Rev1

swissbit®

Secrets Unveiled – What Attackers Find in Embedded Systems

Roland Marx

12.12.2024

Store. Secure. Trust.

Vorstellung

```
roland@presentation:~$ whoami
```

Roland Marx
Senior Product Manager – Embedded IoT Solutions
Freelance Cybersecurity Consultant & Trainer

Focus Topics

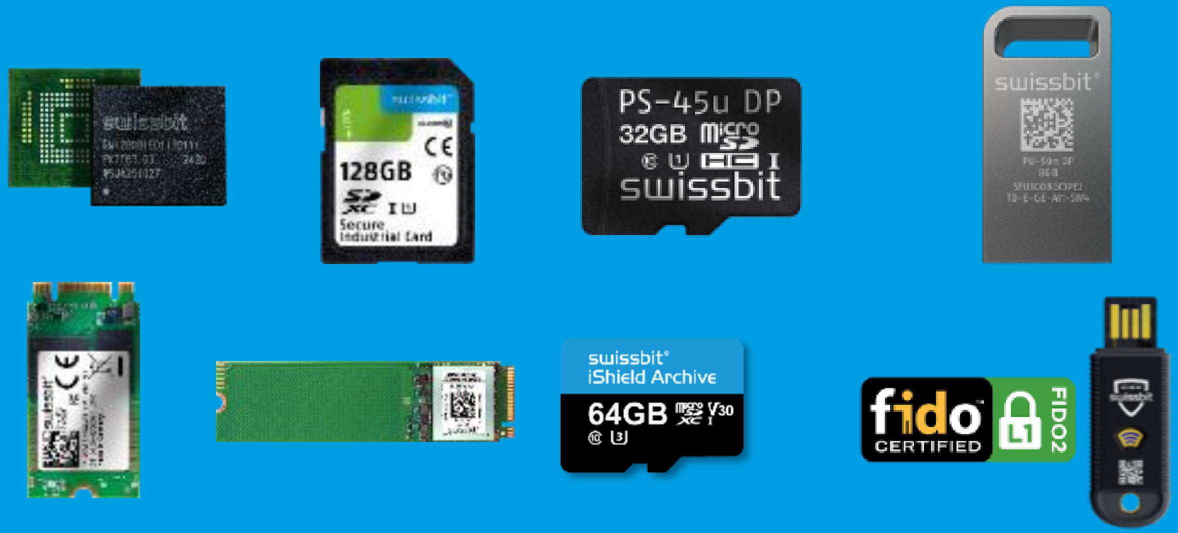
- IoT System Architecture and Design
- IoT Lifecycle & Device Management
- Embedded & IoT Security

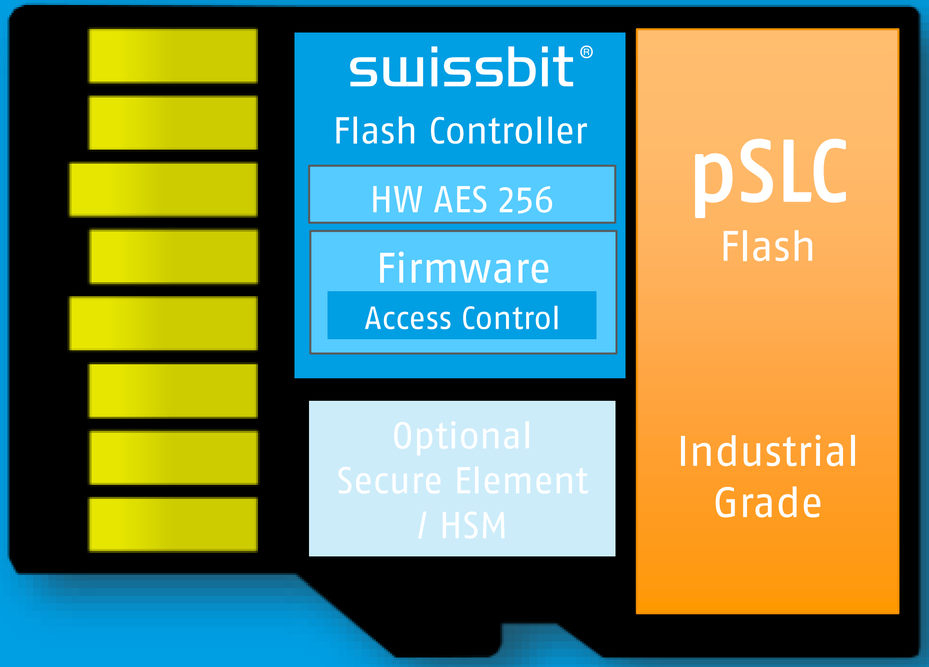
Maker, Hacker, Enthusiast
linkedin.com/in/**marxram/**
roland.marx@swissbit.com

