Yubiduino: TOTP on an Arduino

Jessica Fleck, Brandon Mills, Paul Tela
Department of Computer Science and Engineering
The Ohio State University
Columbus, OH 43210
{fleck.48, mills.511, tela.3}@osu.edu

***Abstract*—Two-factor authorization (2FA) is a more secure method for authenticating users during login. Many 2FA systems, including the one used by the popular Gmail service, follow the Time-based One Time Password (TOTP) algorithm standardized in RFC 6238. Per the standard, the client and server agree on a shared secret key and starting timestamp at setup. Then, each time the user attempts to log in, the algorithm generates an ephemeral numeric passcode by running a cryptographic function with the shared secret key and the number of elapsed time intervals since the starting timestamp as inputs. However, this system requires a user to obtain and manually enter another unique password every time they log in. Inspired by the commercial YubiKey multi-factor authorization key, we designed and built a device, called a Yubiduino, that calculates and automatically enters the one-time password on the user's behalf. The device consists of an Arduino Due microcontroller, a DS3231 real time clock with a battery backup, a microSD card for persistent storage, and a button that the user presses to activate the algorithm. After setup, the user has only to plug the Yubiduino into a USB port, select the field in the login form for the one-time password, and press the button, and it will calculate and type the password automatically.**

I. Introduction

Two-factor authentication is becoming more and more popular in todays world because hacks are becoming more frequent. Two-factor authentication is an addition to your password that requires your password and then something you have, like a phone or a YubiKey. This keeps unwanted people out of your accounts because they would not have access to the 2nd factor of the authentication because you physically have it. The websites that support two-factor authentication usually hold valuable information about the user, which is a good reason to have extra security over your account. Because of this, we decided to create a YubiKey out of an Arduino and called it the Yubiduino. The Yubiduino creates the second layer of the two-factor authentication and makes your account more secure because no one else will have your Yubiduino. Using the Arduino IDE, we created cryptography code that generates a key each time the Arduino is pressed using the secret key and current time to add that extra layer of security to the account of your choosing. This Yubiduino will work with and base32-encoded key to allow users the option to use the Yubiduino for many different websites.

II. System Design

The process of creating the Yubiduino started with gathering all of the hardware needed. Everything needed was an Arduino Due, many wires, a real time clock, a button, a breadboard, and an Ethernet Shield. After the materials were collected, the wiring was done as shown in the two figures below.

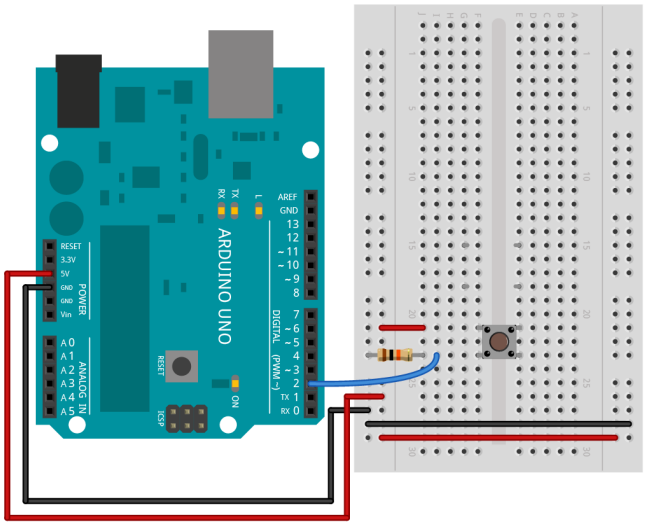

Fig. 1. Button Circuit

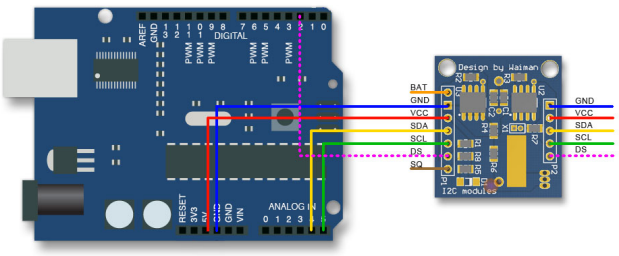

Fig. 2. RTC Circuit

The first step after the wiring was making the Arduino act as a keyboard when the button was pressed. The next step in the process was getting the real time clock to connect with the Arduino and being able to get the time each time the button is pressed to allow the new key to be created. Getting the clock

to work then lead into reading in the secret key from the SD card, which is the second half of the data needed to create the one time password. Then once both parts needed for the one time password were retrieved, we wrote the cryptography code to combine the two parts into a one time password needed to logging to the application.

III. Implementation and Evaluation

A. Implementation

* Handling button press * Programming the RTC * Writing via the keyboard * Accessing SD card storage and reading key from file. * Base 32 decoding * Getting the time from the RTC * Fixing Sha library * Generating C to pass to SHA1-HMAC * Calling SHA1-HMAC * Truncating the result according to TOTP spec

Yubiduino's software was implemented in C++ using the AVR-GCC compiler to target the Arduino Due board. This board uses a 32 bit ARM core micro controller running at 84Mhz. This micro controller allows for 4 byte wide data operations to occur in one clock cycle, which provides significant performance benefits for the cryptographic hashing code.

Several libraries were used in order to build the Yubiduino software package. Standard libraries used were `Serial`, `Keyboard`, `SD`, and `SPI`. In addition to these standard libraries, two third party libraries were used. These were `DS3231` and `Sha`.

The `Serial` library provides a way for the Arduino to communicate via a serial interface. This is primarily used for interacting with the Arduino using the Arduino IDE's built in Serial Monitor. The `Keyboard` library is used to allow the Arduino to send keyboard input to a connected computer. The `SD` library is used to read and write files on an SD card and supports both the FAT16 and FAT32 file systems. An ethernet shield was used in order to provide an SD card input. The `SPI` library allows for access to the Serial Peripheral Interface. This interface allows for short distance communication between micro controllers. This library was used to communicate with the Real Time Clock (RTC) over I^2C.

Third party libraries were used when standard library was not available. The `DS3231` library was used to communicate with the RTC. It provided convince methods for reading the current time and converting the time between different formats. The `Sha` library provides SHA1 and SHA1-HMAC cryptographic hashing capabilities.

B. Evaluation

IV. Conclusion

Security is very important when it comes to a users information. The Yubiduino helps make sure that everyone's information is secure and safe from intruders while being very easy to use and set up. Two Factor Authentication makes it harder to get into accounts because the second factor has to be physically there. The Arduino, ethernet shield, SD card, and real time clock made it possible to create this second factor to help shield intruders from getting into important accounts.

Additional references
http://arduino.cc/en/Main/ArduinoBoardDue
http://arduino.cc/en/reference/serial
http://arduino.cc/en/Reference/MouseKeyboard
http://arduino.cc/en/Reference/SD
http://arduino.cc/en/Reference/SPI
https://www.yubico.com/about/intro/yubikey/
http://tools.ietf.org/html/rfc6238
http://en.wikipedia.org/wiki/Time-based_One-time_Password_Algorithm
http://en.wikipedia.org/wiki/Multi-factor_authentication

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.