Two-Factor Authentication with a Dedicated Hardware Device

Dylan Praul – Summer 2016

Abstract

Two-factor authentication (2FA) is a method of confirming a user's claimed identity, often implemented using a known password and an RFC 6238 - Time-Based One-Time Password (TOTP) – a time-sensitive token generated by a secret key. The goal of this project was to create a 2FA system where the user's authority is verified by a password and location is verified by a mounted TOTP-generating hardware device.

To accomplish this, the TOTP algorithm was implemented on an Arduino by taking in the current time from a DS3231 Real-Time Clock (RTC) module and using a secret key hard-coded in the Arduino memory. The current token is displayed on a series of 7-segment displays. An accompanying web-server was developed in Python using the Flask micro-framework to verify the token in a 2FA setup. This web-server authorizes the user to view a restricted page when the correct password and correct 2FA token is entered. The token can be verified by a parallel-implementation of the TOTP algorithm, or by direct communication to the Arduino over serial connection.

Table of Contents

Reasoning

The use case for this device is analogous to a ship, where many people can pilot the ship (because many people have boating licenses), but the current captain is only the person sitting in front of the wheel. In such a situation, the person’s authority is verified by a license that each individual knows, and the person’s current role is verified by his or her location. Only the person sitting in front of the wheel of the boat is assigned the role of captain.

For web services, there may be situations where company employees utilize a cloud-based recordkeeping service to which everyone knows the password to but only the person in the office needs access to. Instead of assigning each user a password and restricting their usage to the schedule they are assumed to be in the office, a hardware 2FA device can be placed in the office that restricts users from signing in unless they are physically seated in the office.

Such a system will be implemented with Stevens’ campus radio station, WCPR, as a means for DJs currently on-air (i.e. in the studio) to update the WCPR homepage with their name and song playlist. Such a system will restrict DJs not in the studio from tampering with other shows.

Hardware Implementation

The Arduino is an excellent choice for running periodic code that interfaces with physical sensors and output devices because of its low cost, ease of use, and small form-factor. The TOTP algorithm relies on two pieces of data: a secret key and the current time. Having no built-in network interface (unless installed separately) makes it safe from remote penetration, meaning it can safely store the secret key in onboard-memory. This also means that an onboard RTC module is necessary, since the Arduino lacks a perpetual time-keeping device and a time server cannot be used without a network connection.

The Arduino also has no display ports or built-in display panels, so something is needed to display the output of the TOTP algorithm to the user. A series of 7-segment displays is perfect because it is low cost, low power, and has a very standardized communications protocol.

RTC Module

The DS3231 Real-Time Clock module was selected because it is an extremely accurate time-keeping device with a modular battery and simple communication protocol. The module is wired to the 3.3V power line, and its SCL and SDA pins are wired to A5 and A4, respectively, on the Arduino. Thereafter, the Wire package works together with the DS3231 software library to easily read and write time from the device. As long as the battery remains attached and has battery, the module will keep time.

Because the TOTP algorithm relies on all devices having accurate timing, the RTC module needed to be synchronized with another timekeeping device. To do this, a sync_time sketch and accompanying python script was created. After the sync_time sketch is synced to the Arduino, it opens a Serial communication pipe with the host computer, sends a greeting bit, then waits for a UTC timestamp to be sent to update the current time with. The sync_time.py file automatically opens the Serial communication, listens for the greeting bit, then sends the current time packaged as a timestamp. This configuration allows the time on the RTC module to be synced with very high accuracy – on further implementations, Serial delays can be taken into account for even higher accuracy.

7-Segment Display Module

The 7-segment display module selected is an Anycubic 8-digit display assembly driven by the MAX7219 LED-matrix driver. The MAX7219 is a powerful LED-matrix driver that allows all 8 digits to be driven simultaneously via a simple software library. The module is wired 5V to VCC and DIN, LOAD/CD, CLK to data pins 12, 10, and 11, respectively. The displays can now be interfaced with using the LedControl library.

TOTP Algorithms

The TOTP Algorithm is based on the HMAC-based One-Time Password Algorithm (HOTP) with a timestamp replacing the incrementing counter found in the HOTP algorithm. Because it still utilizes HMAC, the TOTP algorithm relies on SHA1. Thus, software implementations of both the SHA1 and TOTP algorithms are needed for a 2FA system.