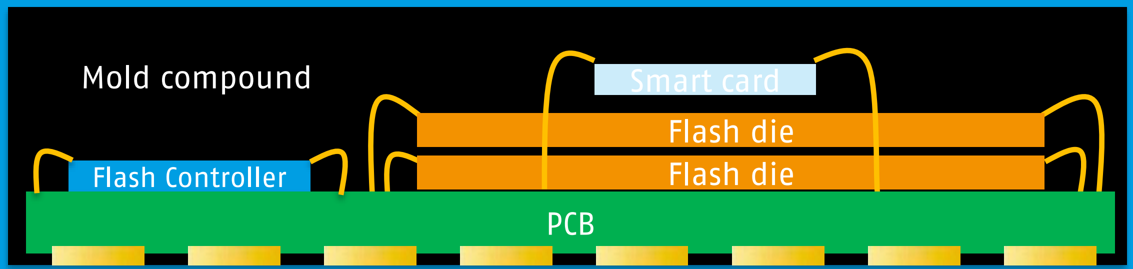
01 – SWISSBIT SECURITY

SECURED BY swissbit

Security Solutions for IoT and Industry







swissbit®

Multiple use cases

Various protection goals

Privacy

Access Restriction

Identity Protection

Access Control

Data Manipulation

Access Restriction

Fraud Protection

Digital Signatures

Privacy Protection

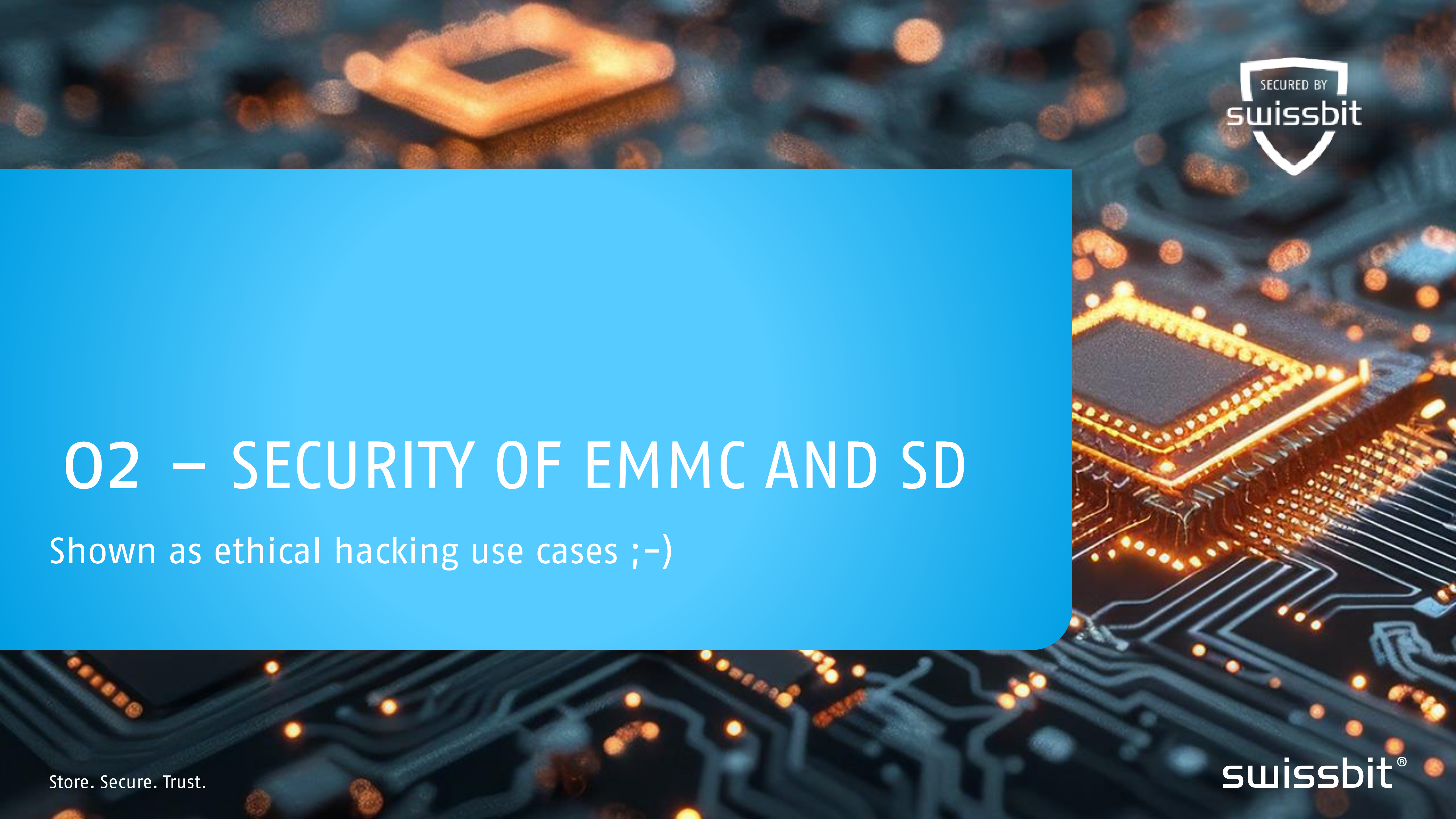
Service Solution

System Integrity

Secure Storage

Authenticity

Licensing



02 – SECURITY OF EMMC AND SD

Shown as ethical hacking use cases ;-)

