Semiconductor
Research
Corporation

hacking for fun and glucose

reverse engineering an insulin pump

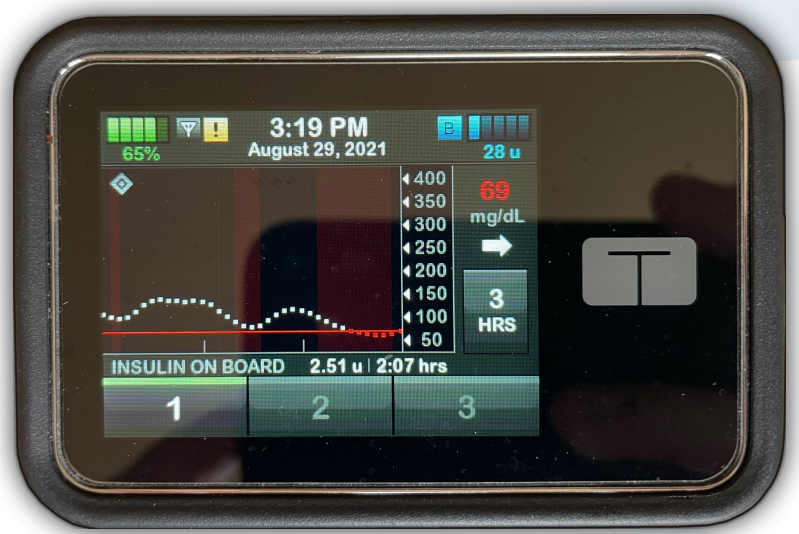
Alex Bellon, Deian Stefan, Alex Snoeren (UCSD)

- blood sugar = how much glucose is in your blood
- pancreas = keeps your blood sugar levels in check
  - insulin = lowers blood sugar, normally made naturally by your pancreas
- diabetes = pancreas doesn't work well (or at all)
  - insulin pump = helps manage blood sugar by administering insulin

- high blood sugar is bad
  - thirst (body wants to flush out the extra sugar), headaches, blurred vision, coma, etc
- low blood sugar is worse
  - your body starts to shut down organs to conserve energy, and in the worst case you can die

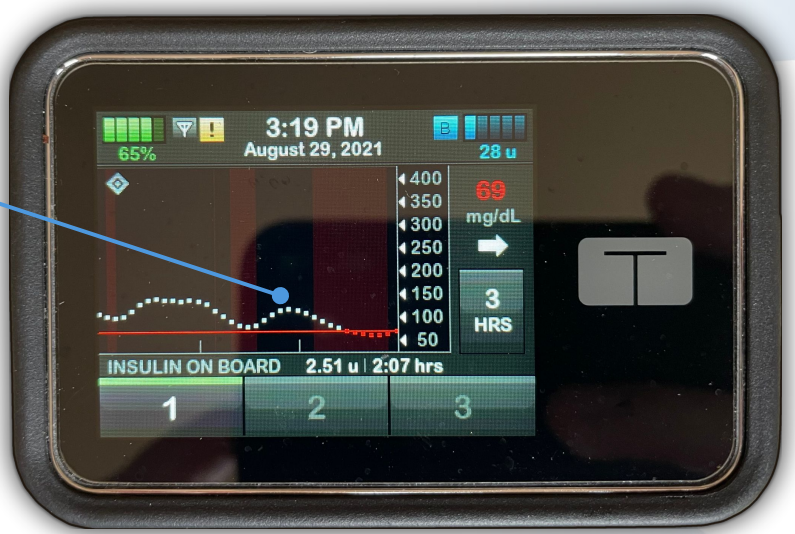
- high blood sugar is bad
  - thirst (body wants to flush out the extra sugar), headaches, blurred vision, coma, etc
- low blood sugar is worse
  - your body starts to shut down organs to conserve energy, and in the worst case you can die
- we want to make sure that these devices are robust

