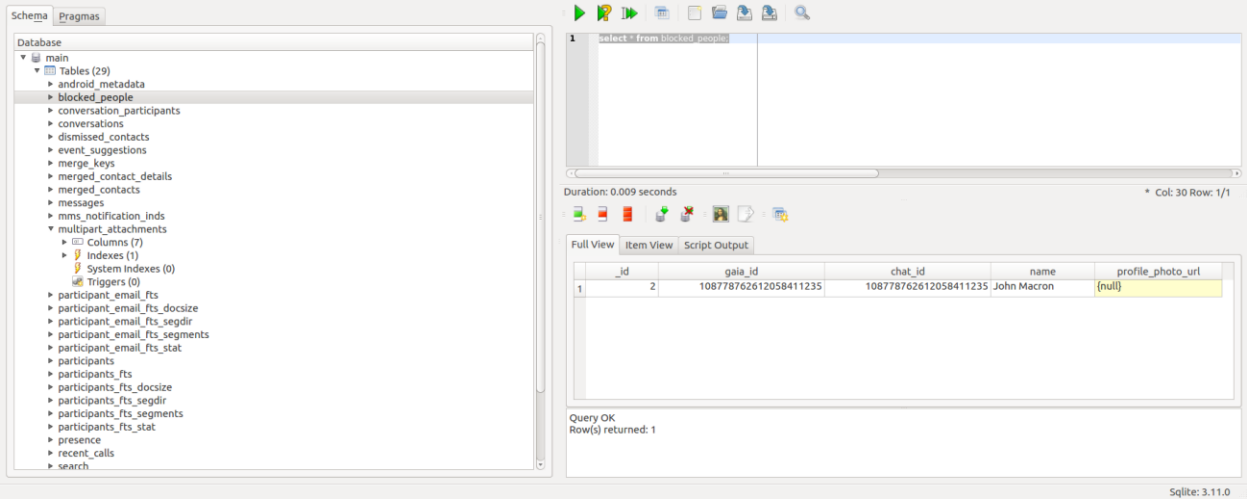


Figure 2 - Betty blocks John Macron's messages

The majority of the audio captures from the Alexa web-scrape are in the voices of Simon (ex: 11.wav, “Alexa, turn on TV”) and Betty (ex: 10.wav, “Alexa, turn on Pandora”).

The audio in 8.wav and 7.wav is distinctly different: two different speakers are captured (15:12 GMT+9). One voice is consistent with other captures of Betty, and it says “Alexa….. stop” in both captures. The second voice sounds like a male, and is unlike any capture of Simon’s voice. The male voice says first says “How could you do this? What are you thinking?”, and then “I can’t believe you would do this to me. We said we would. What are you thinking?” These statements are consistent with the Google Hangouts argument between Betty and John Macron less than ten minutes earlier.

Notably, the SmartThings packet capture identifies a command to an actuator (IPSO Alliance Object ID 3201) right before the in-person argument captured by Alexa. It would appear that Betty let John into the home by disengaging the deadbolt. There are two minutes between Betty’s final voice capture by Alexa (15:13:12), and her final movement registered in the MiFit (15:15).

Simon’s final interaction with the Smart TV occurs at 15:20:07. A closing door can be heard in the audio capture at this time, indicating Simon’s exit from the bedroom. This gives John Macron a full four minutes to depart the apartment before Simon discovers the body.

Tool Build & Operating Instructions

Our tools are delivered as python source code, and scripts to build a set of Docker images.

The general flow of data is as follows:

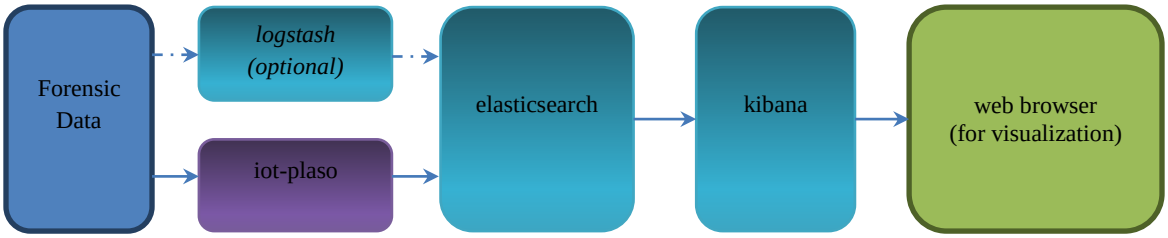


Figure 3 - Tool Data Flow

Prerequisites

The tools are intended to be run from an **Ubuntu 16.04** host with **SIFT-CLI** installed as described here:

https://github.com/sans-dfir/sift-cli#installation

The user must have **sudo** access to install packages and run docker containers. The build scripts provided will invoke sudo as needed; the user must be prepared to enter their password. It must also have support for **docker-compose** and the protobuf bindings for python.

The host must also have internet or other network access that allows it to download images from Docker Hub and Elastic (https://hub.docker.com and https://www.docker.elastic.co).

Building the Docker Images

From the top-level of the delivery package, type:

```
cd docker
make images build
```

Figure 4 - Commands to build docker-hosted tools

This will download the required Docker Hub images and apply a patch to the standard log2timeline/plaso image. The system should now have five docker images installed:

Table 3- Docker images after tool build

Docker Image	Description
Iot-plaso:latest	Plaso with IoT plugins added
log2timeline/plaso:latest	Base image for iot-plaso
docker.elastic.co/kibana/kibana:5.2.1	Visualizer for Elasticsearch
docker.elastic.co/logstash/logstash:5.2.1	Data ingest for Elasticsearch
docker.elastic.co/elasticsearch/elasticsearch:5.2.1	Search and analytics engine

Building Python Protobuf Bindings

We also developed a tool to parse the OnHub parser diagnostic. These are the instructions to build that parser.

Download and install protobuf >= v3.5.1.

```
wget  \ https://github.com/google/protobuf/releases/download/v3.5.1/protobuf-all-3.5.1.tar.gz
tar zxf protobuf-all-3.5.1.tar.gz
cd protobuf-3.5.1
./configure
make
sudo make install
cd python
python setup.py build
sudo python setup.py install
```

Figure 5 - Commands to build Google OnHub protobuf bindings

Launching Tools

Elasticsearch and Kibana

Elasticsearch and Kibana are launched using **docker-compose**. The file **docker/Makefile** can automatically launch these tools via the commands:

```
# The following setting is reset each time your machine boots
# You can set it in your /etc/sysctl.conf file to make it permanent
sudo sysctl -w vm.max_map_count=262144

cd docker; make elastic-start
```

Figure 6 - Command to launch Elasticstack in Docker

The provided **docker-compose.yml** script will bind Elasticsearch to port 9100 and Kibana to port 5601 on the host machine. Kibana will be reachable by navigating your web browser to http://localhost:5601

Note: *You may need to wait a while (~60 seconds) after launching before Kibana will accept browser connections.*

The default login and password for Elasticsearch and Kibana will be:

Login: elastic

Password: changeme

The containers can be shut down and deleted via the commands:

```
make elastic-stop ; make clean
```

