A Portable Password Vault, Version 1.3

By R. G. Sparber

Protected by Creative Commons.[1]

The Problem

You walk up to someone else's computer and want to log into one of your accounts. Since you are security-minded, your password is long and filled with all types of characters.

```
/24vdt4~VmFDFG3v!*[TtO0+{#
```

After much hunt-and-peck typing, you press Enter and are met with an error message saying your login or password is invalid. Ah, you mistyped one character. Damn! Start over.

A Solution: A Portable Password Vault

Let's try this again. You walk up to someone else's computer and want to log into one of your accounts.

You take out a device not much larger than a thumb drive and plug it into a USB port. The device has a single button on it.

After you click into the login window, you push the button. Your login appears. Then you click into the password window and press the button again. Your password[2] appears. Remove your device, and you are in.

Inside this device are only two parts: A Sparkfun Pro Micro and a pushbutton.

---

[1] This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

[2] I suggest you leave off either the first or last few characters of the password. In this way, even if someone could steal the device and know how to get to the login screen, they would not gain access. You would add these missing characters either before or after pressing the button for the second time.

A word of warning: keep this Vault in a secure place.

I didn't want to turn you off to this idea by initially showing you my proof of concept.

In this jumble are my Pro Micro, my pushbutton, and the USB cable. The clamp is my cable relief for the pushbutton wires.

Adafruit sells the Trinket MO, which is half the price and half the size yet still does the job.

The Software

The key to making the Portable Password Vault work is making the Pro Micro look like a keyboard.

At the top of my program, I have

```
#include <Keyboard.h>
```

In `setup()` I have

```
Keyboard.begin();
```

The rest of the code looks for button pushes and prints predefined strings of characters. The PC sees these prints as typing on a keyboard.

The majority of the code is dedicated to the mundane task of de-bouncing the button.

```
#include <Keyboard.h>
byte buttonPin = 9;

void setup(){
  pinMode(buttonPin, INPUT_PULLUP);
//make pin 9 an input and turn on the pullup resistor so it goes high unless connected to ground
  Keyboard.begin();
}

void loop(){
        while(debouncedRead(buttonPin) == HIGH);//wait until button is pushed for first time
         //then send the login
        Keyboard.print("my login");
        while(debouncedRead(buttonPin) == LOW);//wait for button to be released
        while(debouncedRead(buttonPin) == HIGH);//wait until button is pushed for second time
        //then send the password
        Keyboard.print("my password");
        while(debouncedRead(buttonPin) == LOW);//wait for button to be released
}


boolean debouncedRead(int Pin){
        byte debounce = 0;
        for (byte i = 0;i < 10; i++){//read pin 10 times over 20 ms and take majority vote
                debounce = digitalRead(Pin) + debounce;
                delay(2);
        }
        if (debounce <= 5)return false;
        return true;
}
```

Replace “my login” and “my password” with your values.

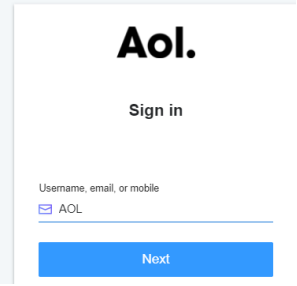
More

If you need to juggle more than one login/password pair, I can extend this idea.

At first, I assumed I would need a display. But after sleeping on it, I realized I already had one: the login window!

There are now two buttons.

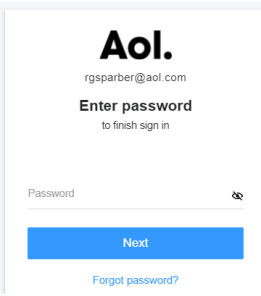
I click on the login window and push the top button to scroll back or the bottom button to scroll forward through the list of categories.



Eventually, I will see “AOL” in the login window. Then I push both buttons to select it.

When I push the top button, AOL is replaced with my login.

I press Next.



If my password is missing the first few characters, I enter them. Then I press the bottom button to inject the rest of the password. If I am missing the last few characters, I press the bottom button and then type the rest.

After getting in, I unplug the Portable Password Vault or move on to another site. The buttons again move me up and down the list of categories.

Each of the categories with their login and password is provisioned before the program is compiled. This makes it more difficult for someone to extract the data. It also avoids the complexity of creating an input screen.

I have not written the code because I currently do not need it. That might change in the future.

Acknowledgment
Thanks to Andrew Ayers for reminding me of the security aspects of this design.

I welcome your comments and questions.

If you want me to contact you each time I publish an article, email me with "Subscribe" in the subject line. In the body of the email, please tell me if you are interested in metalworking, software plus electronics, kayaking, and/or the Lectric XP eBike so I can put you on the right distribution list.

If you are on a list and have had enough, email me "Unsubscribe" in the subject line. No hard feelings.

Rick Sparber
Rgsparber.ha@gmail.com
Rick.Sparber.org