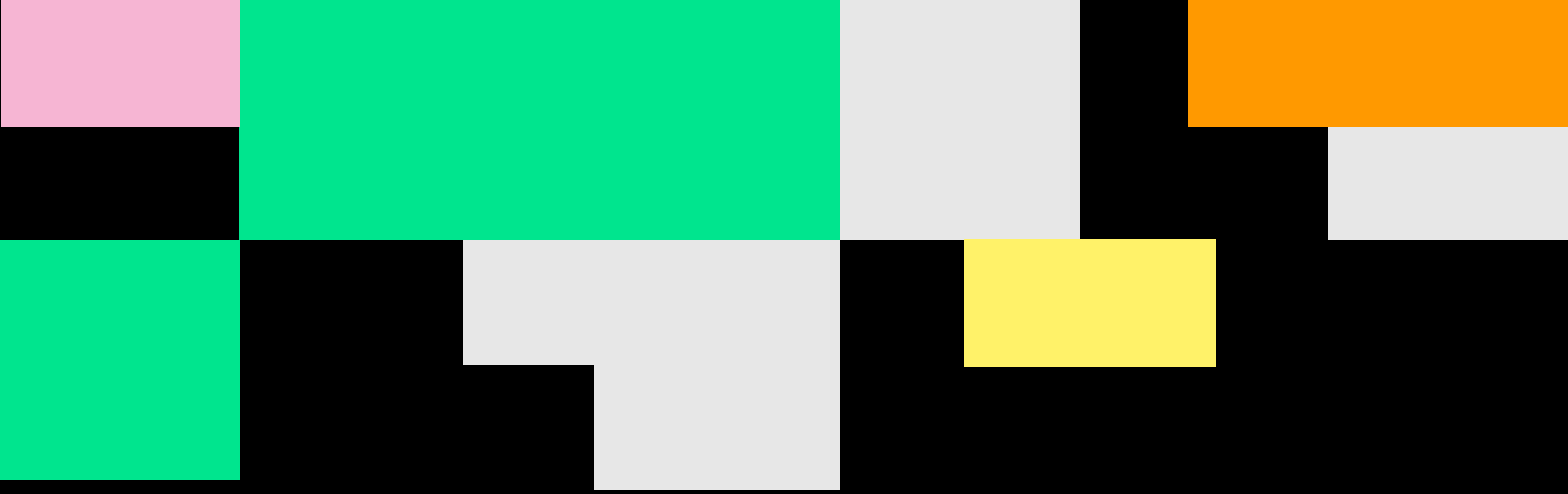




Power Analysis Attacks 101: From Waveform to a Private Key

Igor Brkić

WHOAMI

- Hardware & software developer
- Custom hardware development, prototyping, R&D
- <https://hyperglitch.com>

How it started

The Hardware Hacking Handbook

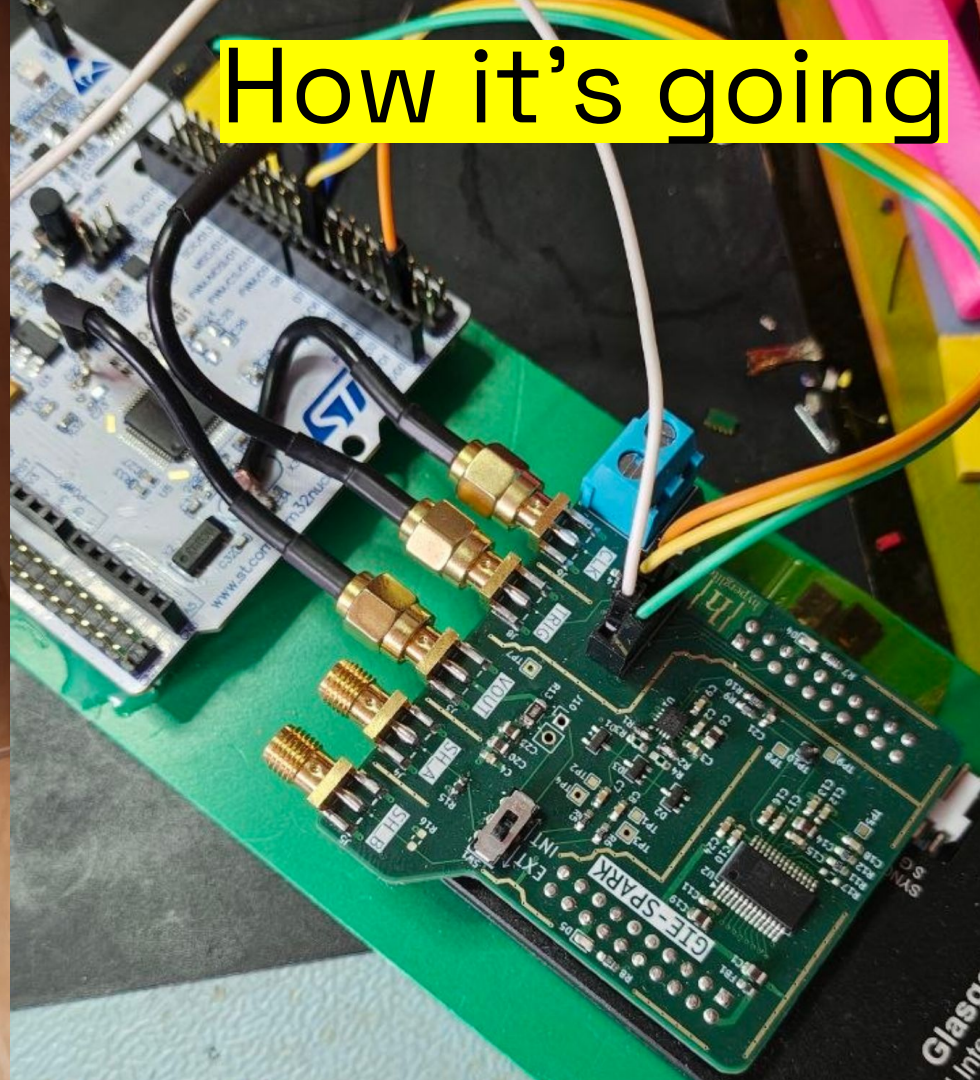
*Breaking Embedded Security
with Hardware Attacks*



Jasper van Woudenberg and Colin O'Flynn



How it's going



Side-channel attacks

In [computer security](#), a **side-channel attack** is a type of security exploit that uses information inadvertently leaked by a system—such as timing, power consumption, or electromagnetic or [acoustic emissions](#)—to gain unauthorized access to sensitive information. These attacks differ from those targeting flaws in the design of [cryptographic protocols](#) or [algorithms](#) (notwithstanding the fact that [cryptanalysis](#) may identify vulnerabilities relevant to both types of attacks).

(https://en.wikipedia.org/wiki/Side-channel_attack)

Side-channel attacks

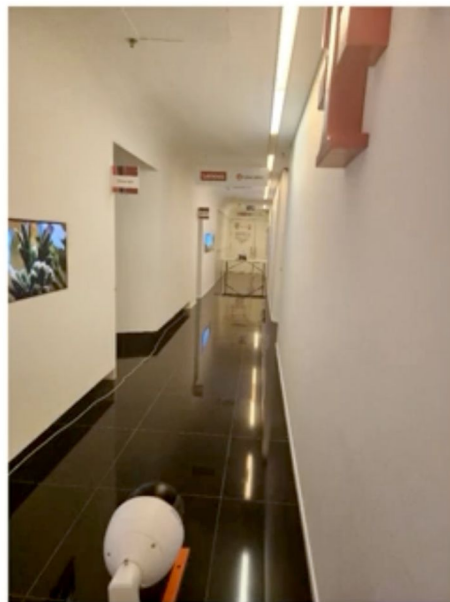
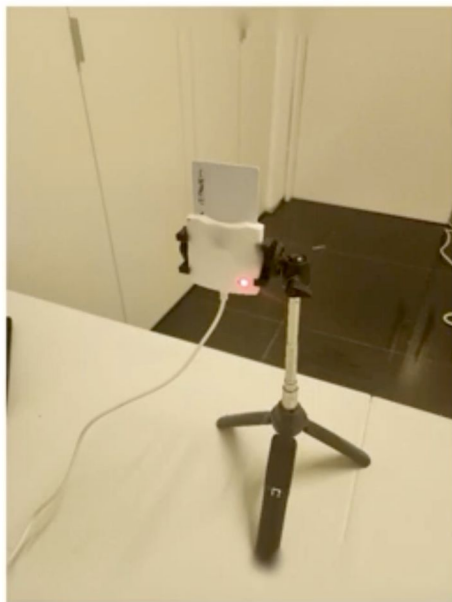
Fun examples