Figure 7- Commands to terminate and delete Elasticstack containers

Ingested data will remain in the docker storage volume **esdata1** until you purge it with the command:

```
make realclean
```
Figure 8 - Command to delete Elasticsearch storage volume(s)

iot-plaso

The iot-plaso image can perform many different functions, depending on command-line arguments provided. Typical invocations will be as follows:

Load plaso database into elasticsearch:

```
# Set the IP address of your Elasticsearch server.
# You can use the IP address of the elasticsearch container launched via
# "make elastic-start", the IP address of your host workstation,
# or the IP address of your Docker network switch (normally 172.17.0.1)
# You cannot use 127.0.0.1 since that address points to the plaso container itself
export ELASTIC_IP=172.17.0.1

# Set your desired Elasticsearch index name.  You must configure Kibana to process
# this index name if you wish to see ingested events
export INDEX_NAME=dfrws2018

sudo docker run --rm -ti -v /cases:/cases iot-plaso:latest psort -o elastic --
raw_fields --index_name ${INDEX_NAME} --server ${ELASTIC_IP} --elastic_user elastic
/cases/dfrws2018/events.plaso
```
Figure 9 - Command to load Plaso database into Elasticsearch

Run IoT-Plaso interactively:

```
sudo docker run --rm -ti -v /cases:/cases -v /mnt:/mnt --entrypoint /bin/bash iot-
plaso:latest
```
Figure 10 - Command to interactively launch iot-plaso container with case files

onhub_dump.py

To extract command output and files from a OnHub Diagnostic Report, run the following command where "004-Onhub-diagnostic-report" should be replaced by the name of your report.

```
python onhub_dump.py 004-Onhub-diagnostic-report
```
Figure 11 - Command to process the OnHub diagnostic report

Detailed Findings

001-SmartTV-RaspberryPi

Based on the forensics image and the IoT Forensic Challenge Case Details.pdf this appears to be a Raspberry Pi that is being used as a SmartTV device. The image that was provided was one physical image of the system's entire filesystem. While processing the image, we noticed that the timezone for the system was set to GMT-5 (America/New York), which differs from the other forensics artifacts.

```
$ cat /etc/timezone
America/New_York
```
Figure 12 - Observing that SmartTV has incorrect timezone setting

After exploring the system's file structure and forensics timeline data, it was determined that the device had Kodi installed. Kodi is a free open-source media player that is used for streaming media such as videos, music, and podcasts from the Internet.

Based on a document on Github (https://gist.github.com/pretorh/276bd351b0e2283a93b7) and the Kodi reference documentation (https://kodi.wiki/view/Databases) it was determined that Kodi stores a list of the videos watched and the time/date that the videos were watched. The file where this was stored was `/home/osmc/.kodi/userdata/Database/MyVideos107.db`. The file type of the file is a sqlite3 database. Using the information, we built a parser for the Digital Forensics tool `log2timeline` to parse the database and display events in Kibana

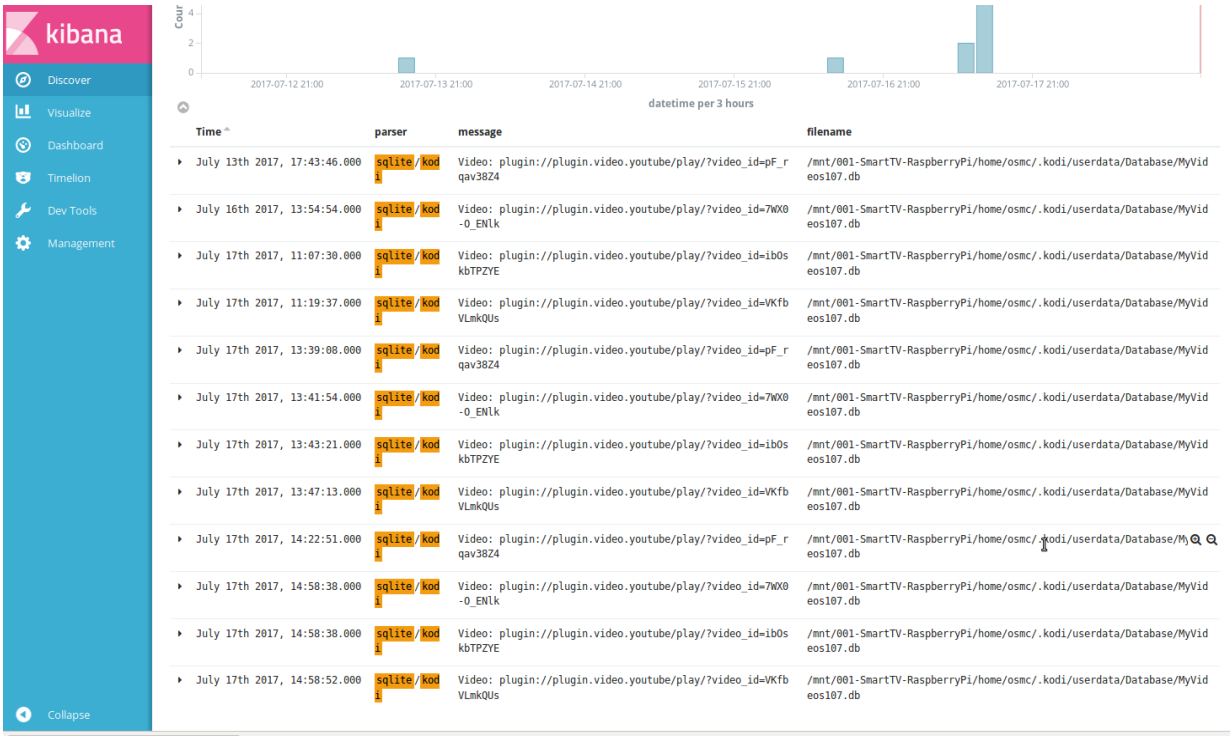


Figure 13 – SmartTV Viewing events as captured by our plugin and viewed in Kibana

Analyzing the Kodi database we can see that several Youtube Videos were played between July 13 and July 17 at 14:58:52

We note that within the `\E001SmartTVMMC\Partition2\home\osmc\.kodi\userdata\Thumbnails\c\` folder the below image (`cf092F21.jpg`) is a screen capture from YouTube Video ID: `ibOskbTPZYE`. The thumbnail was created on 7/17/2017 at 15:03:32 GMT+9.

Figure 14 - cf092f21.jpg thumbshot from Simon's last viewing session

(from E001SmartTVMMC\Partition2\home\osmc\.kodi\userdata\Thumbnails\c\)

002-BettyNote2Black

The zip file that was provided had 1 folder which had 7 raw partition images from a Samsung Note2 device. The partitions were CACHE, EFS, HIDDEN, RADIO, SYSTEM, TOMBSTONES, and USERDATA. After exploring the system's file structure and forensics timeline data that there were two databases on the phone which held significant forensics data. These databases were sqlite3 databases in the USERDATA partition. Additionally, it was discovered that the Xiaomi MiFit device had a log with relevant data.