charting blood sugar level over time, the red line is the dangerous zone



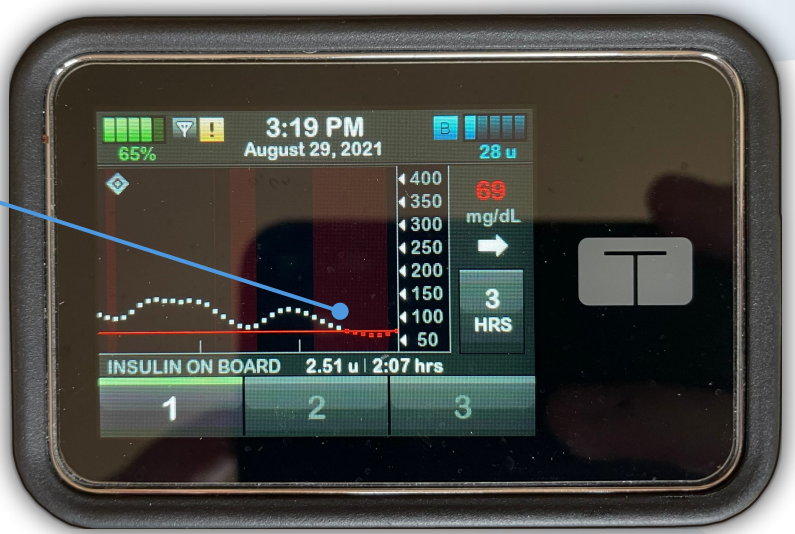
insulin pump

high blood sugar,
administer insulin



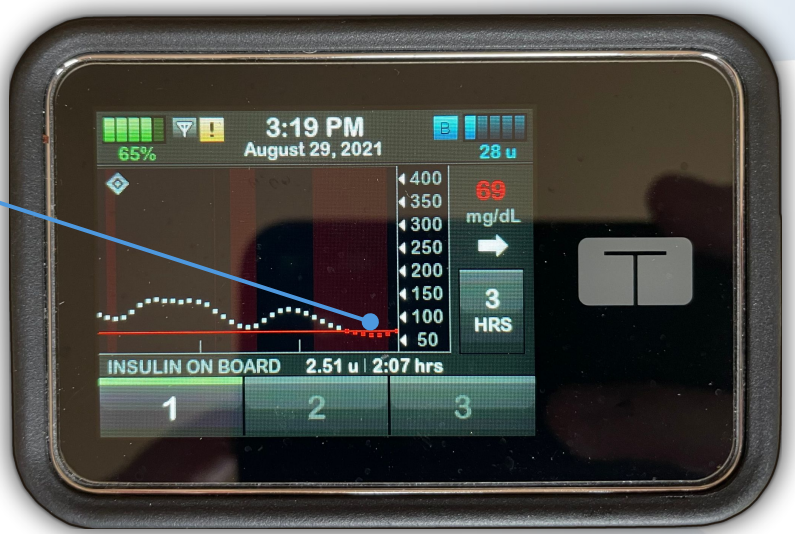
insulin pump

blood sugar continues to lower, approaching danger zone



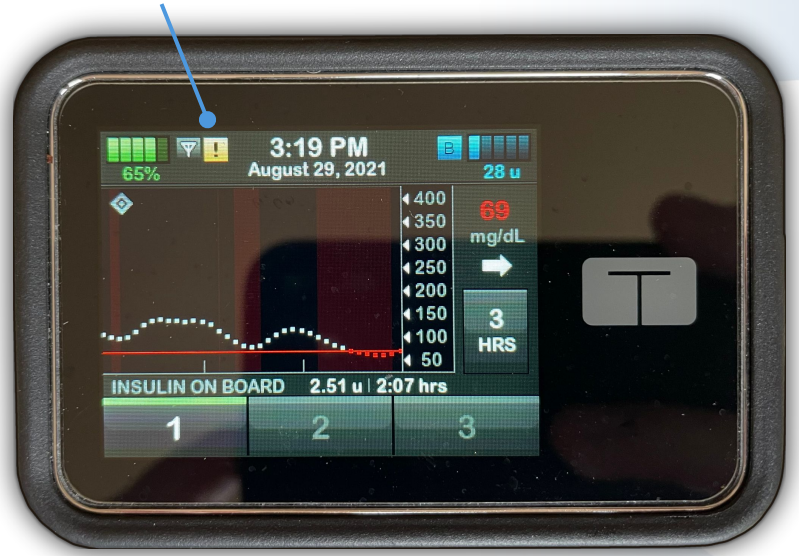
insulin pump