- **VanEck phreaking**
Electromagnetic radiation from video display units:
An eavesdropping risk?, 1985
- **Keyboard acoustic emanations**
Dmitri Asonov and Rakesh Agrawal, 2004
- **The Visual Microphone**
Davis et al., 2014

Hackers can steal cryptographic keys by video-recording power LEDs 60 feet away

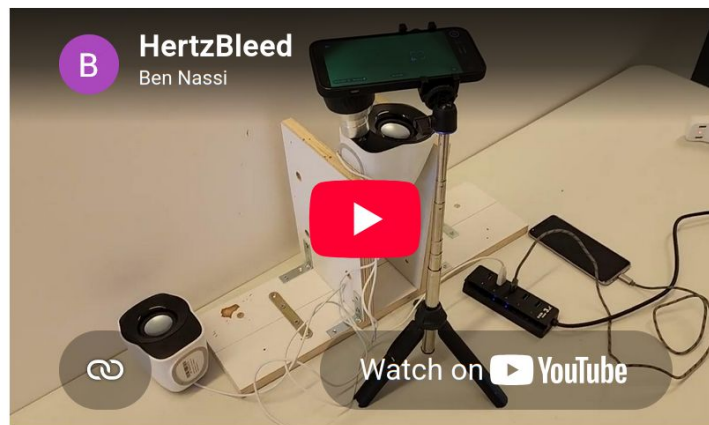
Key-leaking side channels are a fact of life. Now they can be done by video-recording power LEDs.

DAN GOODIN · JUN 13, 2023 3:30 PM · 142



Recovering SIKE Key from Samsung Galaxy S8 using Video Footage of the Logitech Z120 USB Speakers' Power LED obtained using iPhone 13 Pro Max

Recording the power LED of the connected USB speakers



One of the videos that we used to recover the key

<https://www.nassiben.com/video-based-crypta>

Video-Based Cryptanalysis: Extracting Cryptographic Keys from Video Footage of a Device's Power LED Captured by Standard Video Cameras

Ben Nassi^{1,2}, Etay Iluz², Or Cohen², Ofek Vayner², Dudi Nassi², Boris Zadov², and Yuval Elovici²

¹Cornell Tech, New York, USA

²Ben-Gurion University of the Negev, Beersheba, Israel

bn267@cornell.edu, {nassib, etayil, ora2, ofekvay, nassid, zadov}@post.bgu.ac.il, elovici@bgu.ac.il

Website - <https://www.nassiben.com/video-based-crypta>

Abstract—In this paper, we present *video-based cryptanalysis*, a new method used to recover secret keys from a device by analyzing video footage of a device's power LED. We show that cryptographic computations performed by the CPU change the power consumption of the device which affects the brightness of

methods employ equipment like scopes to obtain power traces (e.g., [1, 2]), software-defined radio (SDR) and probes to capture electromagnetic radiation (EMR) traces (e.g., [3–9]), photodiodes and dedicated photon detectors to capture optical traces (e.g., [10–14]) as well as ultra-

<https://www.computer.org/csdl/proceedings-article/sp/2024/313000a163/1Ub24CjWdlm>

(e.g., an iPhone 13's camera or Internet-connected security

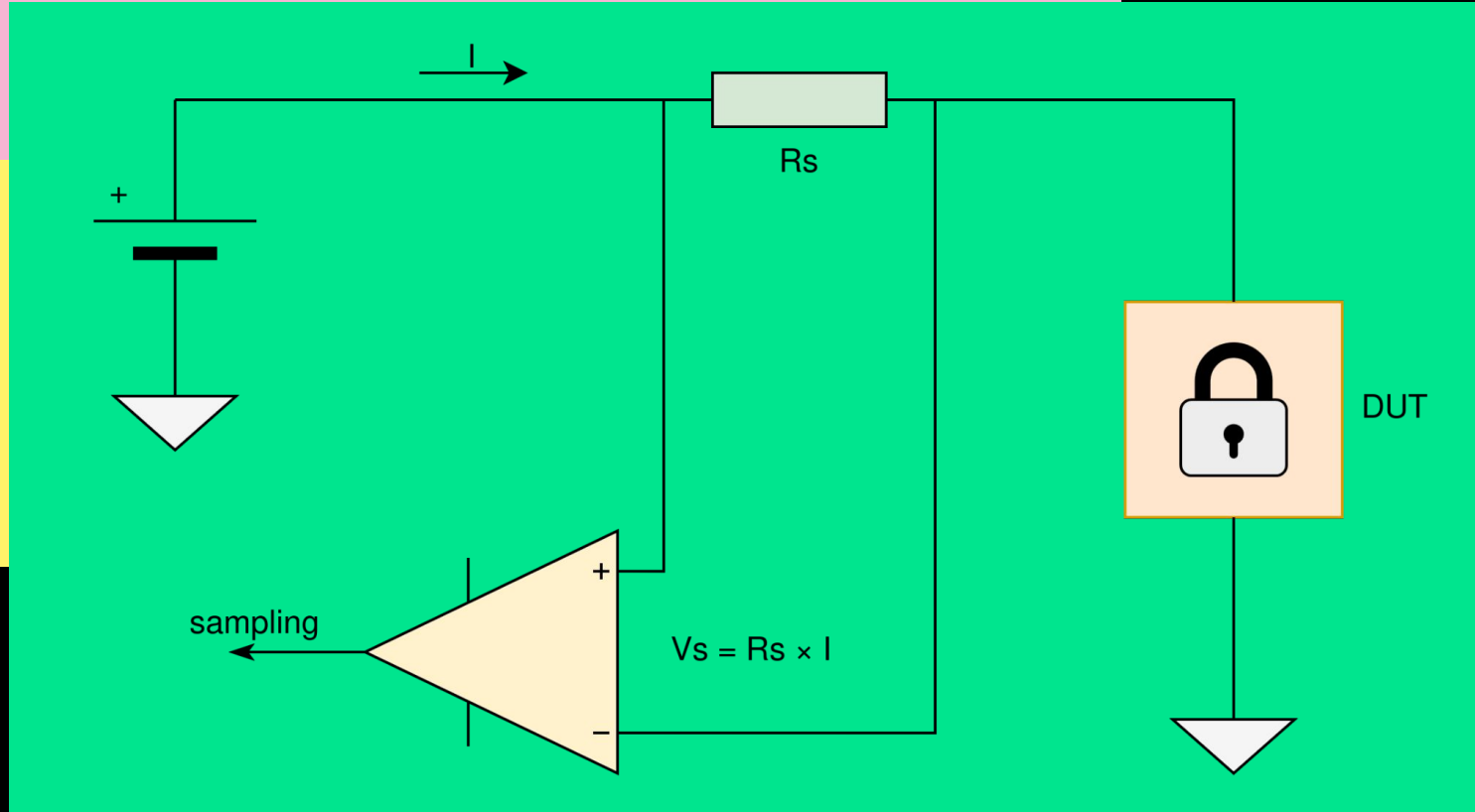
high-end hardware poses to secret keys, it is

Power Analysis

- CPU is made from transistors (MOSFET pairs)
- they are being switched to form 1s and 0s, the numbers
- by being switched they are using more or less power
- CPU is leaking information!!!

⇒ How granular can we go?

Power Analysis



Power A

You're a unit of power, Harry.