Google Hangouts

Betty's phone had useful information in its Google Hangouts app database, located in her USERDATA partition at `/data/com.google.android.talk/databases/babel1.db`.

We've included a plaso sqlite3 plugin that parses this database to record message times and contents.

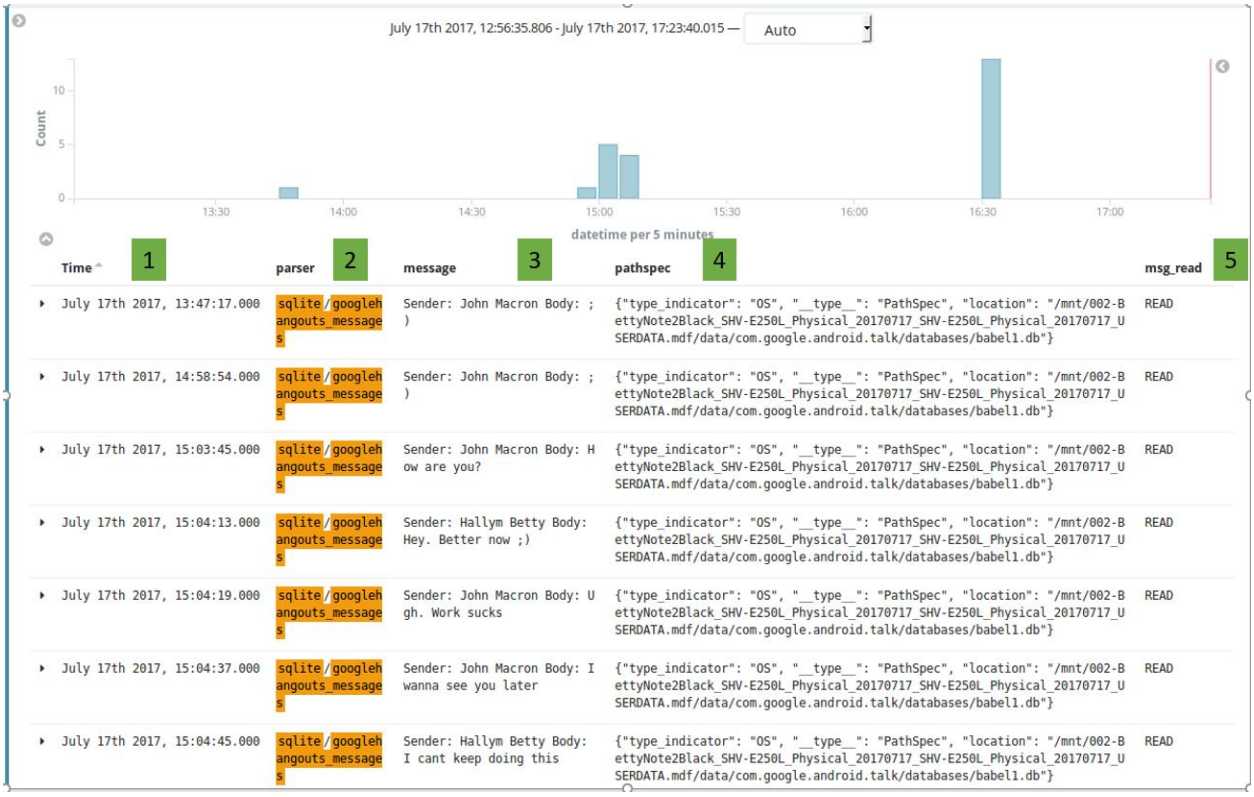


Figure 15 - Google Hangouts events as captured by our plugin and viewed in Kibana

In the above Figure, we have focused in on the Google Hangouts conversations retrieved from Betty's phone and demonstrated the flexibility of using Plaso Log2timeline with Elasticsearch and Kibana. You can also use Kibana to show other details from plaso such as the sha256 hash value of the artifact being parsed.

1 - Shows the date/time of the event

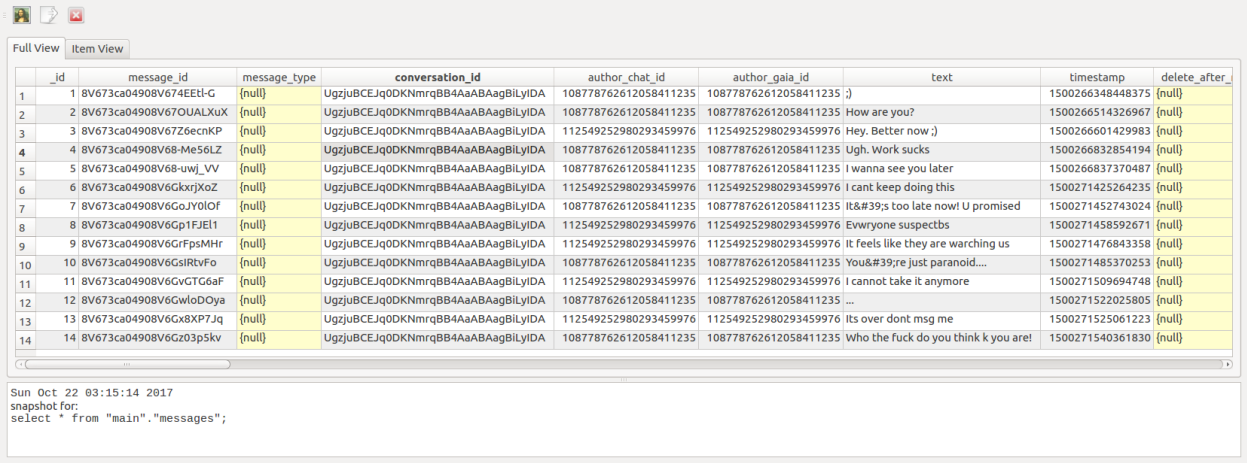
2 - Shows the parser that was used to retrieve and parse the database from the forensics evidence.

3 - Displays the Sender and the Message Body.

4 - Shows the pathspec which indicates the source file that was used to pull the evidence. In this case we can see that it was retrieved from `/mnt/002-BettyNote2Black_SHV-E250L_Physical_20170717_SHV-E250L_Physical_20170717_USERDATA.mdf/data/com.google.android.talk/databases/babel1.db`

5 - Shows the status of the Message and indicates whether or not the message was read.