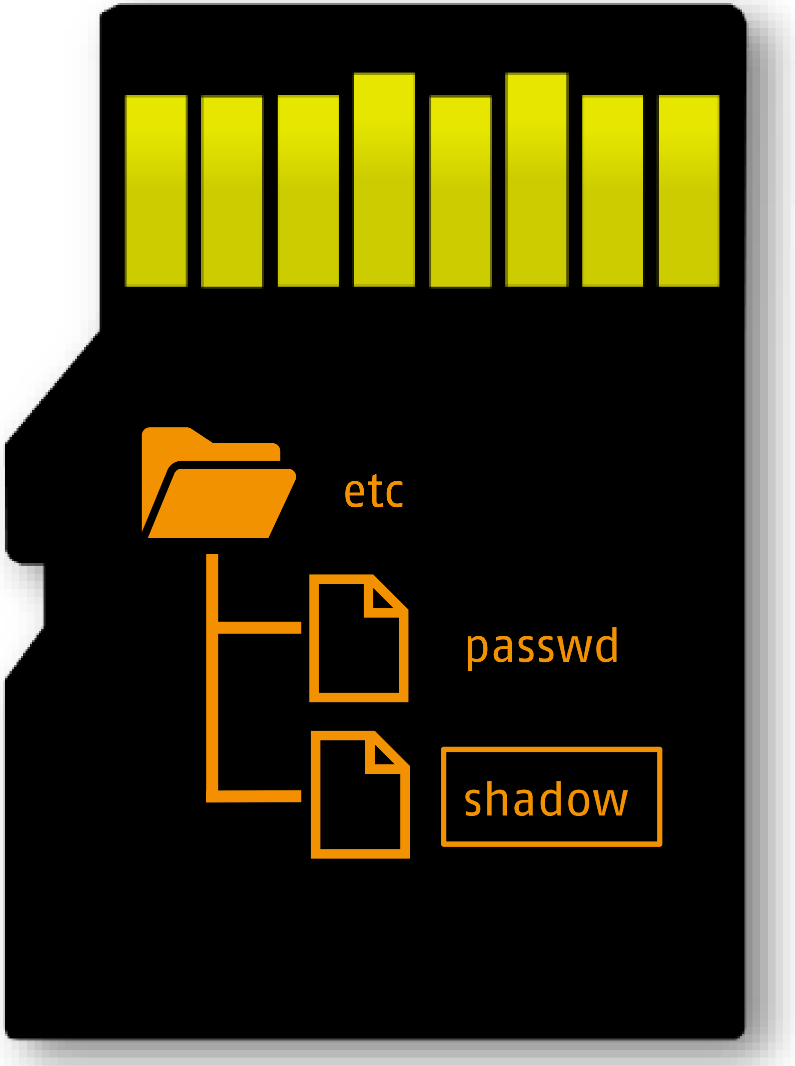
Lost Password Problem

„Oh F*!#… did I change the password for this RPI?“

How to reset the password of a Rpi?

swissbit®

Where are they anyway?