**I'm a
Watt?**

Simple Power Analysis

- Base for timing attacks
- Base for glitching
- “Visual” secret extraction

```

//Get password
my_puts(c: "Please enter password to continue: ");
my_read(buf: passwd, len: 32);

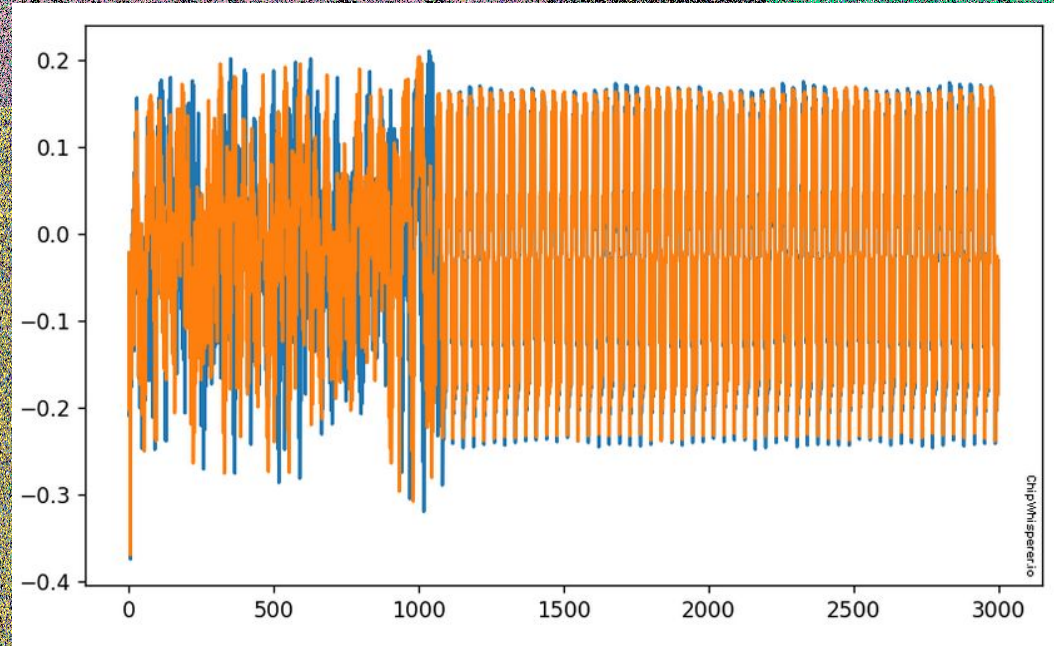
uint8_t passbad = 0;

trigger_high();

for(uint8_t i = 0; i < sizeof(correct_passwd); i++){
    if (correct_passwd[i] != passwd[i]){
        passbad = 1;
        break;
    }
}

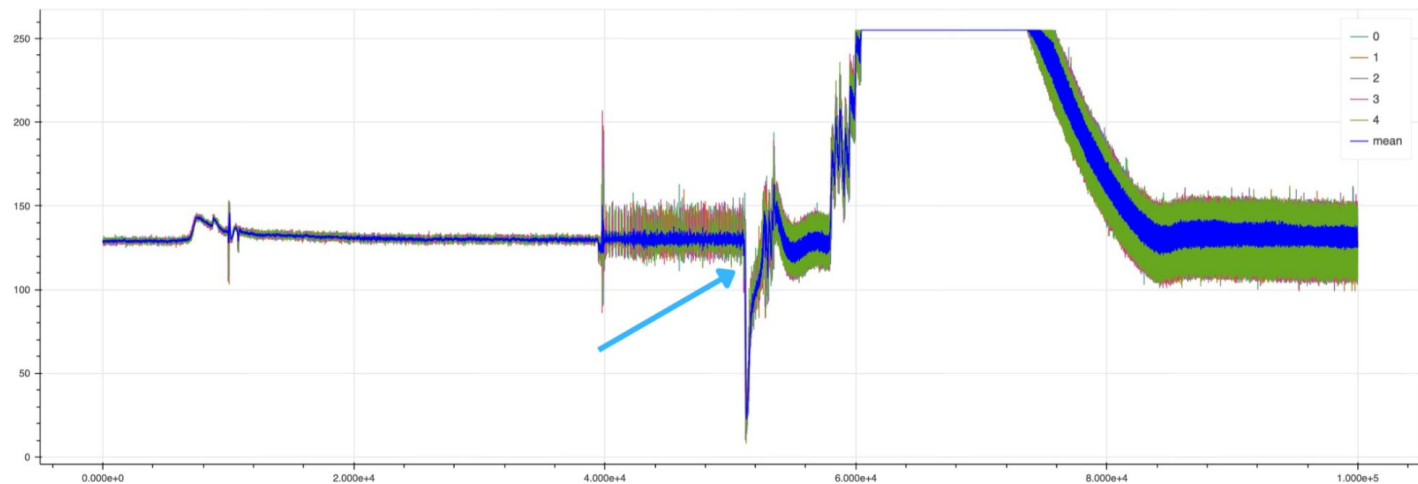
if (passbad){
    //Stop them fancy timing attacks
    int wait = 1;
    for(volatile int i = 0; i < wait; i++){
        ;
    }
    delay_2_ms();
    delay_2_ms();
    my_puts(c: "PASSWORD FAIL\n");
    led_error(a: 1);
} else {
    my_puts(c: "Access granted, Welcome!\n");
    led_ok(a: 1);
}

```



Simple Power Analysis

F4 RDP DOWNGRADE: POWER ANALYSIS



<https://grandideastudio.com/portfolio/security/stm32-fault-injection/>

Simple Power Analysis

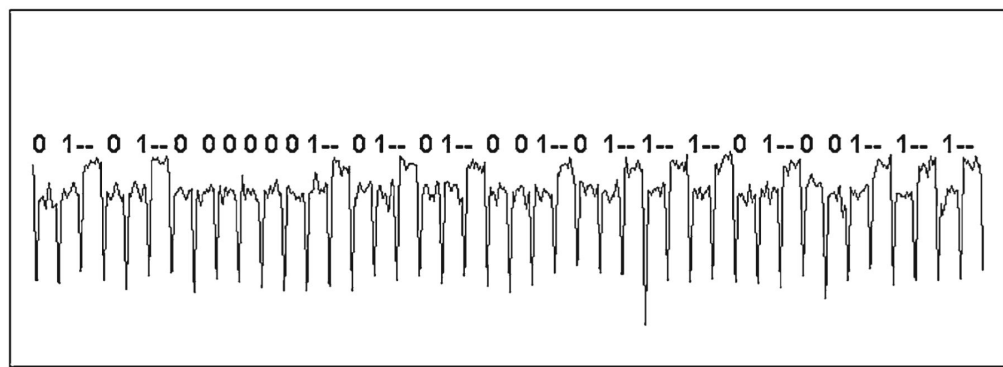


Fig. 11 SPA leaks from an RSA implementation

• “Visual” secret extraction

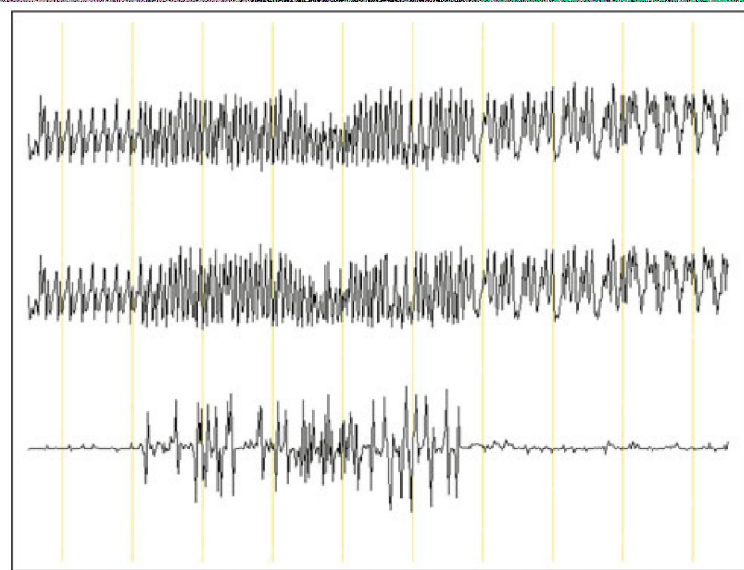


Fig. 14 Trace pair comparison of a permutation with different inputs

But what if the order of
operations doesn't depend
on the secret material?