The secret key and time-refresh of the algorithm can be anything, but the standardized values established by Google Authenticator are 10-character secret keys and 30 second refresh time. Using these values allows the secret key to be distributed to administrators of the system, who can then access the web system using Google Authenticator running on their own phone.

On Arduino, the SHA1 algorithm is used courtesy of the Cryptosuite package, and the TOTP algorithm is provided in a separate package managed by Luca Dentella. The implementation works by getting the current time from the RTC module every 100ms and getting the current token from the TOTP algorithm. If the code is different from the one currently displayed on the display matrix (i.e. 30 seconds have lapsed), then a new display configuration is sent to the 7-segment array.

A Serial connection is also left open and polled frequently for connections, so that the web system has the option to retrieve the current 2FA over Serial rather than by re-implementing the 2FA system. Both functionalities are available on the implemented Flask system.

Software

All software for the system is available at https://github.com/dpraul/hardware-2fa. Instructions for setting up and running the Flask server are in README.md and copied here:

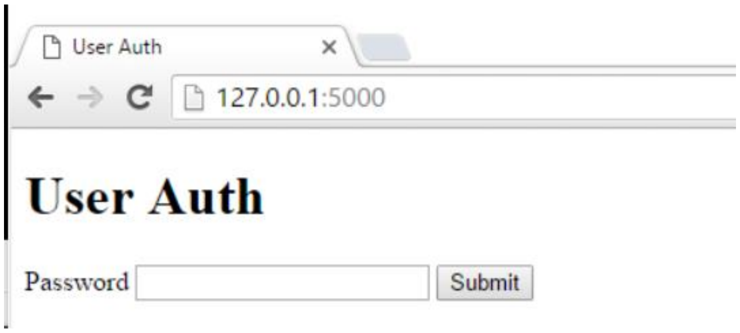
Running the server

Run the following commands from inside the `server` directory to launch the Flask 2FA server.

1. Create a virtualenv: `virtualenv env`
2. Join the virtualenv: `env\Scripts\activate`
3. Install dependencies: `pip install -r requirements.txt`
4. If the Arduino is connected to the computer, change `USE_DEVICE_COMMUNICATION` in `2fa.py` to `True`. Otherwise, leave it set to `False`. **
5. Start server at http://127.0.0.1:5000: `python 2fa.py`

**Enabling this feature allows the server to gather the 2FA token from the hardware device instead of by using the Python implementation.

Implementation

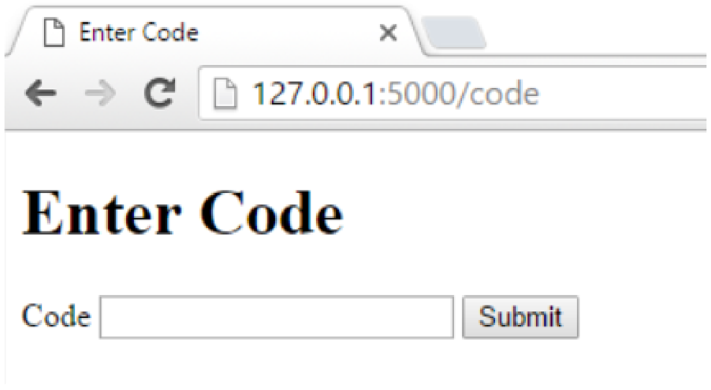


User Auth

Password

Submit

Figure 1 – The system waits for the user to enter a password.

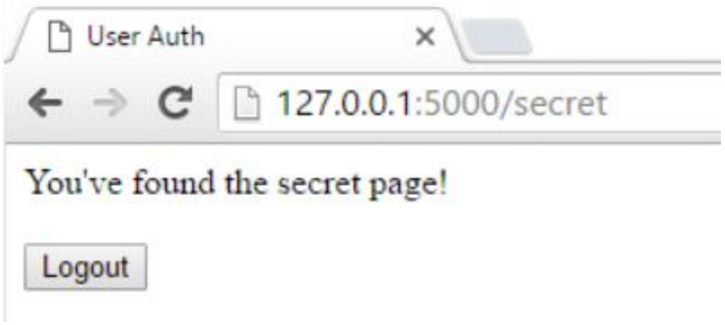


Enter Code

Code

Submit

Figure 2 – After receiving a password, the system asks for the 2FA token.