blood sugar is now dangerously low



insulin pump

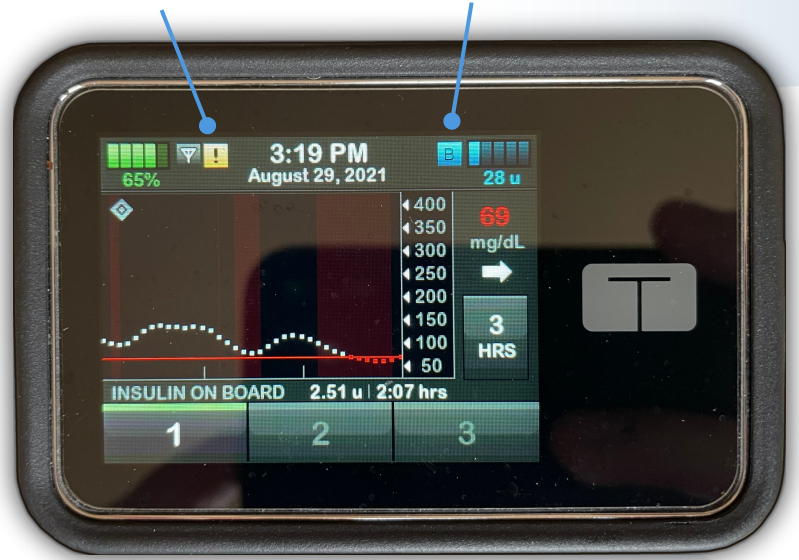
pump *knows* blood sugar is too low, but...



insulin pump

pump *knows* blood sugar is too low, but...

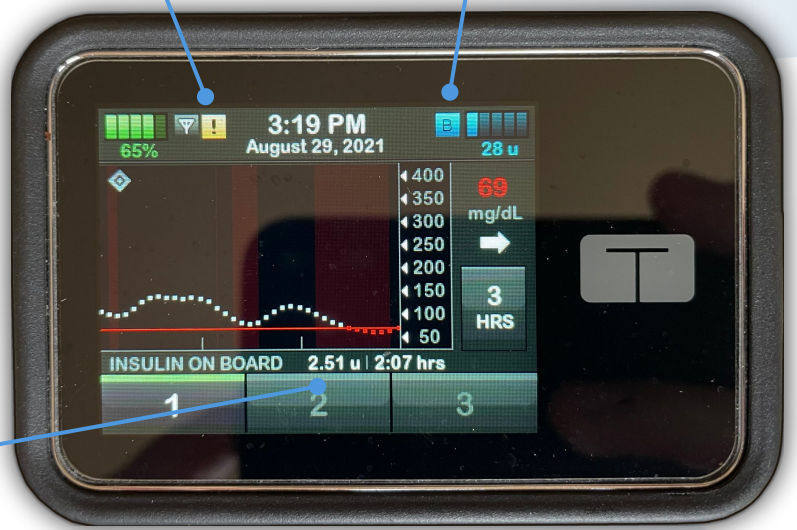
...is still administering insulin!



insulin pump

pump *knows* blood sugar is too low, but...

...is still administering insulin!