DPA

(differential power analysis)

- Power consumption correlated to the actual data being manipulated – consumption is different for 1 and 0
- Much smaller changes, overshadowed by measurement errors and noise
- Attacking a single bit at a time:
 - record N traces with different inputs
 - for a key/byte guess predict one bit
 - split into two groups (predicted “1”s and “0”s)
 - average each group and subtract
 - was the guess correct? Repeat for other 255 key guesses

CPA

(correlation power analysis)

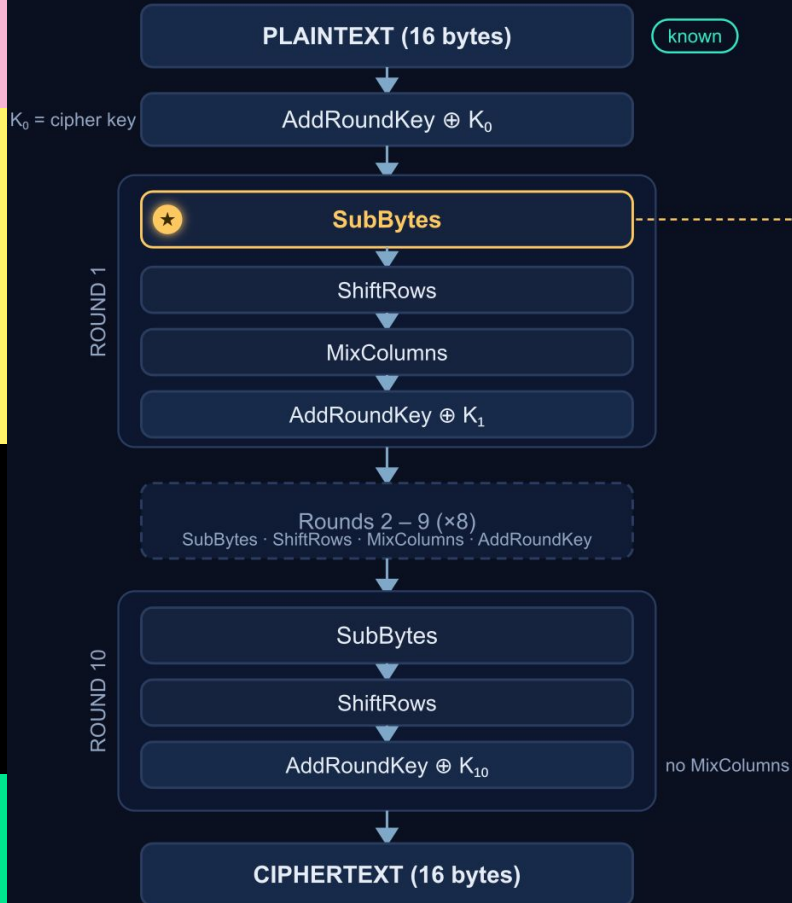
- Introduce better leakage model
 - Hamming weight: $HW(0b00110011) == 4$
 - Hamming distance: $HD(0b1010 \rightarrow 0b1110) == 1$
- “split into two groups” => “measure correlation”
 - record N traces with different inputs
 - predict a whole byte and get its HW
 - does the prediction correlate with power trace? (all traces)
(Pearson's correlation)

AES-128 encryption

K_0 = cipher key



AES-128 encryption



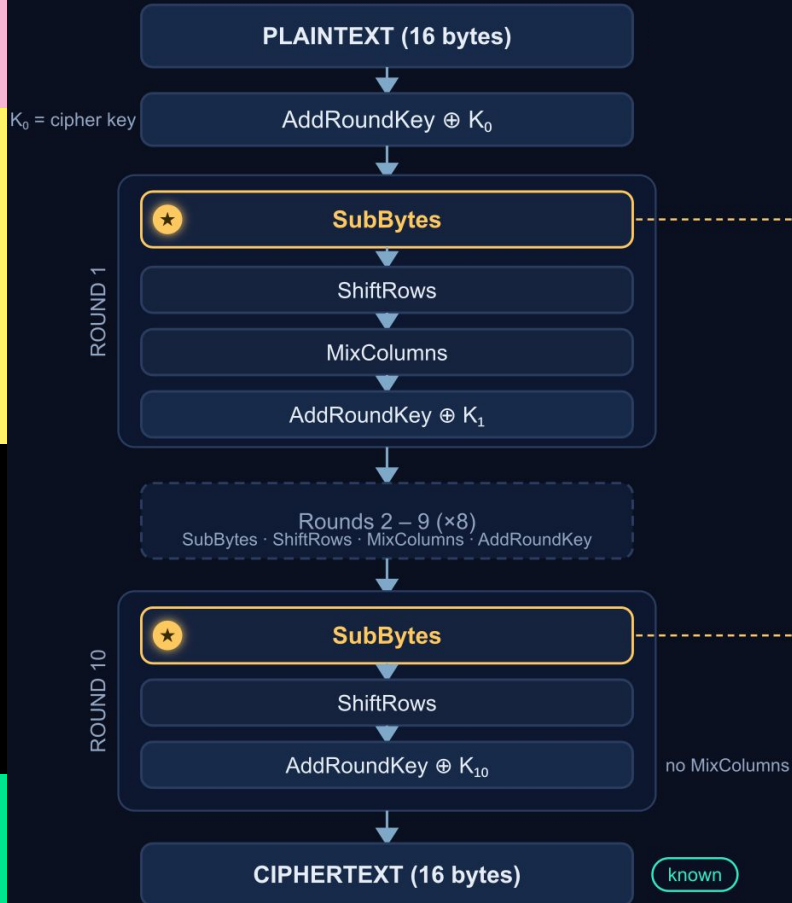
① First round · known plaintext

$HW(Sbox(pt \oplus K_0))$

knownplaintextguess K_0 byte (256)

recover K_0 = the key directly

AES-128 encryption



① First round · known plaintext

$HW(Sbox(pt \oplus K_0))$

knownplaintextguess K_0 byte (256)

recover K_0 = the key directly

② Last round · known ciphertext

$HW(InvSbox(ct \oplus K_{10}))$

knownciphertextguess K_{10} byte (256)

recover $K_{10} \rightarrow$ invert key schedule $\rightarrow$ key

mind ShiftRows

How to do it?

- Every change synced to CPU clock
- Async sampling
 - 5-7 x clock frequency
 - Requires resampling/aligning
- Sync sampling
 - Everything aligned
 - Adjust offset

Available tools

- ChipWhisperer

<https://github.com/newaetech/chipwhisperer-jupyter>

- Oscilloscope + LNA

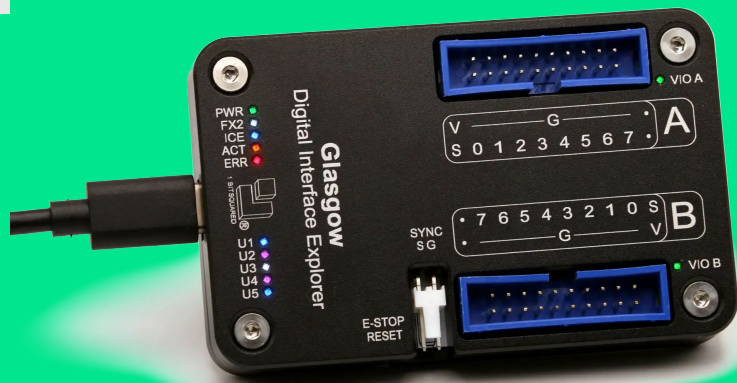
- RedPitaya

<https://github.com/fdurvaux/sca-redpitaya>

- Glasgow + something?