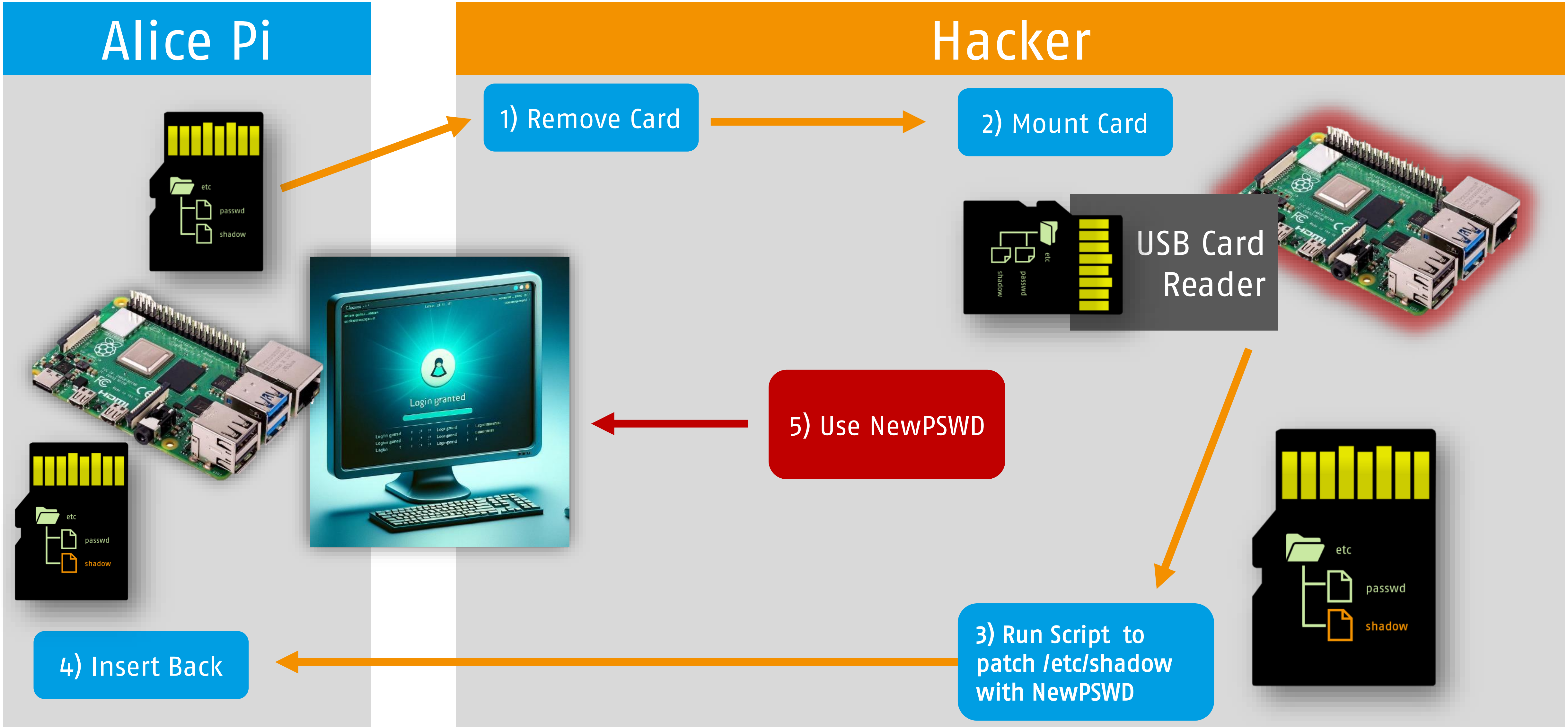


Example Entry in /etc/shadow

```
johndoe:$6$SALT$encryptedpassword:18020:0:99999:7:::
```

- Username: johndoe
- Password: An encrypted password where $6$ indicates the type of hash (SHA-512 in this case), followed by SALT and the actual hashed password.
- Last Change: The password was last changed on the 18,020th day since January 1, 1970.
- Minimum Age: No minimum age is set (0 days).
- Maximum Age: The password must be changed every 99999 days.
- Warning Period: The user receives a warning 7 days before the password expires.
- Inactivity Period: Not specified (the field is empty, indicating no limit).
- Expiration Date: Not specified (the field is empty, indicating the account does not expire).
- Reserved Field: Empty.

How to reset the password of a Rpi?