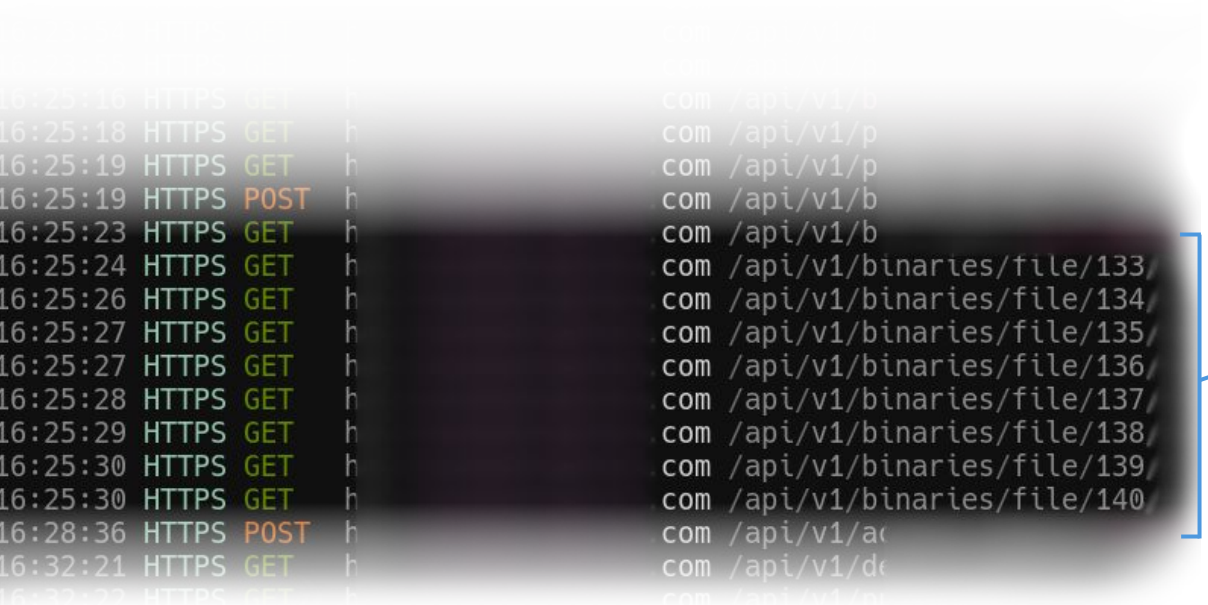
insulin pump

already a non-zero amount of insulin in the body

- a bug like this can have serious consequences
  - ...and this is just *one* bug that we *know of*
- what could an attacker get the pump to do when acting maliciously?
  - spoof blood sugar levels, control insulin delivery...
- we better look into the code behind this!
  - not so fast...

- getting the firmware
  - we have the desktop program that updates the firmware on the insulin pump
  - try intercepting the firmware

intercepting web traffic



downloading the files for the firmware!

- unfortunately the files were encrypted, so we either need to find the encryption keys, or code that would decrypt the firmware
- luckily, we have the insulin pump itself!

- lots of filing and prying later...

(other side) main STM chip, handles main computing

- lots of filing and prying later...

