Specifications for EV Charger

- Implementation in **Arduino C/C++**
- Battery level is an `int`. Normal values are between 0 and 100.
- M5 device displays the current battery level + a customizable message.
- Battery level decreases every second.
- Button A "inserts a coin". This gives 3 *charge tokens*.
- Button B uses one of the charge tokens. This increases battery level by 10.
- As soon as battery level is *< 0 or > 100*, the "organizer" **flag displays** on the M5 device **and** on the serial link. You may **not** modify the flag.

Serial link:

The serial link, baud rate 115200, offers 2 menus: message customization and admin backdoor. You **must** implement them.

- **Menu #1: message customization**. Selected by pressing `1`. User is asked to enter a text. Once entered, the console prints the customized message (see that as a confirmation). The message is displayed on the M5.
- **Menu #2: admin backdoor**. Selected by pressing `2`. When an admin enters the correct password (no, we don't give you the password), the battery level is automatically set to **-10**. You may **not** modify the SHA256 of the password.

Use the *same exact messages, numbers, and stop character for menus*. For example "Enter (# to finish):". Same baudrate.

> Basically, the functionalities + look and feel of the device may **not** be changed. Don't change the buttons to use, the strings, the flag, okay? You must only fix vulnerabilities :)

How to get the challenge flags

Bring your device, flashed with your patched code, to organizers,

1. We will first test if your device *complies to the rules*. If your device does not comply, we will tell you why (if possible). In particular, expect the organizers to test that the backdoor mechanism works ;)
2. After that, we, organizers, will run several exploits on your device. Our exploits are considered successful **if they retrieve the flag without using the backdoor**. Trust us: we're so pow3rful, we won't cheat :) Yeah. **Organizers will not show you their exploits, nor any screenshot or proof whatsoever.** You will have to trust us. You will only know the result: failed (no flag for you) or success (we give you the flag).

You get no flag if your device does not comply with the specs. If the specs are not clear to you, ask us *before* you bring your device for a test, or you'll waste an attempt...

You only have 2 attempts. (Only 2 for "Defend" challenges, not 2x2).

Example source code

We provide an example which *works* for *M5 StickC*, *M5 StickC Plus*, *M5StickC Plus 2*, *M5 Basic* and *M5 Core2* (but it has security issues). To win the flag, technically speaking, you do not need to use this source code and can implement yours from scratch. However, we believe it will save you time :)

Installation steps:

- Install esptool
- Use Arduino IDE and set it up for M5 StickC Plus 2

In Arduino IDE, install the following libraries and their dependencies:

- *M5Unified*
- *Crypto by Rhys Weatherley*

Clarification on flags

- "Organizer flag". That's the "flag" that organizers try to get when they exploit your device. It's fixed, and you may not modify it.
- "Challenge flags"/"Participant flags"/"CTF Flags"/Flags. There are 2 flags, depending on how well you secure your implementation. You enter those flags in CTFd. Organizers will give you the flag you deserve manually.

M5 Devices

We have a limited number of M5 Devices:

- Please only borrow one when you are ready to upload your program.
- Once you have finished, please re-flash it with the organizer's example (and bring it back).
- You are not authorized to tamper with the M5 in a way that would no longer make it usable by other participants.

References

- M5StickC Plus 2