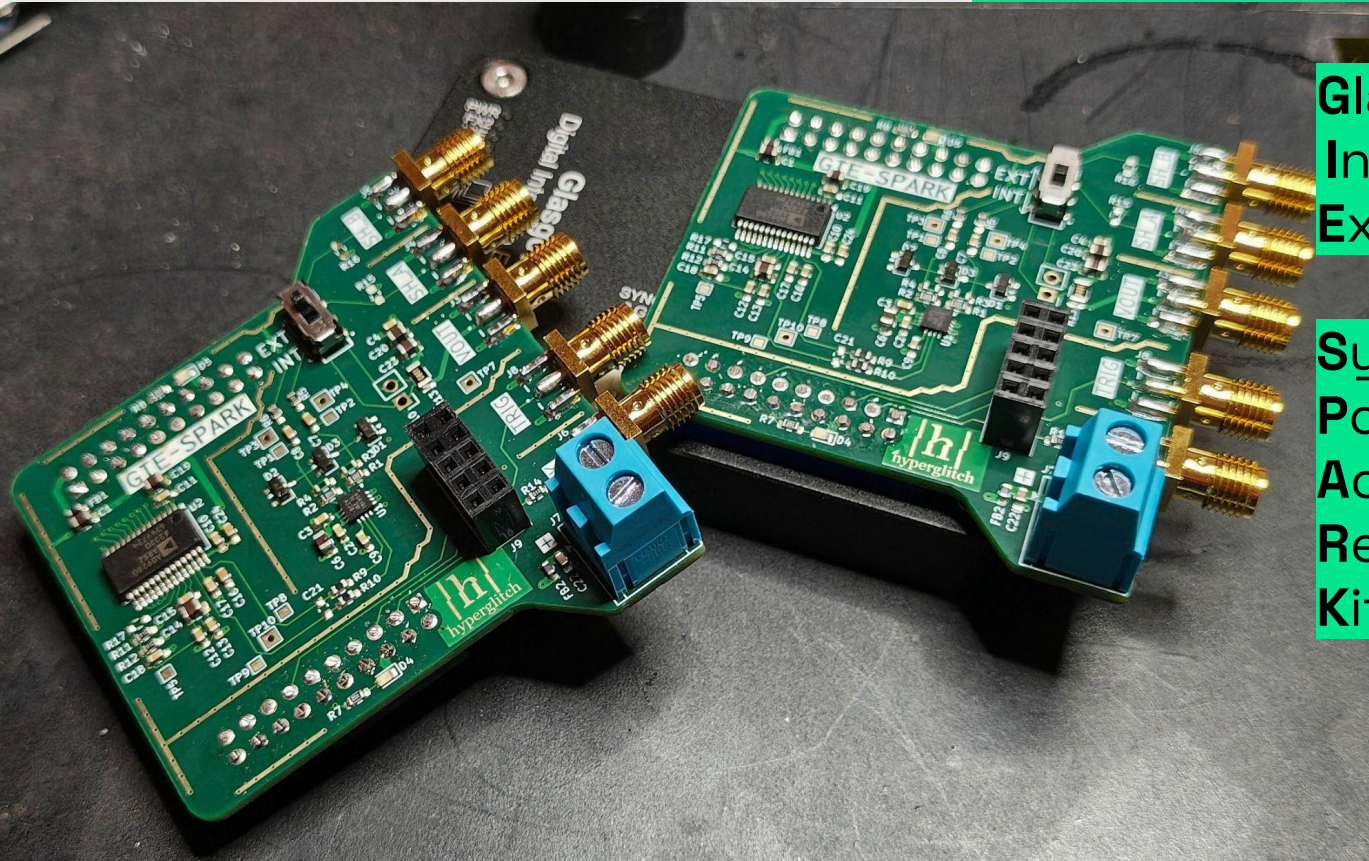
} synchronized

} free-running



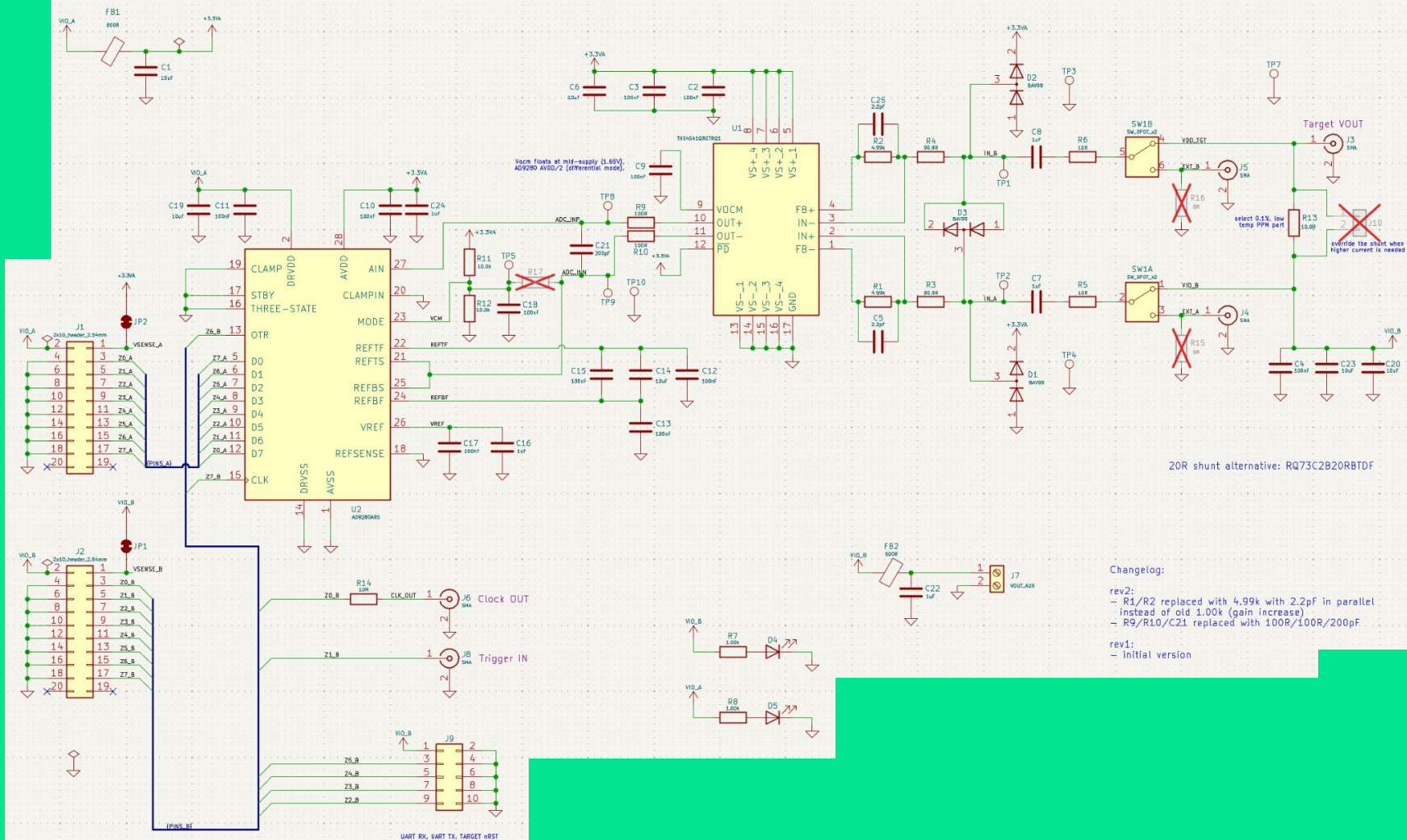
<https://glasgow-embedded.org>

I made a tool!