nRF chip, handles bluetooth

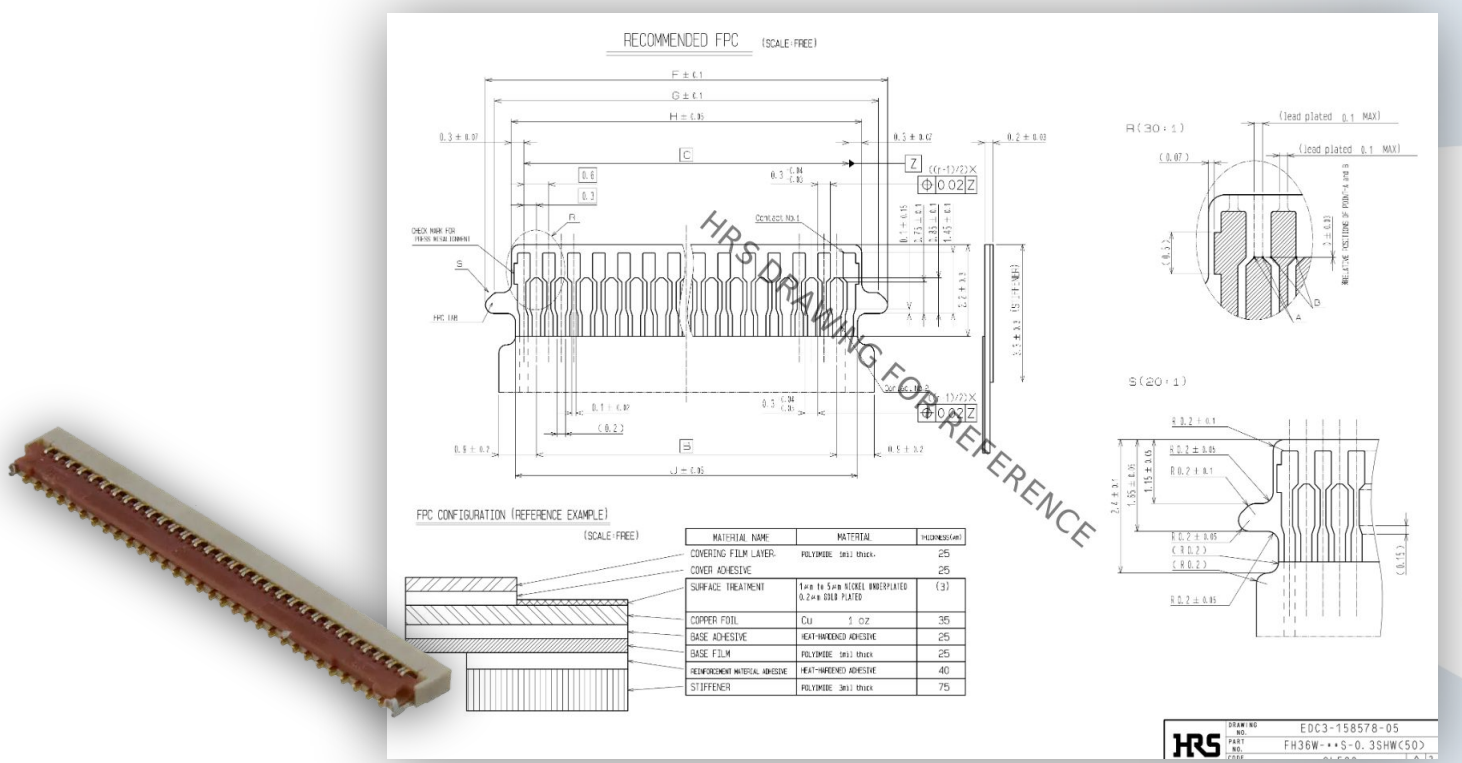
- lots of filing and prying later...

