Fitbit Firmware Hacking

ReCon 0xE

Jiska Classen & Daniel Wegemer
jclassen@seemoo.de
dwegemer@seemoo.de

Technische Universität Darmstadt
Secure Mobile Networking Lab - SEEMOO
Department of Computer Science
Motivation

Devices can be hacked, research
Communication
Communication Paradigm

Tracker → BLE → App → HTTPS → Server

Device-specific symmetric key

End-to-end encryption

Recent trackers only ...

Memory readout attack in firmware before October 2017
Activity Record Synchronization

Tracker

Store activity locally

Server settings

Susi Fit

App

Megadump & Megadump response
Local activity / Data deletion, user and device settings

Server

Activity database

each 16min

Display activity summary

Decrypted summarized data

JSON summary
Authenticated Live Mode

1. User login
2. Local pairing
3. Remote association to local user
4. Authentication credentials
5. Authenticated live mode / memory readout
6. Plaintext data

obtain once
reused forever

Authenticated Live Mode

Authenticated Live Mode

Authenticated Live Mode

Authenticated Live Mode

Authenticated Live Mode

Authenticated Live Mode

Authenticated Live Mode

Authenticated Live Mode
Accessing the Fitbit Hardware
Goals:
1. Access/Modify local storage
2. Get encryption keys
Fitbit Flex Hardware & Software

Firmware Libs:
- Encryption: LibTomCrypt
- BLE: LibBLEShield
- ...

Main SOC:
STM32L151UC (ARM Cortex M3)

Bus (e.g. UART)

BLE CI:
nRF8001

Accelerometer:
LIS2DH
Hardware Access

Testing points to connect to debugger:
- TP8 SWDIO
- TP9 SWCLK
- TP10 NRST
- GND (from battery)

Goals:
- Dump firmware
- Modify stored data

CC BY-NC-SA 3.0 ifixit.com
(Sam Lionheart)
Flash
  • Firmware code
EEPROM
  • Information that should survive empty battery
SRAM
  • Firmware variables
Fitbit Flex Hardware is Cheap!

Some Fitbits were harmed during our experiments...
Flash Contents

- **APP** ~ 1000 functions (including BSL duplicates)
- **BSL** ~ 500 functions
- Both BSL and APP run **independently**

- **Serial number**
- Encryption **key**
- Encryption switch
- Fitness data
Debugger Access
- Debugging is only enabled **during reset**
- Firmware initialization **disables GPIO ports** necessary for debugging
- Lets reset them!

How? NexMon!
- NexMon is a **binary patching framework**
- We adapt NexMon for the Fitbit firmware
- Goal:
  - **Modify** firmware
  - Enable **dynamic debugging** (GDB)
Re-Enabling GDB Access (2)

- **Connect after reset** works fine

- GPIOs needed for Debugging get **reassigned** during initialization
  \(\rightarrow\) Where? No idea!

- Let **patch the firmware** to reenable the GPIOs necessary for debugging **after initialization has finished**
  \(\rightarrow\) This might comes with **side effects**
  \(\rightarrow\) Use **Bluetooth commands** to trigger the reprogramming
Wireless Fitbit Firmware Flashing
Update Process

Fitness record synchronization includes FW version

New FW available

FW request

https://.../firmware.json

Request: Microdump
Response: BSL, APP

Validate
Write to flash
Reboot to BSL

Tacker
BSL
APP
Firmware & Dump Encryption

Newer trackers come with encryption **enabled by default**
→ We need to know how **encrypted firmware updates** work

Trackers use **XTEA/AES in EAX mode**:
- **2 byte nonce** in beginning of each dump
- **128 bit encryption key**, extractable from EEPROM via memory readout attack
- **8 byte authentication MAC** in the end of each dump before length field
  → Firmware is based on **LibTomCrypt** (C)

Based on own reverse-engineering and insights of M. Schellevis
Steps to Flash Modified Firmware

1. Get key (memory readout)
2. Get firmware (memory readout)
3. Copy firmware
4. Modify with nexmon
5. Copy firmware
6. Format + Encrypt
7. Flash modified firmware back

https://github.com/seemoo-lab/fitness-firmware
https://github.com/seemoo-lab/fitness-app
Demo

```
return steps * 100;
```
Accessing the Accelerometer

Locating Accelerometer Values

- **Factory test** functions include accelerometer printout
- Points to the correct register which is updated by the accelerometer driver
## Configuration Registers

<table>
<thead>
<tr>
<th>Register Address</th>
<th>Purpose</th>
<th>Bit Mask</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>Data rate and power mode</td>
<td>57</td>
<td>All axes active, <strong>100Hz</strong></td>
</tr>
<tr>
<td>21</td>
<td>Filter mode</td>
<td>57</td>
<td>Highpass filter enabled</td>
</tr>
<tr>
<td>22</td>
<td>Interrupts</td>
<td>00</td>
<td>All interrupts disabled</td>
</tr>
<tr>
<td>23</td>
<td>Endian data selection and self test</td>
<td>00</td>
<td>LSB at lower address and normal self test mode</td>
</tr>
<tr>
<td>24</td>
<td>Boot mode</td>
<td>57</td>
<td>Normal boot mode and FIFO enabled</td>
</tr>
<tr>
<td>25</td>
<td>Interrupts</td>
<td>00</td>
<td>All interrupts disabled</td>
</tr>
<tr>
<td>2e</td>
<td>FIFO control register</td>
<td>8f</td>
<td>FIFO stream mode</td>
</tr>
<tr>
<td>30</td>
<td>Interrupt event register</td>
<td>00</td>
<td>All interrupts disabled</td>
</tr>
<tr>
<td>32</td>
<td>Interrupt threshold register</td>
<td>00</td>
<td>All interrupts disabled</td>
</tr>
</tbody>
</table>

Looks okay 😊 Can we copy this with >25Hz?
Accelerometer Live Mode

- Live mode normally only updates if activity data changes

- Modifications for accelerometer live mode:
  - Maximum update rate: \(~66 \text{ Hz}\)
  - Copy \(x, y, z\) accelerometer data to live mode variables
  - Ensure **backward compatibility** by only enabling accelerometer live mode after a special command
Use Cases

• **Develop** accelerometer-based applications on any **Bluetooth** capable platform.

• Possibility to **port** these applications later on with C/Nexmon and wirelessly flash them onto your Fitbit, e.g. recognition of different types of movements and gestures with the same **low battery usage**.
**Hackable Models & Versions**

Encrypted wireless *firmware modifications* (requires memory readout):

<table>
<thead>
<tr>
<th>Tracker</th>
<th>Firmware Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>One</td>
<td>5.60 (before October 2017)</td>
</tr>
<tr>
<td>Flex</td>
<td>7.81 (before October 2017)</td>
</tr>
<tr>
<td>Charge HR</td>
<td>18.102 (older...)</td>
</tr>
</tbody>
</table>

If you buy new trackers online, they have a firmware < October 2017 😊
Project Status

• Cooperation with the Fitbit Security team for Responsible Disclosure
  • Communication encrypted with PGP 😊
  • Professional categorization of security issues we report.
  • They get early versions of our publications, we get early feedback.

• Open source tool to run server-independent actions on fitness trackers, such as live mode, memory extraction and firmware flashing & a framework to craft your own firmware:
  https://github.com/seemoo-lab/fitness-app
  https://github.com/seemoo-lab/fitness-firmware
Summary

1. Go out and flash your neighbors’ devices.
2. Keep control of your own data.
3. Run any code on your Fitbit.