Alice Pi

Hacker

1) Remove Card

2) Mount Card

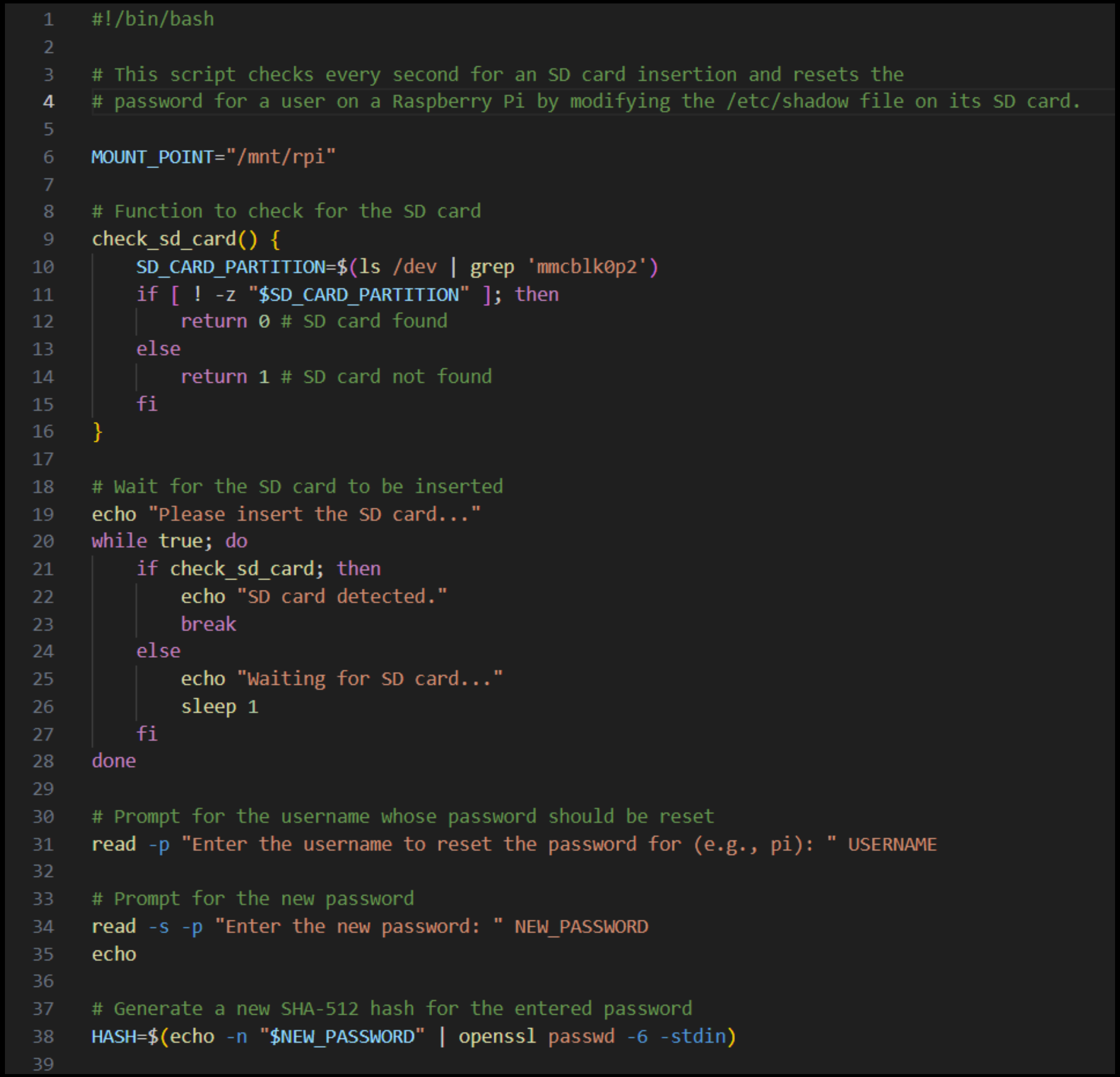
USB Card Reader

3) Run Script to patch /etc/shadow with NewPSWD

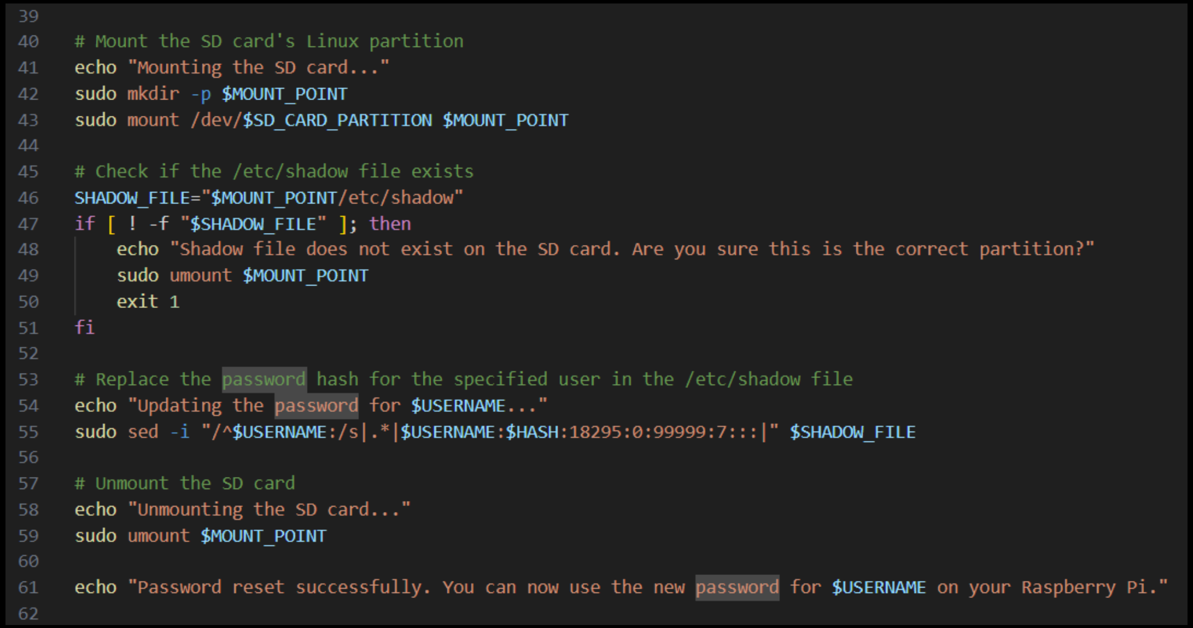
4) Insert Back

5) Use NewPSWD

Some Password Changing Script Fu



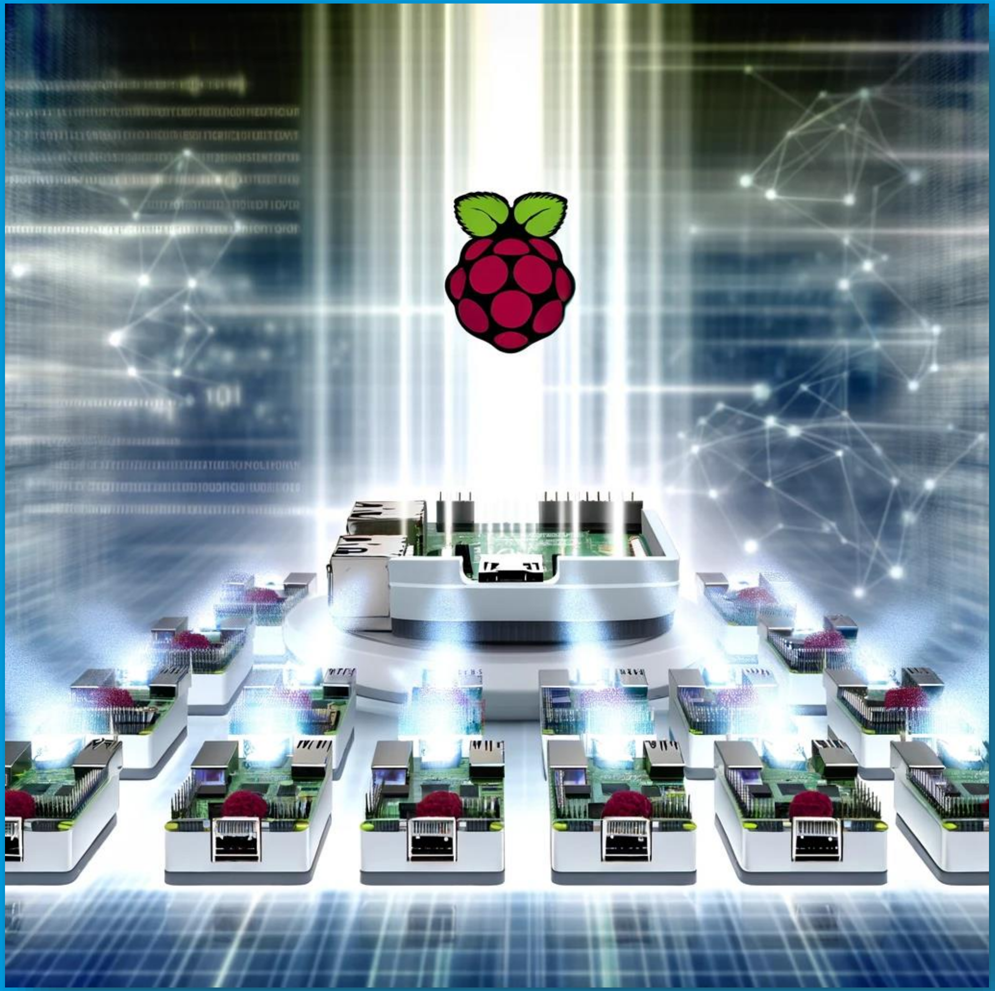
```bash
#!/bin/bash

# This script checks every second for an SD card insertion and resets the
# password for a user on a Raspberry Pi by modifying the /etc/shadow file on its SD card.

MOUNT_POINT="/mnt/rpi"

# Function to check for the SD card
check_sd_card() {
    SD_CARD_PARTITION=$(ls /dev | grep 'mmcblk0p2')
    if [ ! -z "$SD_CARD_PARTITION" ]; then
        return 0 # SD card found
    else
        return 1 # SD card not found
    fi
}

# Wait for the SD card to be inserted
echo "Please insert the SD card..."
while true; do
    if check_sd_card; then
        echo "SD card detected."
        break
    else
        echo "Waiting for SD card..."
        sleep 1
    fi
done

# Prompt for the username whose password should be reset
read -p "Enter the username to reset the password for (e.g., pi): " USERNAME

# Prompt for the new password
read -s -p "Enter the new password: " NEW_PASSWORD
echo

# Generate a new SHA-512 hash for the entered password
HASH=$(echo -n "$NEW_PASSWORD" | openssl passwd -6 -stdin)

```



```bash

# Mount the SD card's Linux partition
echo "Mounting the SD card..."
sudo mkdir -p $MOUNT_POINT
sudo mount /dev/$SD_CARD_PARTITION $MOUNT_POINT

# Check if the /etc/shadow file exists
SHADOW_FILE="$MOUNT_POINT/etc/shadow"
if [ ! -f "$SHADOW_FILE" ]; then
    echo "Shadow file does not exist on the SD card. Are you sure this is the correct partition?"
    sudo umount $MOUNT_POINT
    exit 1
fi

# Replace the password hash for the specified user in the /etc/shadow file
echo "Updating the password for $USERNAME..."
sudo sed -i "/^$USERNAME:/s|.*|$USERNAME:$HASH:18295:0:99999:7:::|" $SHADOW_FILE

# Unmount the SD card
echo "Unmounting the SD card..."
sudo umount $MOUNT_POINT

echo "Password reset successfully. You can now use the new password for $USERNAME on your Raspberry Pi."