The complete conversation is shown as a sqliteman screen shot below:



Full View | Item View

	_id	message_id	message_type	conversation_id	author_chat_id	author_gaia_id	text	timestamp	delete_after_
1	1	8V673ca04908V674EEtl-G	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	;)	1500266348448375	{null}
2	2	8V673ca04908V67OUALXuX	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	How are you?	1500266514326967	{null}
3	3	8V673ca04908V67Z6ecnKP	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	Hey. Better now ;)	1500266601429983	{null}
4	4	8V673ca04908V68-Me56LZ	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	Ugh. Work sucks	1500266832854194	{null}
5	5	8V673ca04908V68-uwj_VV	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	I wanna see you later	1500266837370487	{null}
6	6	8V673ca04908V6GkxrjXoZ	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	I cant keep doing this	1500271425264235	{null}
7	7	8V673ca04908V6GoJY0lOf	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	It's too late now! U promised	1500271452743024	{null}
8	8	8V673ca04908V6Gp1FJEl1	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	Evwryone suspectbs	1500271458592671	{null}
9	9	8V673ca04908V6GrFpsMHr	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	It feels like they are warching us	1500271476843358	{null}
10	10	8V673ca04908V6GsIRtvFo	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	You're just paranoid....	1500271485370253	{null}
11	11	8V673ca04908V6GvGTG6aF	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	I cannot take it anymore	1500271509694748	{null}
12	12	8V673ca04908V6GwloDOya	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	...	1500271522025805	{null}
13	13	8V673ca04908V6Gx8XP7Jq	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	112549252980293459976	112549252980293459976	Its over dont msg me	1500271525061223	{null}
14	14	8V673ca04908V6Gz03p5kv	{null}	UgzjuBCEJq0DKNmrqBB4AaABAagBiLyIDA	108778762612058411235	108778762612058411235	Who the fuck do you think k you are!	1500271540361830	{null}

```
Sun Oct 22 03:15:14 2017
snapshot for:
select * from "main"."messages";
```

Figure 16 - Sqliteman view of Google Hangouts conversation

We can see in this capture that Betty's conversation with author *108778762612058411235* (John Macron) got heated near the end. The timestamps are UNIX epoch time with milliseconds, like most other databases on android phones. These timestamps place the conversation on the day of the murder, ending at 15:05:40 GMT+9.

We can also see from the same database that Betty banned John Macron. This table also correlates his gaia_id with his name:

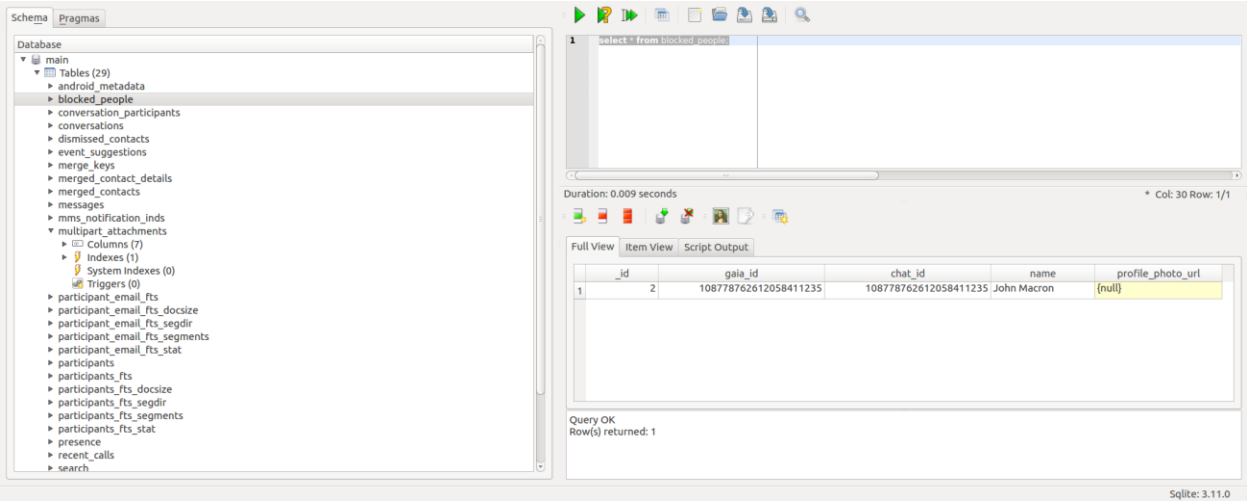


Figure 17 - Sqliteman view of John Macron blocked on Betty's phone

This argument continues in person, and can be heard in the Amazon Echo voice captures less than ten minutes later. In the argument, Betty attempts to break off contact with John, and John becomes irate. The conversation is suggestive of an affair, and termination of this affair would give John a motive for murder.

MiFit

Analysis of Betty's phone showed the presence of the *com.xiaomi.hm.health* (https://play.google.com/store/apps/details?id=com.xiaomi.hm.health) application, as evidenced by files located in her USERDATA partition at */data/com.xiaomi.hm.health*.

Two particular artifacts were found to have useful data:

Table 4 - Significant MiFit artifacts from Betty's phone

MiFit Userdata File (Betty's Phone)	Description
/data/com.xiaomi.hm.health/databases/origin_db_8ddbe561538e754225d70a5a122dd1f5	sqlite3 Application Database. Contains device, user, and step information.
/data/com.xiaomi.hm.health/files/mili_log.txt	Diagnostic application log. Contains detailed description of device/phone interactions. Contains detailed step information, some redundant with Application Database.

Further research into this application and compatible product lines revealed a few reverse-engineering projects:

Table 5 - References for MiFit protocol and function

Project Site	Description
http://allmydroids.blogspot.hu/2014/12/xiaomi-mi-band-ble-protocol-reverse.html	Describes Mi Band Bluetooth protocol, links to several libraries and apps.
http://changy-.github.io/articles/xiao-mi-band-protocol-analyze.html	Further describes Mi Band protocols and message fields. Describes meanings of field data.

Armed with this information, we started with mili_log.txt and found the daily summary for the day of the murder:

```
2017-07-17 16:19:11.238  StepTimeLineFragment  update :
       //======================||
       ||     day step: 3242 steps  ||
       || day distance: 2.2 km  ||
       ||  day calorie: 56 Cal  ||
       ||======================||
 >>>============================<<<
 active item:
       time: 10:43 AM ~ 10:47 AM Slow walking
duration: 0:4
       step: 133 steps
distance: 82 m
 calorie: 2 Cal

 >>>============================<<<
 active item:
       time: 10:51 AM ~ 10:58 AM Fast walking
duration: 0:7
       step: 801 steps
distance: 593 m
 calorie: 14 Cal

 >>>============================<<<
 active item:
```

```
       time: 11:10 AM ~ 11:20 AM Slow walking
duration: 0:10
       step: 530 steps
distance: 359 m
 calorie: 8 Cal

 >>>==========================<<<
 active item:
       time: 11:52 AM ~ 11:56 AM Slow walking
duration: 0:4
       step: 102 steps
distance: 63 m
 calorie: 1 Cal

 >>>==========================<<<
 active item:
       time: 01:09 PM ~ 01:14 PM Slow walking
duration: 0:5
       step: 257 steps
distance: 160 m
 calorie: 3 Cal

 >>>==========================<<<
 active item:
       time: 01:51 PM ~ 01:57 PM Slow walking
duration: 0:6
       step: 216 steps
distance: 134 m
 calorie: 3 Cal

 >>>==========================<<<
 active item:
       time: 02:12 PM ~ 02:19 PM Slow walking
duration: 0:7
       step: 241 steps
distance: 150 m
 calorie: 3 Cal
```

Figure 18 – MiFi Summary of Betty's activity on the day of the murder

This was helpful in determining we were on the right track, identified Sally's activity at a course level, but did not provide any information on the critical time window from 15:13 (last voice activity) to 15:40 (police arrive on scene).

