HWallet
The simplest Bitcoin hardware wallet
Vulnerabilities in hardware wallets

**Ledger**

While the software on the SE can be attested to, the MCU is a non-secure chip and its firmware can be replaced by an attacker

**TREZOR**

https://blog.trezor.io/fixing-physical-memory-access-issue-in-trezor-2b9b46bb4522
...an attacker with physical access to a TREZOR device could have created a custom firmware which extracts the seed from the RAM of the device.
Hardware wallets

<table>
<thead>
<tr>
<th>Device</th>
<th>MCU</th>
<th>Secure Element</th>
<th>Secure MCU</th>
<th>TRNG</th>
<th>SHA256</th>
<th>secp256k1</th>
<th>Open Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>TREZOR</td>
<td>STM32F205</td>
<td>OLED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>ST31H320</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ledger</td>
<td>STM32F042</td>
<td>OLED</td>
<td>STM32F042</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>ST31H320</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Coldcard</td>
<td>STM32L475</td>
<td>OLED</td>
<td>ATECC508A</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>HWallet</td>
<td>NXP K20</td>
<td>OLED</td>
<td>NXP K(L)82</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

nemanja@hacke.rs
Don't roll your own crypto!

YOU DON'T ROLL YOUR OWN CRYPTO

IF YOU USE HARDWARE CRYPTO
Code size comparison

```
git clone https://github.com/{PRODUCT}/{FIRMWARE} --recurse-submodules

cd {FIRMWARE}
wq -l `find . -name "*.c" -o -name "*.h"`
```

<table>
<thead>
<tr>
<th></th>
<th>ColdCard</th>
<th>Ledger</th>
<th>TREZOR</th>
<th>keepkey</th>
<th>HWallet</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2.5M+</td>
<td>346k+</td>
<td>162k+</td>
<td>122k+</td>
<td>~4k</td>
</tr>
</tbody>
</table>

nemanja@hacke.rs
Code layers

To Communication MCU

- Tx/Rx speed fixed to 115200 bps
- SPI bus clocked at 1 MHz

NXP K82

UART  CRC  SPI  GPIO

OLED

https://gitlab.com/nemanjan/hwallet

LTC

256-bit operations

\[ A = A \mod N \]
\[ B = (1/A) \mod N \]
\[ A = (A+B) \mod N \]
\[ A = (A*B) \mod N \]

\[ y^2 = x^3 + A[3] \cdot x + B[0] \]
\[ (B[1], B[2]) = E \cdot (A[0], A[1]) \]

ECDSA: secp256k1

SHA256D

Bitcoin TX

TX Signature

nonce

nemanja@hacke.rs
typedef struct {
    uint16_t type;
    uint16_t length;
    uint8_t data[32];
    uint32_t crc;
} Packet;

Packet_Send();
Packet_Receive();

typedef struct {
    SPI* spi;
    GPIOx* dcGpio;
    GPIOx* rstGpio;
    uint8_t dcPin;
    uint8_t rstPin;
    uint8_t buffer[];
} OLED;

OLED_WriteRow();
OLED_Clear();

OLED_Clear();

typedef struct {
    uint8_t num[32];
    uint8_t len;
} Bignum;

CRYPTO_Bignum_Init();
CRYPTO_Bignum_Mod();
CRYPTO_Bignum_Div();
CRYPTO_Bignum_Sub();
CRYPTO_Bignum_IsNull();

B' = (1/B) mod N
A' = A – A mod B
(A/B) mod N = (A'B') mod N

N - a large prime, larger than any A or B, e.g. p from secp256k1

https://gitlab.com/nemanjan/hwallet

nemanja@hacke.rs
```c
while(1) {
    Packet msg;
    PACKET_Receive(&msg);
    switch(PACKET_MODULE(msg.type)) {
        case PACKET_BITCOIN:
            Bitcoin_Process(&msg);
            ...
    }
}
```

https://gitlab.com/nemanjan/hwallet
void Bitcoin_Process(Packet* msg) {
    switch(PACKET_FUNC(msg->type)) {
        case BITCOIN_FUNC_INIT_TX:
            Bitcoin.Tx_Init();
            ...
    }
}

void Bitcoin_Process(Packet* msg) {
    switch(PACKET_FUNC(msg->type)) {
        case BITCOIN_FUNC_INIT_TX:
            Bitcoin.Tx_Init();
            ...
    }
}
What's next?

**FIDO U2F**

- Google
- Dropbox
- WebAuthn
- CTAP

**Recovery seed**

- Entropy 128-512 bit
- BIP-32
- BIP-44
- m
- m/0
- m/44'
- m/44'/0'
- 0' – BTC
- 60' – ETH
- 144' – XRP

**More cryptocurrencies**

- BIP-32
- BIP-39
- BIP-44

**Anti-Tamper**

- NXP K(L)81
- NXP K(L)82
- Comm MCU

- nRF52840

**Tools**

- Bitcoin
- Ethereum
- Dash

---

nemanja@hacke.rs
Questions?