```

No backup no compassion

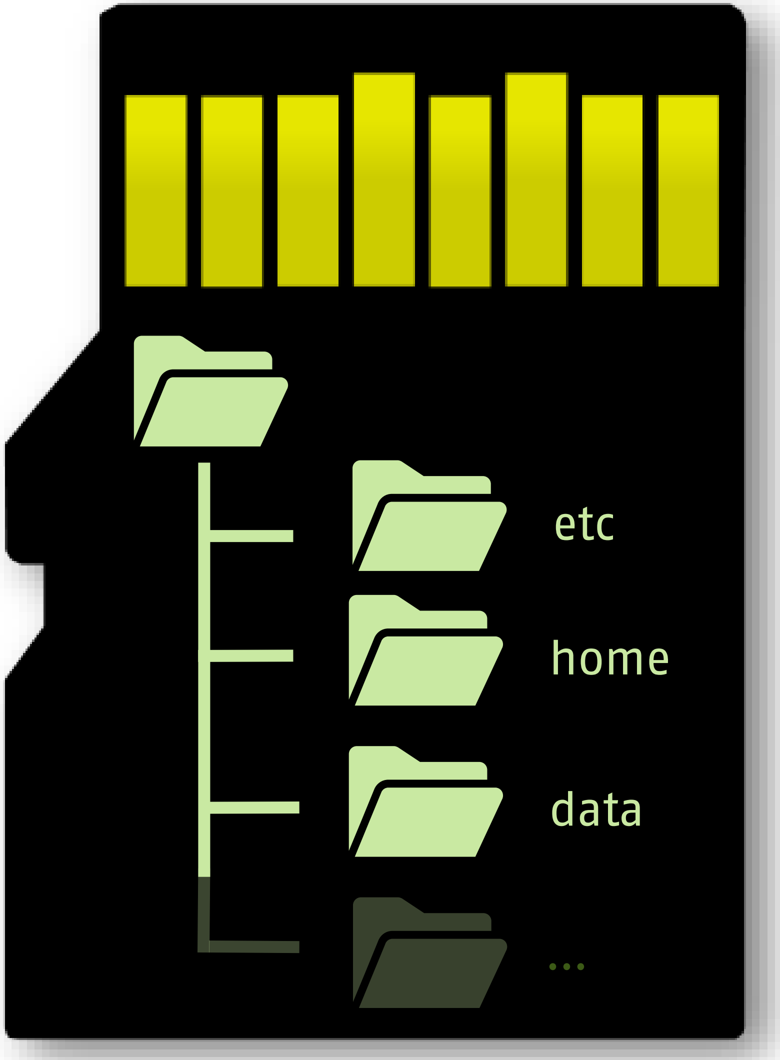
„Better quickly dump the config and important data to be able to recover!“



Example Entry in /etc/shadow

What should I backup?

Which folders are important



/home/

Personal directories for all system users
Stores personal files, configs (e.g., .bashrc, .config)

/root/

Home directory for the root user
Contains root's files and settings

/boot/

Contains boot-related files: Linux kernel, GRUB bootloader configs

/data/

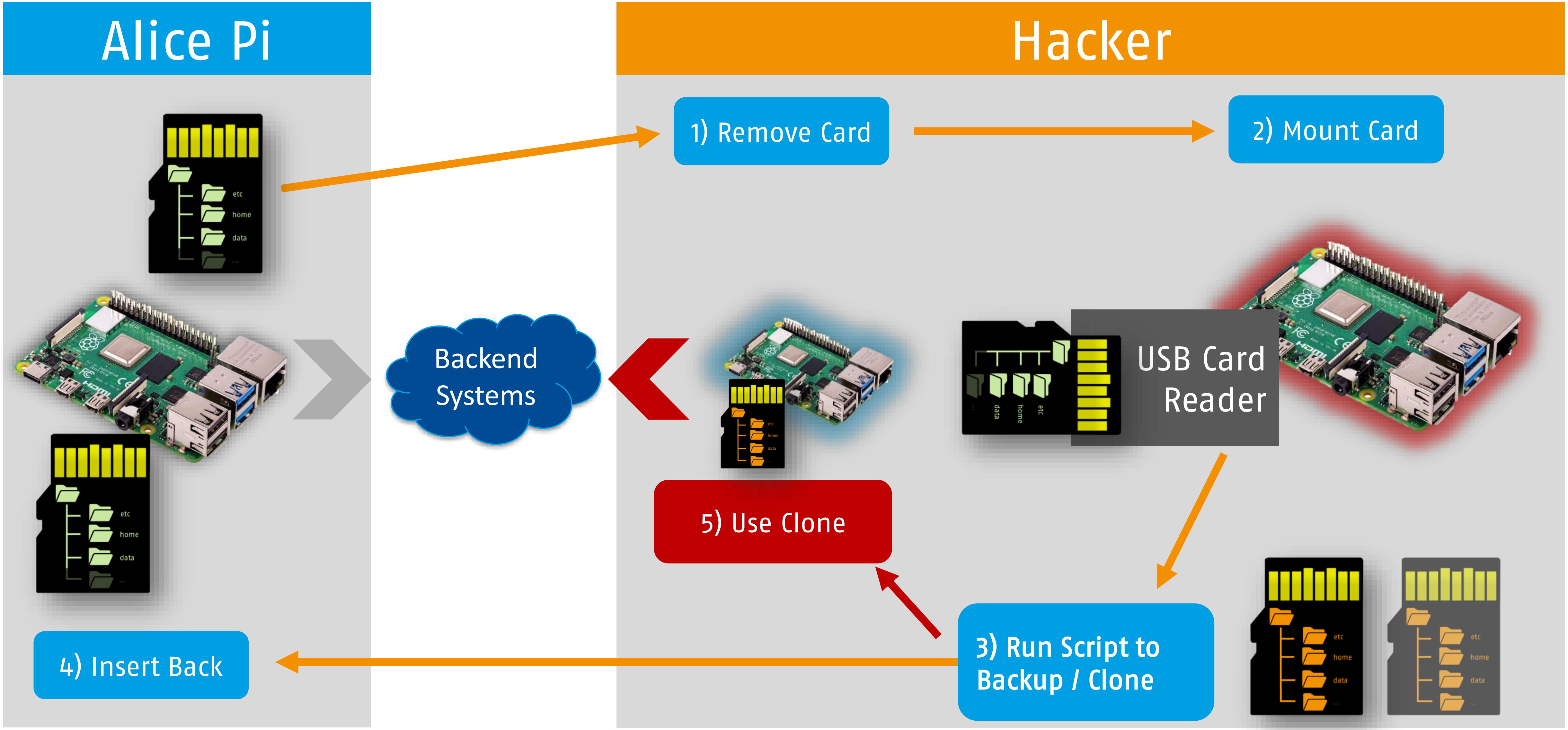
Application or industrial data sets, Database entries
Can also contain specially **privacy** or data of **monetary or business value**