Fortunately, the logs also displayed detailed step information reported each time the band synchronized with the phone:

```
2017-07-17 16:17:15.240  MyActivityDataSyncCallback  data size:65,data time:Mon Jul
17 15:12:03 GMT+09:00 2017,phone time:Mon Jul 17 16:17:15 GMT+09:00 2017,isEnd:false
2017-07-17 16:17:15.271  MyActivityDataSyncCallback
============source:MILI_1A============
============ActivityData============
912:ActivityData [intensity=30, steps=0, category=0]
<912:0:30:0:254>
913:ActivityData [intensity=88, steps=41, category=1]
<913:1:88:41:254>
914:ActivityData [intensity=76, steps=0, category=0]
<914:0:76:0:254>
915:ActivityData [intensity=35, steps=0, category=0]
<915:0:35:0:254>
```

```
916:ActivityData [intensity=0, steps=0, category=3]
<916:3:0:0:254>
917:ActivityData [intensity=0, steps=0, category=3]
<917:3:0:0:254>
918:ActivityData [intensity=0, steps=0, category=3]
<918:3:0:0:254>
...
```

Figure 19 - Detailed capture of Betty's activity, starting 15:12:03

First thing to note is that the reports are in local time (GMT+9). The time of the first activity report is given by "**data time:Mon Jul 17 15:12:03**". Each **ActivityData** line reports accelerometer movement intensity, number of steps, and category of movement (0="same as last report", 1="slow walk", 17="run", etc.) for a 1-minute interval.

The final ActivityData report with nonzero step information is **913:ActivityData [intensity=88, steps=41, category=1]**, which reports 41 steps during the time interval **15:13:03-15:14:03**. Intensity measurements drop to zero over the next two reports, and remain that way until after the body is discovered.

This same information can be found as a base64 encoded "value" string in the "DATE_DATA" table of the `origin_db_8ddbe561538e754225d70a5a122dd1f5` sqlite3 database. The base64 string expands to 4320 bytes (24 hours * 60 minutes * 3 fields). One byte is recorded for each of (category, intensity, steps), once per minute, for the entire day from local midnight to local midnight.

INSERT INTO "DATE_DATA" VALUES(0,5,'2017-07-17','{"slp":{"usrEd":-
1440,"wk":0,"usrSt":-
1440,"lt":0,"st":1500217200,"wc":0,"dp":0,"ed":1500217200},"v":5,"goal":5000,"stp":{
"rn":1,"cal":56,"runDist":124,"wk":45,"ttl":3242,"runCal":3,"dis":2159}}',NULL,NULL,
'[{"value":"AB0AEA0AABcAAAgAAEAMADwAADkOABEAAA8AAAIAAAUAACIAAA0AABAAAAkAAAYAACIAAAwA
AAcAAAMAAAQAABsAABgAACsAABIAABIAABMAABQAAAwAAA0AAAYAAAQAADIAACsAABAAABwAAAcAAAwAABQA
AAwAAAwAACcAACwAADwAAB4AABwAACYAABoAACQAABUAABEAAAMAAAMAAAUAAA4AAA4AABEAABQAAAkAABEA
AAcAAAQAAAgAAAcAAAgAAAYAABEAAAkAABcAAAwAAAkAABIAAAkAAA8AAA0AAAYAACUAAA0AAB8AAAIAAA4A
ABQAACUAABcAAAEAACUAABMAAB0AACgAAAkAAAEACxYAAAgAACIAABUAAAQAACUAABgAAAwAADkKACILAEMA
AEUKAEEAAEAAADgAACAAABoAAAsAABMAABQAABEAAAgAAAgAABQAAAgAABQAABcAAAEAACsAACEAABIAACYK
ACUAABEAACYAACUAABkAABUAABQAAAUAABAAABsAAC0AABQAAAwAAAIAAAkAABEAABkAABAAAD4AABQAAGYN
ADgAAUERACUAACQAAA8AAB8AABYAABcAAA0AABkAAAEAAAYAAA0AAAMAAAQAAA0AABIAAAwAAAIAAAMAAAIA
AA4AAAkAABYAAAYAAAYAABQAABUAAAsAAAEAABcAACkAAAoAAC4AAA4AAAQAAB8AAB0AACQAACoAAAwAAAgA
ABcAACMAAB0AAA4AACMAAB4AAAoAABEAABQAAA4AAAQAAA0AABUAAAUAABIAAAwAABAAAA0AAAgAAAkAABUA
AAQAAAUAAEcKAWsUAC0AAC8AAwAAAxQAAxQAAxQAAwAAAwAAAxQAAwAAAwAAAwAAAwAAAxQAAwAAAwAAAxQA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAxQAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAxQAAxQAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAxQAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAxQAAwAAAwAAAwAAAwAAAwAAAwAA
AxQAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAxQAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAxQAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwYAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAxQAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAxQA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAxQAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAxQAAwAAAxQAAwAAAwAAAwAAAwAAAwAAAwAAAwIAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAwAAAwAAAwMAAwcAAwcAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAA
AwAAAwAAAwAAAxQAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAABsAAVUUABMAHGUmAU4RAXo6
AXVcAXNrAWFRAW9qAXNcAYxxAX1uAXhpAX5vAX50AW4wAC8AADIAABwAAA8AACAAAC8AAC0AACsAADgAADYA
AEEAEVElAUkfADwAAU4cIYd\/IZl+IaN+AWgnAEEAADUAATMQADAAADoAACAAABoAACAAAB0AADkAAB8AACc
AADUAAEUOABgAAB4AACUAACsAABAAABAAADIAADEAADMAAWEkAC8AAEQLAEgAAEEAAEoKAVIiADMAAD4AAD0
AADIAAFEPADkAAVMXAUUQAWswADcJAA0AAB8AABgAAB0AABEAAB4AADwMABsAABYAAA8AAVIVAC0AADwKABg
AACEAAB0AABgAACwAAAQAACoAADUAADIAAEYAACIAAB8AABoAAAMAACQAAB8AAAMAAAcAADkAACwAAWAXABg
AACkAABUAABQAAAIAACAAABMAACkAABsAAA8AABcAABcAACQAACEAADUAAAUAACgAACkAABwAABYAACgAABk
AAA0AABsAACcAAVgiACAAABwAADcAABgAADkAADAAACwAACwAACYAADIAACIAAT0WAWRBAU4iEVQoAVwYAWt
IACcAACQAADUKADsAAFwAAWgdAWIRAWdAAVYlADIAAAoAACYAAB4AABkAABQAABAAAA4AAAgAAV49AWk6AEY
AAEEMADsAAC4AABgAAB4AAB0AAA4AABoAAA4AACoAABQAABUAACAAACAAAEQAAXNUAVwoACoAADEAEV4kAUk
cAUgcABYAAA0AAC4AACIAACkAAAcAABIAADYAAA0AABgAAAsAAA8AABAAABgAEWc8AVIaAEgAAVEVAUgWAUY
ZAVUhAVo2ADAAADQAADwAABgAADgAACsAACgAABoAABwAAEQAABcAABkAAF4AAEUMADMAACgAACgAADcAACo
AACUAACQAADAAACAAABsAABIAAB4AACwAADUAAB8AABcAAB0AABoAACgAAEEAACoAACEAAB4AABMAACAAAC8
AADEAAB8AABIAABUAABMAABYAACQAABAAABcAAB4AABkAABoAAB4AAVgpAEwAACMAAwAAAwAAAwAAAwAAAwA
AAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAACwAAAwAAAwAAAwAAAwAAAwAAAwA
AAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwA
AAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAAwAAABUAAwoAAwAAAwAAAwAAABEAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA

```
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgA
AfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAAfgAA","did":"88F6C8B7050008
ED","stop":1439,"src":5,"tz":36,"start":0}]',NULL,NULL,1,NULL,36,'88F6C8B7050008ED')
;
```