You've found the secret page!

Logout

Figure 3 – Having passed both hurdles, the user is granted access to the secret page.

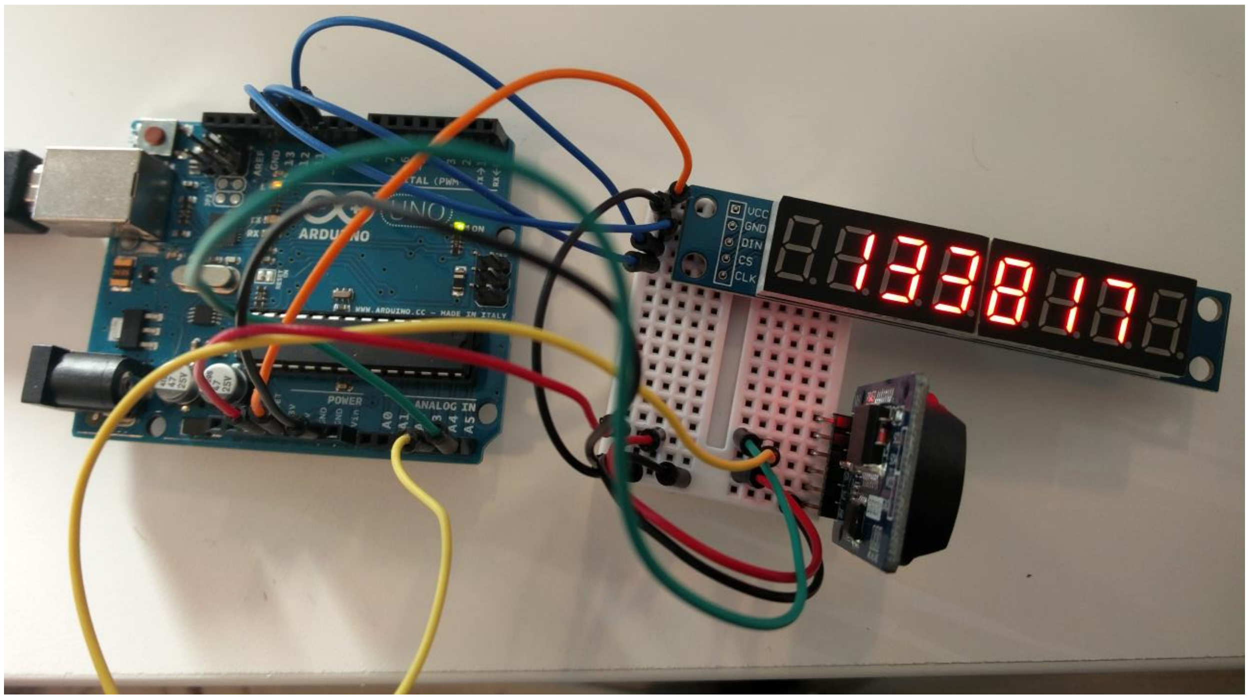


Figure 4 – Picture of the assembled hardware device in operation.

Software Resources

- Time-based One Time Passwords - https://en.wikipedia.org/wiki/Time-based_One-time_Password_Algorithm
- TOTP Arduino implementation - https://github.com/lucadentella/TOTP-Arduino
- SHA1 Arduino implementation - https://github.com/maniacbug/Cryptosuite
- Time library for Arduino - https://github.com/PaulStoffregen/Time
- MAX7219 LED display drivers for Arduino - https://github.com/wayoda/LedControl
- DS3231 drivers for Arduino - https://github.com/switchdoclabs/RTC_SDL_DS3231_ARDUINO
- OTP Tool for Arduino and Google Authenticator - http://www.lucadentella.it/OTP/

Datasheets

- DS3231 Real-Time Clock datasheet - http://www.switchdoc.com/wp-content/uploads/2014/12/File-1377714560.pdf
- MAX7219-MAX7221 datasheet https://www.sparkfun.com/datasheets/Components/General/COM-09622-MAX7219-MAX7221.pdf