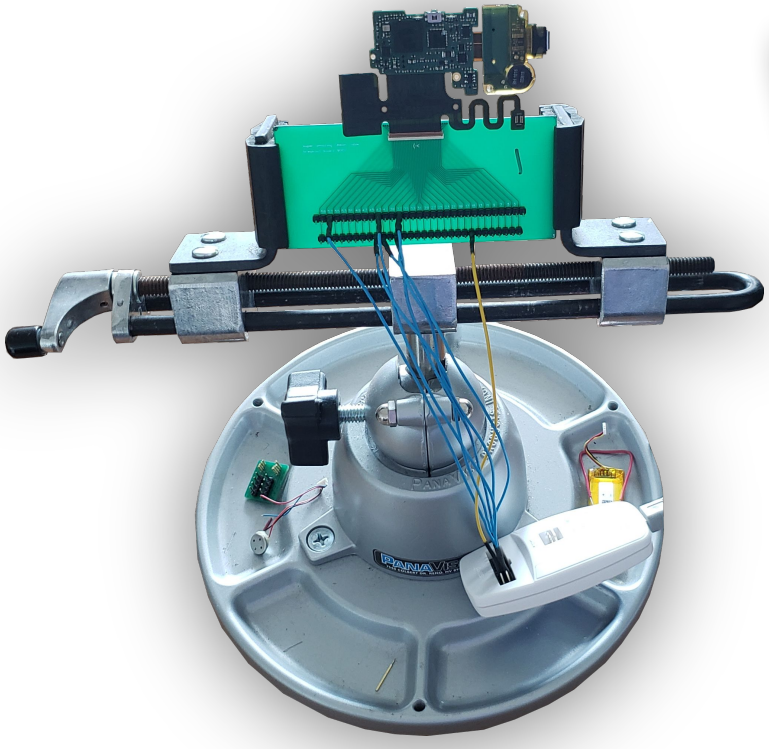
TI chip (other side), use unknown

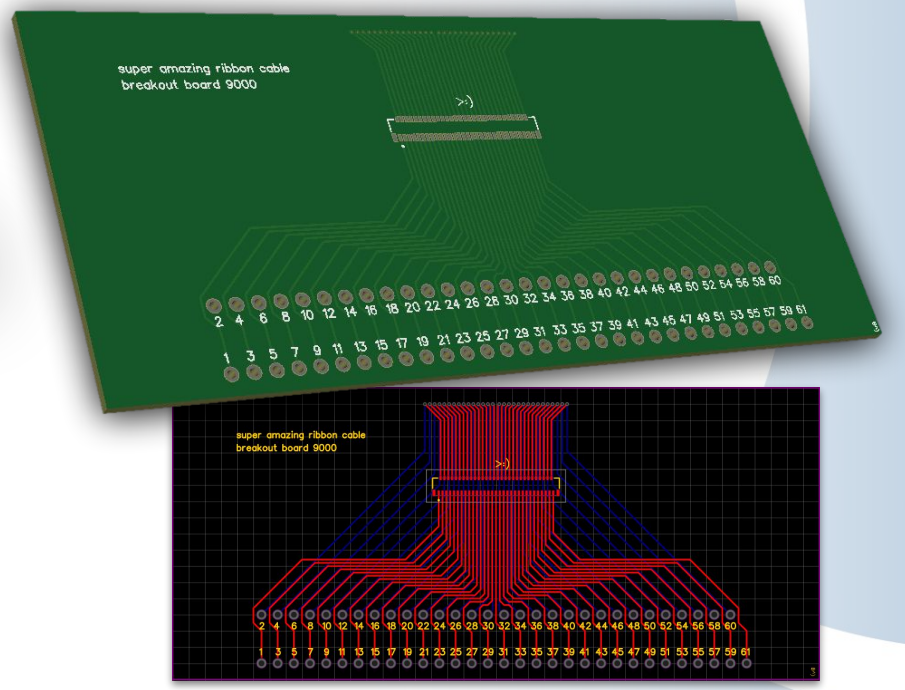
- lots of filing and prying later...

exposed ribbon cable,
probably used for debugging

- the pads on the ribbon cable should connect to the chips on the board
- all of the chips have some form of "debugging" or communication protocol that might allow us to read off their firmware
  - JTAG, SWD, SPI, etc
- if we can find connections for the pins associated with these protocols, we can connect to them


RECOMMENDED FPC
(SCALE:FREE)
HRS DRAWING FOR REFERENCE
FPC CONFIGURATION (REFERENCE EXAMPLE)
(SCALE:FREE)
MATERIAL NAME
MATERIAL
COVERING FILM LAYER
POLYIMIDE 1mil thick.
COVER ADHESIVE
SURFACE TREATMENT
1μm to 5μm NICKEL UNDERPLATED 0.2μm GOLD PLATED
COPPER FOIL
Cu 1 oz
BASE ADHESIVE
HEAT-HARDENED ADHESIVE
BASE FILM
POLYIMIDE 1mil thick
REINFORCEMENT MATERIAL ADHESIVE
HEAT-HARDENED ADHESIVE
STIFFENER
POLYIMIDE 3mil thick
HRS
EDC3-158578-05
FH36W-**S-0.3SHW(50)




super amazing ribbon cable
breakout board 9000
super amazing ribbon cable
breakout board 9000

SRC

- getting the nRF firmware
  - able to get the flash, RAM, registers and UICR
  - comparing the firmware in Ghidra to the SDK that Nordic (nRF) distributes, there's a lot of overlap

```
→ strings flash.bin
…
UPLOAD_START
crash_dump.c
…
```