Figure 20 - Betty's full MiFit sqlite3 activity record, day of murder

Additional personal information about Betty was found in the origin database, including reported birthdate (Feb 1995), height (165cm), and weight (50kg).

```
CREATE TABLE "USER_INFOS" ("USER_ID" TEXT PRIMARY KEY NOT NULL ,"NAME" TEXT,"BIRTHDAY" TEXT,"AVATAR_URL"
TEXT,"AVATAR_PATH" TEXT,"GENDER" INTEGER,"HEIGHT" INTEGER,"SYNCED" INTEGER,"WEIGHT" REAL,"TARGET_WEIGHT"
REAL,"CITY" TEXT,"LONGITUDE" TEXT,"LATITUDE" TEXT,"CREAT_TIME" TEXT,"LAST_LOGIN_TIME" TEXT);
INSERT INTO "USER_INFOS" VALUES('3015345853','betty','1995-
2','https://account.xiaomi.com/static/res/7c3e9b0/passport/acc-2014/img/n-avator-bg.png','',0,165,0,50.0,-1.0,'','','','0','0');
```

Figure 21 - Betty's personal information as recorded by MiFit

We also found information about the Mi Band itself, including its lack of support for GPS or heart rate measurement, its mac address (88:0f:10:f6:c8:b7), and its firmware version (5.16.11.15).

```
$ sqlite3 -batch ./databases/origin_db_8ddbe561538e754225d70a5a122dd1f5 '.dump device'
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE "DEVICE" ("DEVICE_ID" TEXT PRIMARY KEY NOT NULL ,"DEVICE_ADDRESS" TEXT,"DEVICE_TYPE"
INTEGER,"DEVICE_SOURCE" INTEGER,"DEVICE_BIND_STATUS" INTEGER,"DEVICE_BIND_TIME"
INTEGER,"DEVICE_SYNC_DATA_TIME" INTEGER,"DEVICE_SYNC_DATA_TIME_HR" INTEGER,"AUTHKEY" TEXT,"TIME_ZONE"
INTEGER,"TIME_ZONE_SYNC_HR" INTEGER,"SN" TEXT,"HR_FIRMWARE_VERSION" TEXT,"FIRMWARE_VERSION"
TEXT,"MIGRATE_DEVICE" INTEGER,"USER_ID" TEXT,"STATUS" INTEGER);
INSERT INTO "DEVICE"
VALUES('88F6C8B7050008ED','88:0F:10:F6:C8:B7',0,5,1,1499934004579,1500275943000,NULL,NULL,36,NULL,'',NULL,'5.16.1
1.15',NULL,'1714150043',NULL);
COMMIT;
```

Figure 22 - Details of Betty's MiBand

003-SimonNote2White

The zip file provided had a single folder that contained 7 partition images which aligns with the images found on the other Note tablet. The images provided match the basic partitions found on any Android device. This means that for the most part, we are interested in /data or USERDATA as it is labeled in our images. After setting up loop devices for each image and mounting them, we quickly found the SmartThings app at `USERDATA/data/com.smartthings.android`. Within that directory, the subdirectory databases contains a few sqlite3 databases with info on the views interacted within the application, but does not appear to contain any data related to the SmartThings devices on the home network. It appears that the phone is relatively new, and unused other than the most basic of setup.

Our review also revealed Simon's email messages (including registration for a joint Nest account with Betty) and some automation apps. We've included a plaso sqlite3 plugin that processes the SmartThings database as part of our submission, even though the SmartThings events recorded on the phone provided little information to support findings.

004-Onhub-diagnostic-report

The diagnostic reports from OnHub devices utilize Google's protocol buffer specification. By taking some time to extract each piece of data using the command line protobuf interface, a python script and schema definition were designed to expedite discovery of useful files and command results. The included `onhub_dump.py` and `onhub_diagnostic_pb2.py` scripts extract those files and commands. The onhub_dump.py script contains base logic to specify the diagnostic dump and the `onhub_diagnostic_pb2.py` script contains the schema definition. After successful completion, the `ohdump` directory is created and within are subdirectories `command` and `files`.

Extracted data of interest include `./files/_net_proc_arp`, which shows ARP info for the OnHub device, and `./commands/_bin_ps` auxwwf0, which shows processes running on the OnHub. Also, the name of the SSID of the AP can be confirmed by reviewing the results of the command iw dev0 in `./commands/_usr_sbin_iw` dev0.

IP and physical addresses from /net/proc/arp:

```
IP             MAC Address
192.168.86.28 109266000026
192.168.86.20 50f520000027
192.168.86.25 b827eb000028
192.168.86.26 50f520000027
192.168.1.167 2016d800001f
192.168.86.24 1caf05000022
192.168.86.29 2016d800001f
169.254.3.229 2016d800001f
192.168.86.21 a002dc000020
192.168.86.22 18b430000021
192.168.165.1 909f33000025
192.168.87.20 2016d800001f
192.168.86.27 d052a8000023
```

SSIDs from running /usr/sbin/iw dev0:

SSID	Frequency	MAC Address (BSSID)
HOME-guest	2437 MHz	f6f26d00001e
HOME	2437 MHz	f4f26d000018
HOME-guest	5180 MHz	f6f26d000012
HOME	5180 MHz	f4f26d00001a

005-Amazon-Echo-Alexa-Web-Scrape

The provided web scrape contained screen shots, json descriptions of each voice interaction, and some audio captures.