Glasgow
Interface
Explorer

Synchronous
Power
Acquisition &
Reconnaissance
Kit



20R shunt alternative: RQ73C2B20RBTDF

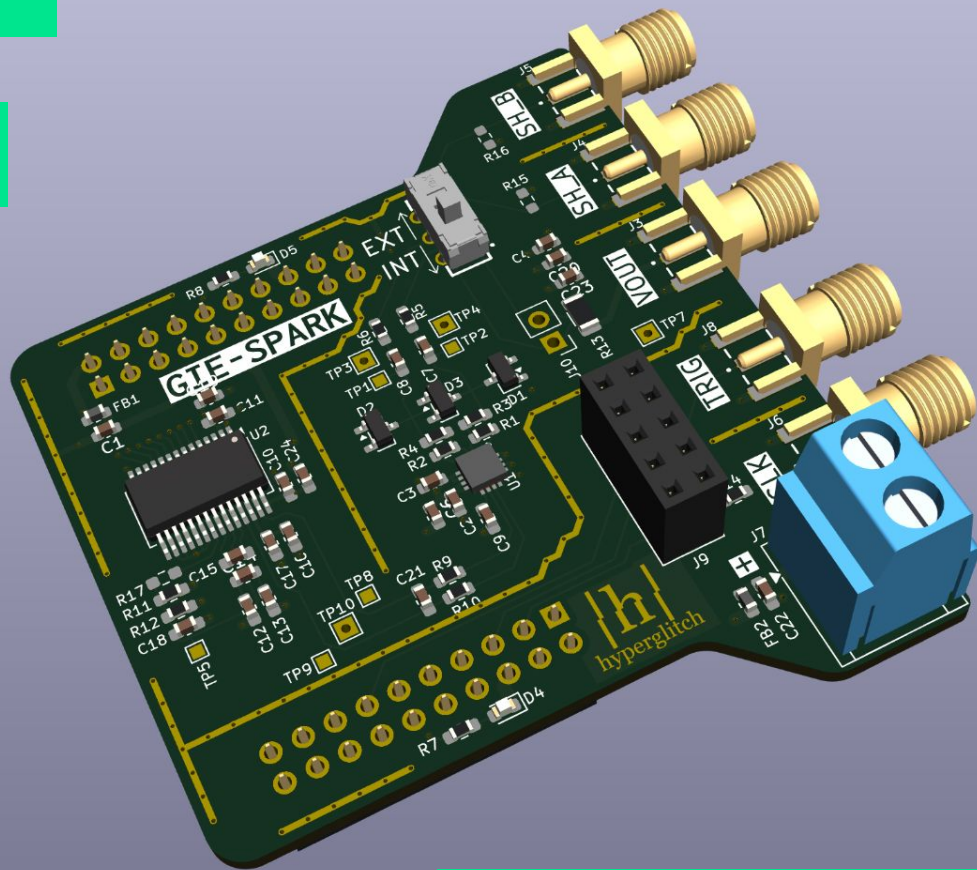
Changelog:

rev2:

- R1/R2 replaced with 4.99k with 2.2pF in parallel instead of old 1.00k (gain increase)
- R9/R10/C21 replaced with 100R/100R/200pF

```
rev1:
```

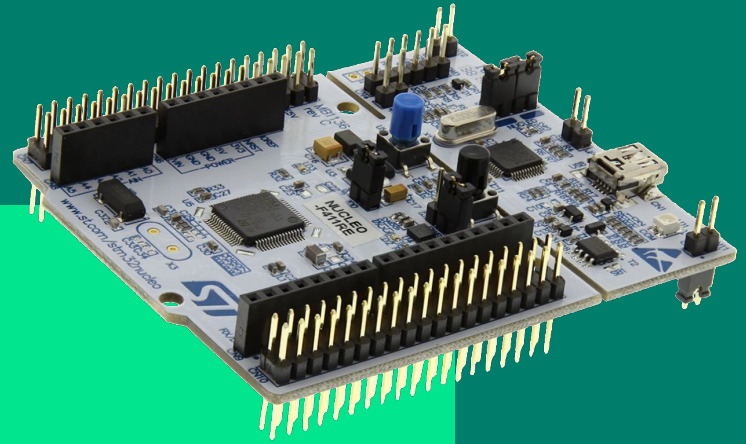
- initial version



<https://gitlab.com/hyperglitch/gie-spark>

Let's simplify things

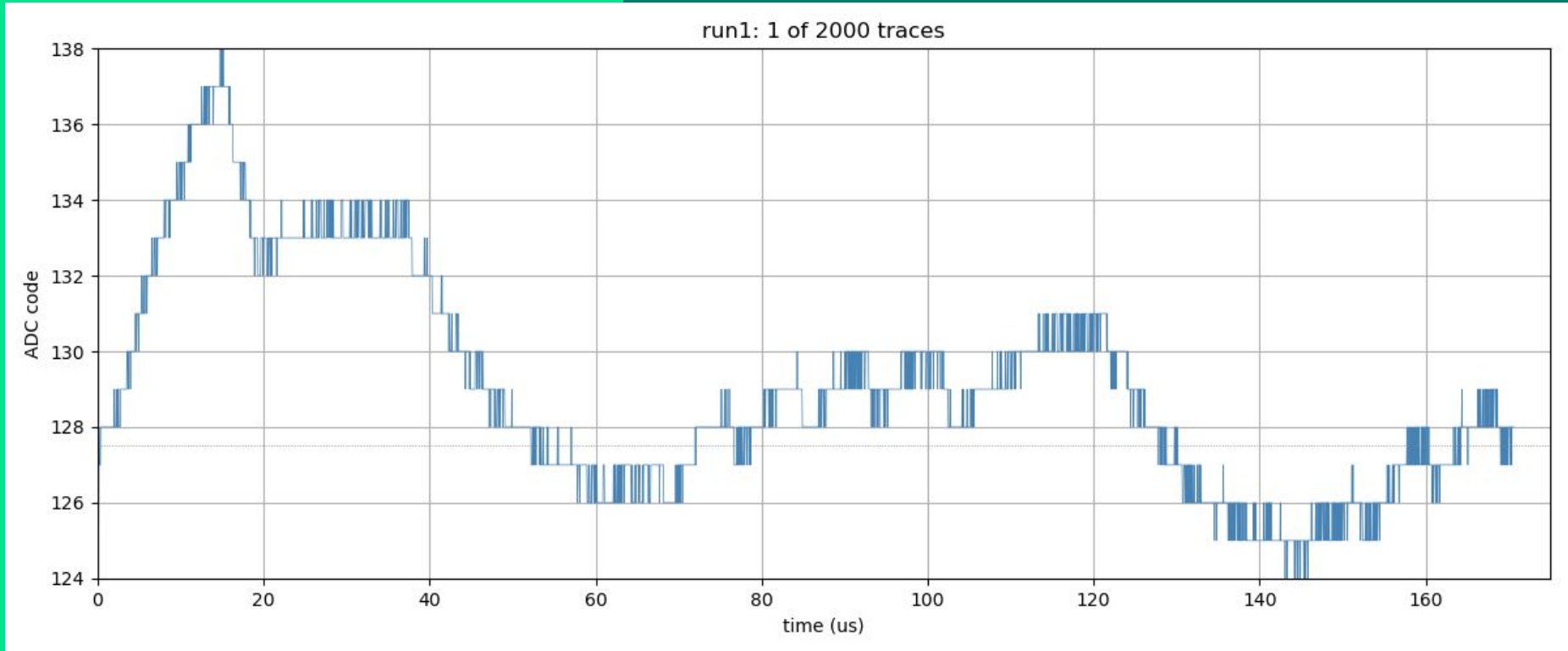
- ST NUCLEO-F303RE
- tinyAES / mbedTLS
- 8MHz clock from SPARK board
- Sampled at 24MHz
- ChipWhisperer tutorial firmware



