Budget SoC

As in Baby SoC, we again have a ESP32 reversing challenge, this time of medium difficulty.

Please see the aforementioned writeup of the previous challenge for a comprehensive description of the extraction process.

Analyzing the Application

The application is almost identical. Only the calculation of the flag seems more complex, as it uses some values from memory that is only

initialized at runtime, it seems that one needs to dynamically analyze this. We didn't manage to get qemu-xtensa-esp32 to run, which is why we opted to just try and cheese this, as we had an ESP32 with us anyway.

Cheesing the Challenge

Using an ESP32, one can simply flash the flashdump onto it. The first page of the flash dump has been erased, probably to prevent it from just running on the device

But this can easily be fixed by just flashing a new bootloader over it.

1. Install the ESP-IDF toolchain for the ESP32
2. Compile one of the example projects
3. Flash just the bootloader from the project to the ESP32, e.g., `esptool.py --port /dev/ttyACM0 write_flash 0 build/bootloader/bootloader.bin`

Once this is done, we can run the ESP.

It will then open up an AP that asks for a Wi-Fi password. At the initial analysis of the strings in the flashdump, looking for a `flag` value, we found the suspicious `thisisnotaflag` string. This works as the Wi-Fi password. Then, we can navigate to the webpage served by the ESP32, which will, also, ask for a password. Retrying with `thisisnotaflag`, we get to the actual page, serving the flag:

> `justCTF{dUmp3d_r3v3rs3d_h4ck3d}`