The artifacts were fairly straightforward, but the following document aided in understanding the Echo and its data formats:

https://lcdiblog.champlain.edu/wp-content/uploads/sites/11/2016/05/EDITED_Amazon_Echo_Report.pdf

A sample json file is as follows (8.json):


```
{
activity: {
_disambiguationId: null,
activityStatus: "SUCCESS",
creationTimestamp: 1500271959818,
description: "{"summary":"alexa how could you do this what are the
flooding","firstUtteranceId":"AB72C64C86AW2:1.0/2017/07/17/06/B0F00715535302W5/12:36::TNIH_2V.91ba6538-f8c8-
4922-8a1f-
8473a8a24c0cZXV/0","firstStreamId":"AB72C64C86AW2:1.0/2017/07/17/06/B0F00715535302W5/12:36::TNIH_2V.91ba653
8-f8c8-4922-8a1f-8473a8a24c0cZXV"}",
domainAttributes: "{"disambiguated":false,"nBestList":[{"spokenAnswerSsml":"<speak><prosody volume=\"x-loud\">Hmm, I
don&apos;t know that
one.<\/prosody><metadata><promptMetadata><promptId>NotUnderstood<\/promptId><namespace>SmartDJ.MusicQA<\/
namespace><locale>en_US<\/locale><overrideId>default<\/overrideId><variant>2017_Variant
6<\/variant><condition/><weight>1<\/weight><stageVersion>Adm-20170501_190432-
15<\/stageVersion><promptData><namespace>SmartDJ.MusicQA<\/namespace><overrideId/><prosodyPreRenderHook/><
promptID>NotUnderstood<\/promptID><\/promptData><\/promptMetadata><\/metadata><\/speak>","entryType":"Knowl
edge","answerText":"Sorry, I couldn't find the answer to your question.","answered":false,"searchQueryText":"what are the
flooding","validForGUI":true,"domains":[],"spokenAnswerText":"Hmm, I don't know that
one.","understood":false,"questionText":"What are the flooding?"}]}",
domainType: null,
feedbackAttributes: null,
id: "A32TRBM6QOXJ5H#1500271959818#AB72C64C86AW2#B0F00715535302W5",
intentType: null,
providerInfoDescription: null,
registeredCustomerId: "A32TRBM6QOXJ5H",
sourceActiveUsers: null,
sourceDeviceIds: [
{
deviceAccountId: null,
deviceType: "AB72C64C86AW2",
serialNumber: "B0F00715535302W5"
}
],
utteranceId: "AB72C64C86AW2:1.0/2017/07/17/06/B0F00715535302W5/12:36::TNIH_2V.91ba6538-f8c8-4922-8a1f-
8473a8a24c0cZXV",
version: 1
}
}
```


Figure 23 - Sample voice interaction record (8.json)

Notably, each interaction has a "summary" of what it thinks it heard, a description of the answer, and a "creationTimestamp" expressed in unix epoch time with millisecond resolution. A quick dump of timestamps could be obtained with the following command:

```
for i in *.json ;do echo -n $i: ; TZ=Asia/Seoul date -d @$(grep creationTimestamp $i | cut -c 20-29); done
```

Figure 24 – Console command to view GMT+9 timestamps on all voice interaction records

The following table describes all relevant captures, what was actually human-audible, and what Alexa thought it heard:

Table 6 - Integrated description of all Amazon Alexa scrapes

creationTime (all dates 2017-07-17)	**Audible Speaker(s)**	**Audible content**	**Alexa Summary**	**Supporting files *(plus history png files)***
15:20:34 KST	Simon	"Alexa! Call the ambulance!"	"who yes"	1.json, 1.wav
15:20:32 KST	Simon	"Alexa! Call the ambulance!"	"alexa"	2.json, 2.wav
15:20:07 KST	Simon	"Alexa <*door close*> turn off tv"	"turn off tv"	3.json, 3.wav
15:20:05 KST	Simon	"Alexa <*door close*> turn off tv"	"alexa"	4.json, 4.wav
15:13:02 KST	Betty	"Alexa <*music stops*> stop"	"stop"	5.json, 5.wav
15:12:58 KST	Betty	"Alexa <*music stops*> stop"	"alexa"	6.json, 6.wav
15:12 (from history first page.png)	Betty/John	*Betty:* "alexa" *John:* "I can't believe you would do this to me." *Betty:* "stop" *John: "* You said we would. What are you thinking."	"alexa what do this to me stop we said we would what are you thinking" *(from history first page.png)*	7.json (empty) 7.wav
15:12:39 KST	Betty/John	*Betty:* "alexa" *John:* "How could you do this?! What are you thinking?" *Betty:* "stop"	"alexa how could you do this what are the flooding"	8.json, 8.wav
15:06:06 KST	Betty	"alexa, turn on pandora"	"turn on pandora"	9.json, 9.wav
15:06:03 KST	Betty	"alexa, turn on pandora"	"alexa"	10.json, 10.wav
15:01:55 KST	Simon	"alexa, turn on tv"	"turn on tv"	11.json, 11.wav
15:01:54 KST	Simon	"alexa, turn on tv"	"alexa"	12.json, 12.wav
14:45:31 KST	Simon/ Betty/ unknown	*Simon:* "And say 'alexa'" *Betty:* [words inaudible] *unknown:* [words inaudible]	"wake up"	13.json, 13.wav
11:57:35 KST	John	"Alexa, time to test AI, [words inaudible] AI"	"alexa kinda patton a.m. next monday"	14.json, 14.wav
12:07:00 KST	*<no audio>*	*<no audio>*	""	15.json
14:31:04 KST	*<no audio>*	*<no audio>*	""	16..json
14:45:29 KST	*<no audio>*	*<no audio>*	"alexa"	17.json
14:45:43 KST	*<no*	*<no audio>*	"alexa stop"	18.json

	audio>			

These captures allowed us to identify John Macron's voice, correlate his dialogue with the Google Hangouts argument, and identify a time when he had to be in the house *(15:12 GMT+9)*. We can also see evidence of Betty's and Simon's activities, including times when Sally must and Simon must have been within audible range of the Echo.

006-SmartHome-Cloud-Provider-AcmeInc-NetworkDump.tcpdump

The challenge included a packet capture, taken at the "cloud provider" for the smart home. Timestamps on the capture began at "2017-07-17 09:00:57 UTC", but it was unclear if the collecting host had the correct system time.

Fortunately, there were two easily-parseable NTPv4 packets early in the capture: packets 2607 and 2608.