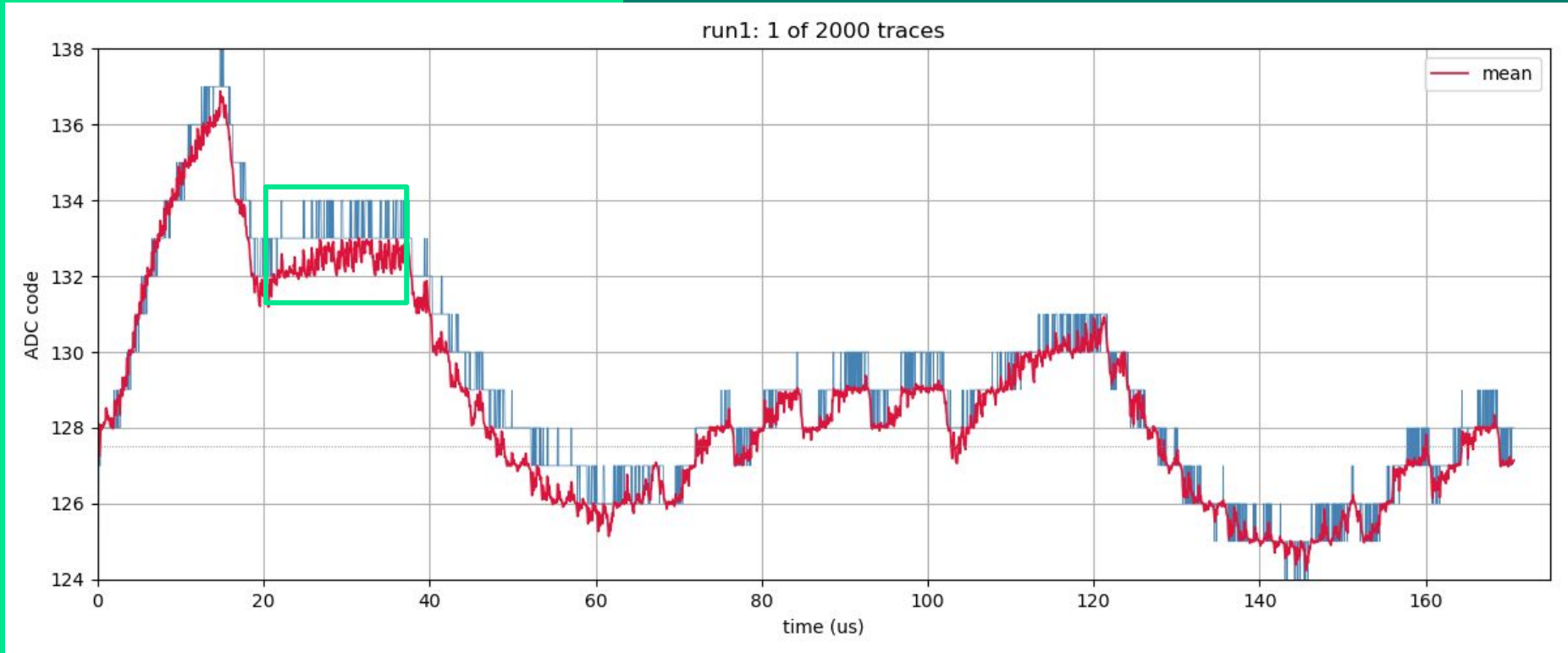
Gateway / Software

- <https://gitlab.com/hyperglitch/gie-spark-applet>
- SPARK board control + UART
- Scripts for checking the traces and key extraction

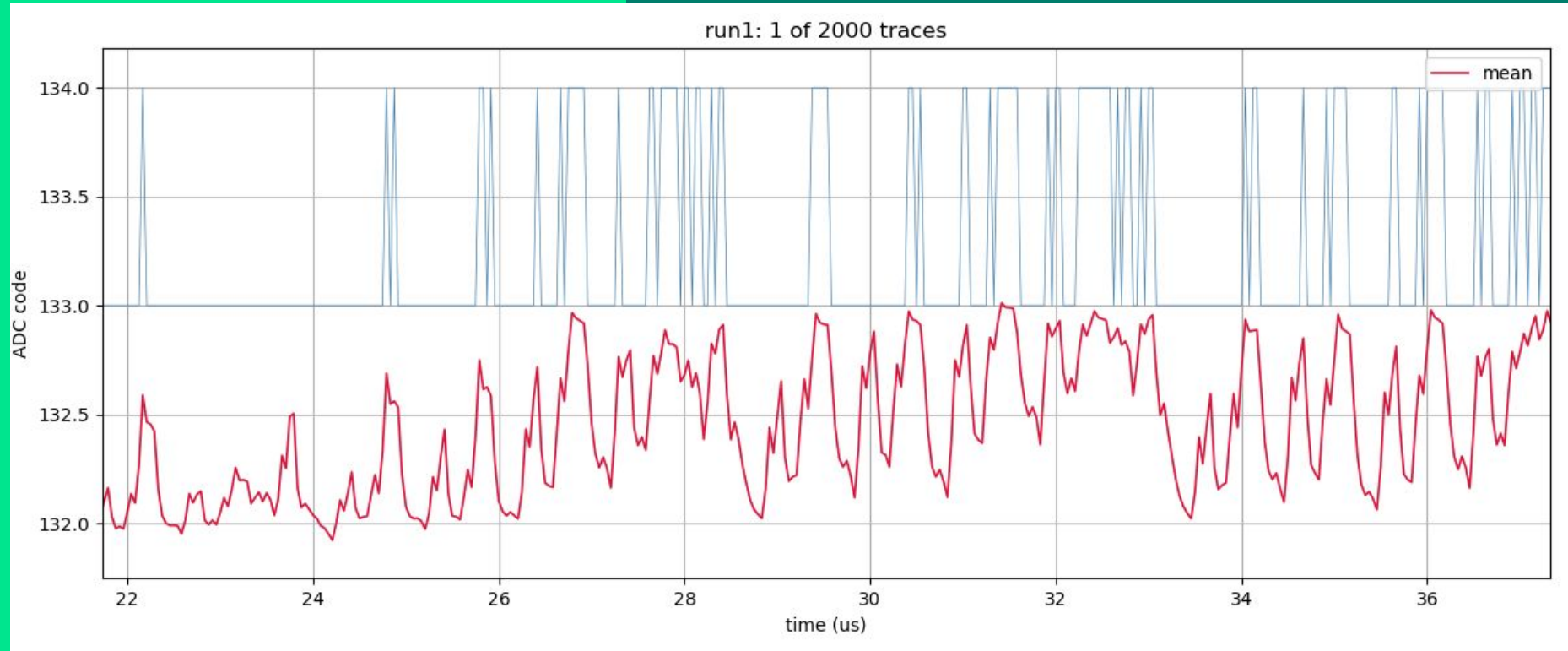
First results and the power of averaging



First results and the power of averaging



First results and the power of averaging



First res

```
> python tools/cpa.py run1 --window 794:1944  
attacking samples 794:1944
```

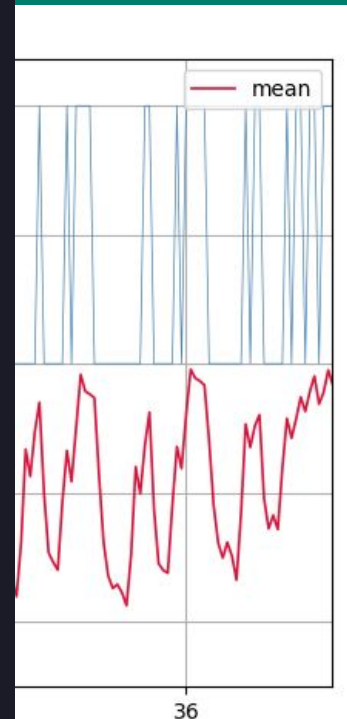
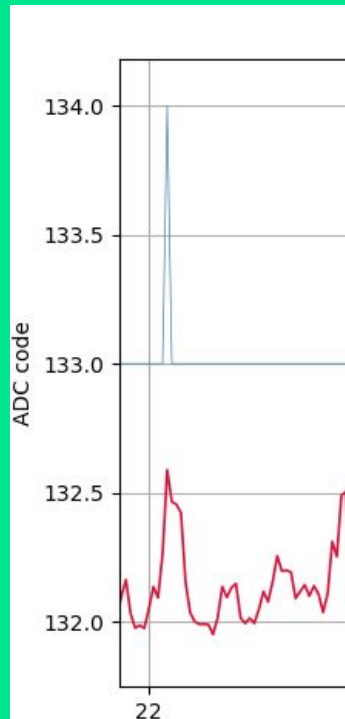
2000 traces x 1150 samples

byte	guess	r	2nd	margin	sample	correct key
0	2b	0.124	21	0.022	994	ok
1	c4	0.116	7e	0.008	1561	7e at rank 2, r=0.108
2	15	0.133	01	0.026	1604	ok
3	bf	0.138	59	0.039	1481	16 at rank 46, r=0.082
4	28	0.117	74	0.004	1038	ok
5	ae	0.126	b3	0.022	1744	ok
6	d2	0.151	72	0.043	1585	ok
7	a6	0.124	31	0.020	1603	ok
8	ff	0.105	8d	0.001	1050	ab at rank 95, r=0.078
9	9c	0.114	00	0.006	850	f7 at rank 5, r=0.103
10	15	0.163	84	0.057	1635	ok
11	88	0.127	94	0.017	1602	ok
12	4a	0.105	d9	0.006	1271	09 at rank 13, r=0.089
13	cf	0.140	7c	0.032	1627	ok
14	4f	0.157	a7	0.046	1591	ok
15	3c	0.146	26	0.028	1600	ok

correct key ranked 1st on 11 bytes, in the top 3 on 12, top 10 on 13 of 16
median rank 1 out of 256 guesses

recovered 2bc415bf28aed2a6ff9c15884acf4f3c
actual 2b7e151628aed2a6abf7158809cf4f3c
11/16 bytes correct