- getting the STM firmware
  - able to dump the multiple regions of memory including the flash, where the main firmware is
  - lots of interesting content

```
→ st-flash read firmware.bin 0x0 1048576
st-flash 1.7.0
2022-02-22T10:52:27 INFO common.c: F1xx XL-density: 96 KiB SRAM, 1024 KiB flash in at least 2 KiB pages.
2022-02-22T10:52:27 INFO common.c: read from address 0000000000 size 1048576
```

```
→ strings firmware.bin
 NRF5 App FAILED CRC AFTER DECRYPT!
 ALPHAMASK FAILED CRC AFTER DECRYPT!
 NRF5 SD FAILED CRC AFTER DECRYPT!
 NRF5 BL FAILED CRC AFTER DECRYPT!
 RASTER FAILED CRC AFTER DECRYPT!
Decrypt and CRC Check Externals...
 ARM FAILED CRC AFTER DECRYPT!
 MSP FAILED CRC AFTER DECRYPT!
Decrypt and CRC Check ...
```

```
INSULIN SUSPENDED
All deliveries were automatically
stopped. Insulin will resume when
sensor readings start to rise.
CLOSE
INSULIN RESUMED
Insulin was automatically resumed.
Your max insulin suspension has
been reached. Insulin was
automatically resumed.
Basal-IQ Suspend
Basal-IQ Resumed
Basal-IQ Auto Resume
```

```
================== Main Menu ==============================
  Download Image To the STM32F10x Internal Flash ------- 1
  Execute The CTX Application -------------------------- 2
  Set SPI Flash Block Offset SW ------------------------ 4
  Set SPI Flash Block Offset HW ------------------------ b
  Download BIN to SPI Flash ---------------------------- 5
  Download BIN to MSP Flash   -------------------------- 7
  Execute The MSP Application -------------------------- 8
  Reboot ----------------------------------------------- 9
  Enable High Current ---------------------------------- 0
  Download Alphamask ----------------------------------- a
  Download Raster -------------------------------------- r
  Download BIN to NRF5 Bootloader ---------------------- e
  Download BIN to NRF5 SD ------------------------------ f
  Set NRF5 Start Adr ----------------------------------- g
  Set NRF5 Chksum Adr ---------------------------------- h
  Download BIN to NRF5 App----------------------------- i
  NRF5 Bootloader Version ------------------------------ j
  Set files download bitmask --------------------------- t
  Bootloader Version ----------------------------------- v
  SPI Flash Version ------------------------------------ w
============================================================
```

```
Hi There
what do ya want for nothing?
```

SRC

- using Ghidra and the "FindCrypt" plugin, found some cryptographic constants for AES and related functions
  - key schedule functions (converting main key into many round keys)
  - AES round functions (subBytes, shiftRows...)
  - overarching AES encrypt/decrypt functions

- unfortunately, none of these AES functions are actually called by anything (at least statically)
  - this might just be boilerplate code from a library
  - the functions might only be called with function pointers or other methods that you won't find statically

- we were able to find other interesting pieces of the firmware
  - functions to load firmware to chips and check the validity of the files
  - functions related to calculating insulin doses
  - functions to check versions, etc.

- currently, we are getting the firmware running in an emulator
  - using HALucinator [1] to handle interactions between firmware and (expected) hardware
  - connect this to a fuzzer to automate vulnerability finding

[1] Clements, Abraham A., et al. "{HALucinator}: Firmware Re-hosting Through Abstraction Layer Emulation." *29th USENIX Security Symposium (USENIX Security 20)*. 2020.

- at the same time, we are looking at the Android app that works alongside the insulin pump
  - recent update that allows fully remote insulin delivery
  - looking into the Bluetooth pairing/authentication, remote dosing, etc.

- short term future work
  - once we find vulnerabilities, craft exploits as proof of concept for the danger to users
  - find fixes to suggest to manufacturer
- long term future work
  - creating a framework that allows for developers to formally verify their firmware to ensure security

- tech transfer
  - not yet involved with industry, will share findings with manufacturer when we find vulnerabilities
  - still a work in progress, aiming to publish once we find vulnerabilities

- thank you to Deian Stefan, Alex Snoeren, Pat Pannuto, Aaron Schulman, Nishant Bhaskar for all the advice and help
- thank you to SRC for supporting this work


ACADEMIA • INDUSTRY •
• GOVERNMENT