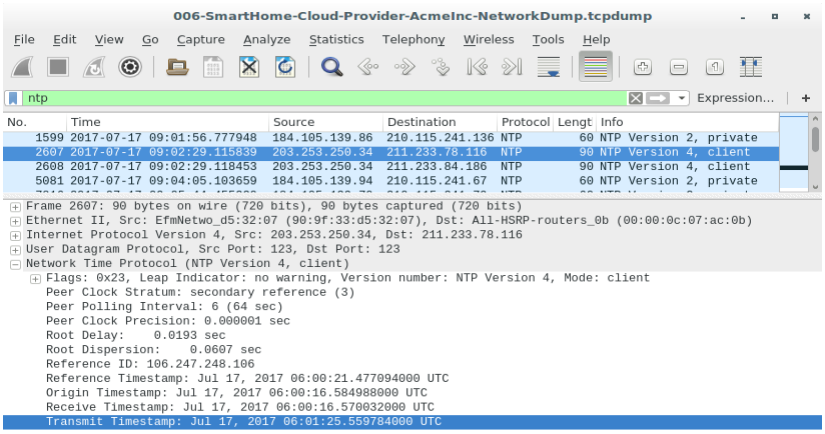


Figure 25 - Mismatch between packet timestamp and NTP-reported time

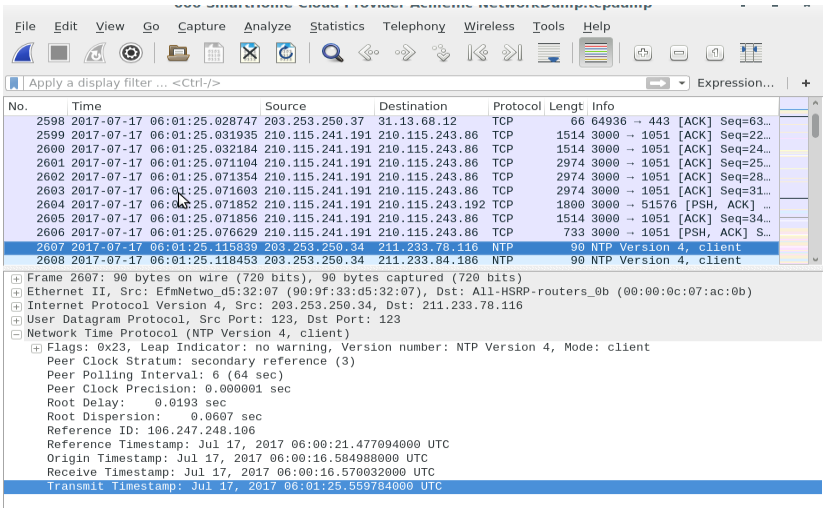


Figure 26 - Packet timestamps after time-adjustment

From this capture we can see that the sender (203.253.250.34) believed it was transmitting at time **06:01:25 UTC**, while the capture timestamp reports **09:02:29**. Both timestamps agreed on the date: **2017-07-17**.

Based on this information, we time-shifted the capture by **-03:01:04** to synchronize capture timestamps with NTP-reported time. After this shift, the packet capture was found to overlap the times reported in the case report. Events were then correlated between packet capture, phones, and Amazon Echo.

The cloud provider tcpdump contained traffic from multiple subnets, but the most relevant traffic was between the following two devices:

203.253.250.32 Neo-IDM IoT Device Management Server

203.253.250.34 SmartThings (CoAP, HTTP) peer

These two devices interacted throughout the entire capture, passing both TCP and UDP traffic between them:

Table 7 - Packet flows traced to the household IoT hub

Flow Number	Protocol	Server/source	Client/destination	Description	Interval (UTC+9)
1	TCP: HTTP+JSON	203.253.250.32:8080	203.253.250.34:55133	IPSO Event reports Resource: **/13001/0/5823** **Payload: *matches flow #7***	Full capture
2	TCP: HTTP+JSON	203.253.250.32:8080	203.253.250.34:55134	IPSO Event reports Resource: **/21011/2/5823** **Payload: *matches flow #6***	Full capture
3	TCP:HTTP PUT	203.254.250.34:8080	203.254.250.32:55141	Put "value: true, id:5550" to: /v1.0/clients/DollHouse_Secu5/3201/0/5550?at=f85f50ed-3b6a-4dab-9259-0c7bfec598ea *Translates to command flow #9*	15:12:13.41 to 15:12:13.69
4	TCP:HTTP PUT	203.254.250.34:8080	203.254.250.32:55142	Put "value: true, id:5850" to /v1.0/clients/DollHouse_Secu5/3312/1/5850?at=f85f50ed-3b6a-4dab-9259-0c7bfec598ea *Translates to command flow #10*	15:19:22.81 to 15:19:23.07
5	UDP:CoAP	203.253.250.34:1024 **Token: ed:53:06:42**	203.253.250.34:5683	Content (2.05) **Example payload: 00600070**	Full capture
6	UDP:CoAP	203.253.250.34:1024 **Token: ed:53:06:3e**	203.253.250.34:5683	Content (2.05) **Example payload: "002552"**	Full capture
7	UDP:CoAP	203.253.250.34:1024 **Token: ed:53:06:3a**	203.253.250.34:5683	Content (2.05) **First payload: "5318733119"** **Subsequent payloads: "notag"**	Full capture
8	UDP:CoAP	203.253.250.34:1024 **Token: ed:53:06:3f**	203.253.250.34:5683	Content (2.05) **All captured payloads: "notag"**	Full capture
9	UDP:CoAP	203.253.250.34:1024 **Token: ed:53:06:69**	203.253.250.34:5683	PUT(3) + Changed(2.04) URI: /3201/0/5550 **Payload: "1"** *CoAP translation for flow #3*	15:12:13.41 to 15:12:13:69
10	UDP:CoAP	203.253.250.34:1024 **Token: ed:53:06:69**	203.253.250.34:5683	PUT(3) + Changed(2.04) URI: /3312/1/5850,	15:19:22.81 to

				Payload: "1" *CoAP translation for Flow #4*	15:19:23.07

After some searching and analysis, these packets were found to match the Constrained Application Protocol (CoAP) as defined in RFC 7252 (https://tools.ietf.org/html/rfc7252) and the IP for Smart Objects (IPSO) Alliance (https://github.com/IPSO-Alliance/pub) standards.

Object Identifiers in Flows 3/9 and 4/10 were found to match standard Object ID and Resource ID's from the IPSO reference (https://github.com/IPSO-Alliance/pub/tree/master/reg).

Based on these references:

- Flows 5, 6, 7, and 8 are CoAP *Observe* (RFC 7641) responses, where the original *Observe* request was not captured.
- At 15:12:13 GMT+9, the command in Flow 3 engages an Actuator (IPSO Object 3201) Digital Output State (Resource 5550). This is translated to CoAP commands in Flow 9. We believe this command opened the deadbolt on the front door.
- At 15:19:22 GMT+9, the command in Flow 4 engages a Smart Power Source (IPSO Object 3312) On/Off (Resource 5850). This command is puzzling because, if it were Simon turning his Smart TV off, the command should have been "false" (*OFF*). Instead, it was "true", which is specified to mean *ON* (from https://github.com/IPSO-Alliance/pub/blob/master/reg/xml/3312.xml) :

```
...
<Item ID="5850">
        <Name>On/Off</Name>
        <Operations>RW</Operations>
        <MultipleInstances>Single</MultipleInstances>
        <Mandatory>Mandatory</Mandatory>
        <Type>Boolean</Type>
        <RangeEnumeration></RangeEnumeration>
        <Units></Units>
        <Description>This resource represents a power relay, which can be controlled, the setting of which is a Boolean value (1,0) where 1 is on and 0 is off.</Description>
</Item>
...
```

Figure 27 - IPSO Alliance specification for Smart Power "On/Off" resource

We also noted that the data reported in Flow 6 tracked with the *Flow 4* command issued to the smart power source (/3312/1/5850). Prior to issuing Flow 4, the top two digits of Flow 6 were consistently "00". After Flow 4, those same digits were consistently "01". This leads us to believe that Flow 6 is Observing the smart power resource.

Unfortunately, we were not able to completely identify the *Observe* events and their meanings within the allotted time. We were able to correlate the Actuator and Smart Power events, however.