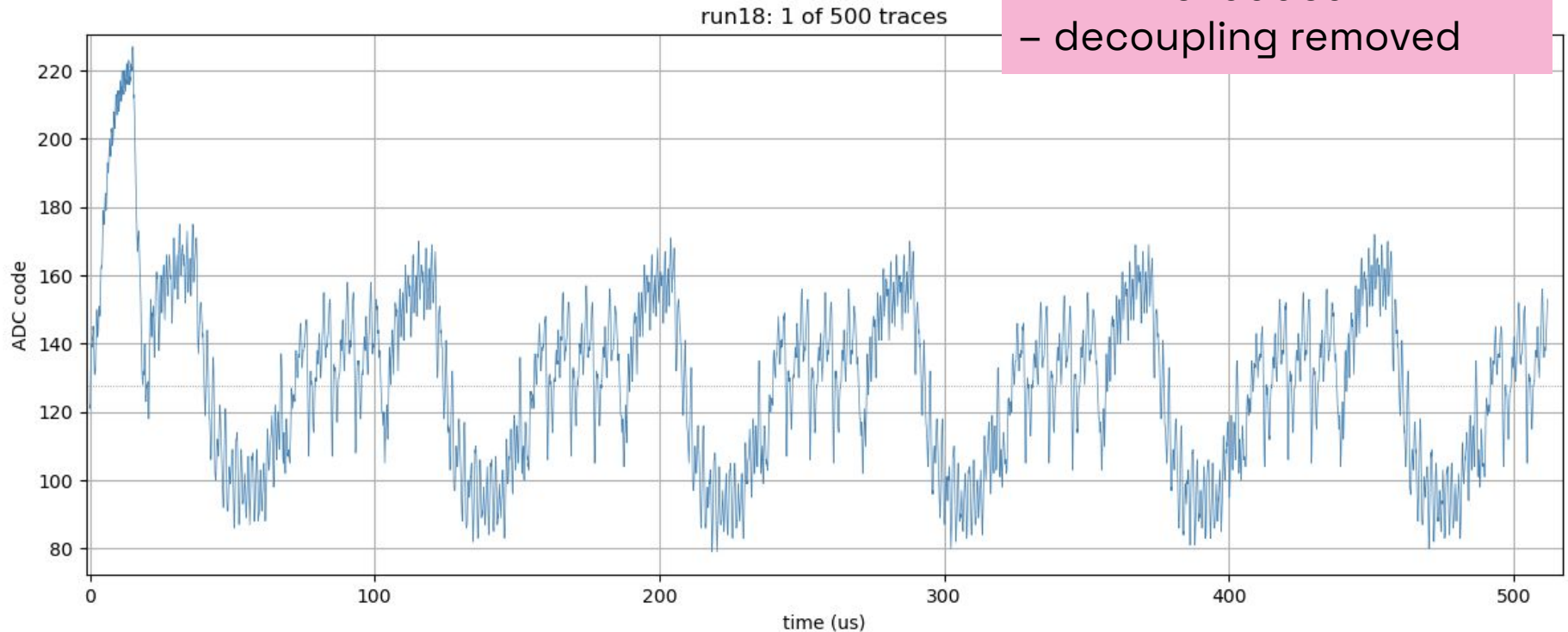
raging



time (us)

After some modifications

- shunt increased to 20Ω
- gain increased to 50
- buffer size increased
- AA filter added
- decoupling removed



```
> python tools/cpa.py run18 --traces 500 --window 0:4096  
attacking samples 0:4096
```

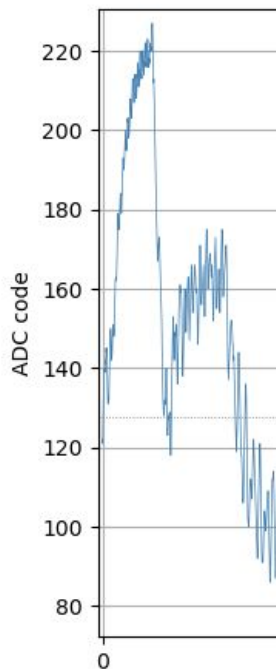
After s

500 traces x 4096 samples

byte	guess	r	2nd	margin	sample	correct key
0	2b	0.331	b6	0.121	994	ok
1	7e	0.298	54	0.086	1573	ok
2	15	0.225	88	0.008	2445	ok
3	16	0.285	7d	0.091	1433	ok
4	28	0.320	a1	0.119	1066	ok
5	ae	0.314	61	0.106	1550	ok
6	d2	0.320	50	0.120	1287	ok
7	a6	0.341	7e	0.151	1466	ok
8	ab	0.259	56	0.052	1093	ok
9	f7	0.326	57	0.120	1557	ok
10	15	0.342	83	0.140	1574	ok
11	88	0.275	bd	0.063	1448	ok
12	09	0.236	d3	0.037	1086	ok
13	cf	0.397	8a	0.202	1561	ok
14	4f	0.381	be	0.170	1586	ok
15	3c	0.291	d8	0.018	1638	ok

correct key ranked 1st on 16 bytes, in the top 3 on 16, top 10 on 16 of 16
median rank 1 out of 256 guesses

recovered 2b7e151628aed2a6abf7158809cf4f3c
actual 2b7e151628aed2a6abf7158809cf4f3c
16/16 bytes correct



d to 20Ω
to 50
eased
moved



The background is composed of several overlapping rectangular blocks of color. A large teal block covers the right side and top. A yellow block is positioned in the center, containing the text. Pink blocks are located in the top right and bottom left corners. The overall style is modern and minimalist.

**DEMO
TIME!**

Countermeasures

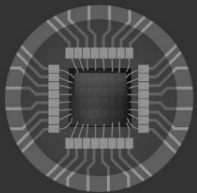
- Masking – split intermediates into parts
- Hiding – add noise, randomize operations, add random delays

Other implementations

- Hardware implementation – harder but still possible (HD leakage model)

**DOES THIS
STILL
WORK?**





Courk's Blog

about

contact

misc



Breaking the Flash Encryption Feature of Espressif's Parts

Posted on Mon 08 January 2024 in [Projects](#)

- **Breaking the Flash Encryption Feature of Espressif's Parts**

- Introduction
- Side-Channel Attack Basics
 - General Idea
 - Power Consumption Models
 - Hamming Weight Model
 - Example: AES First Round Encryption
 - Correlation Power Analysis
- Espressif's Flash Encryption Feature
 - Overview
 - High-Level Attack Plan
- Hardware Platform
 - Goals
 - Overview
 - Block Diagram
 - Key Features
 - Fake External Flash
 - Current Measurement
 - Clock Control
 - Temperature Control
 - Software Support
 - Firmware & Gateware
 - Host Software
- Attacking the ESP32
 - Encryption Method Overview
 - Experimental Results
 - Trace Capture

<https://courk.cc/breaking-flash-encryption-of-espressif-parts>



Thank you!

https://hglt.ch/dpa_balcccon.pdf

Igor Brkić
@1g0rb
<https://hyperglitch.com>