/etc/

Contains all system-wide configuration files
Hosts application and service configs, plus startup/shutdown scripts

/etc/passwd and /etc/shadow
Key for user management: stores account info and encrypted passwords.
/etc/ssh/
Configuration files for SSH, crucial for secure remote access.

How to „make a backup“ of a Rpi?



Alice Pi

Hacker

1) Remove Card

2) Mount Card

USB Card Reader

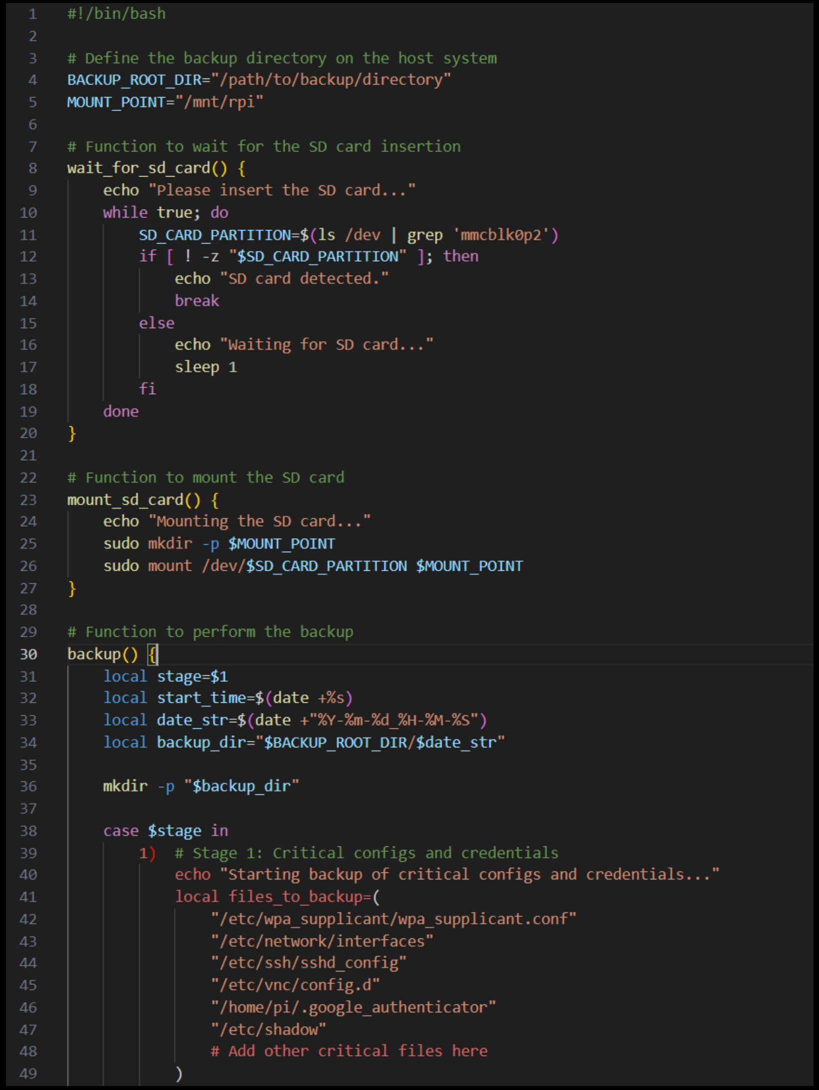
3) Run Script to Backup / Clone

4) Insert Back

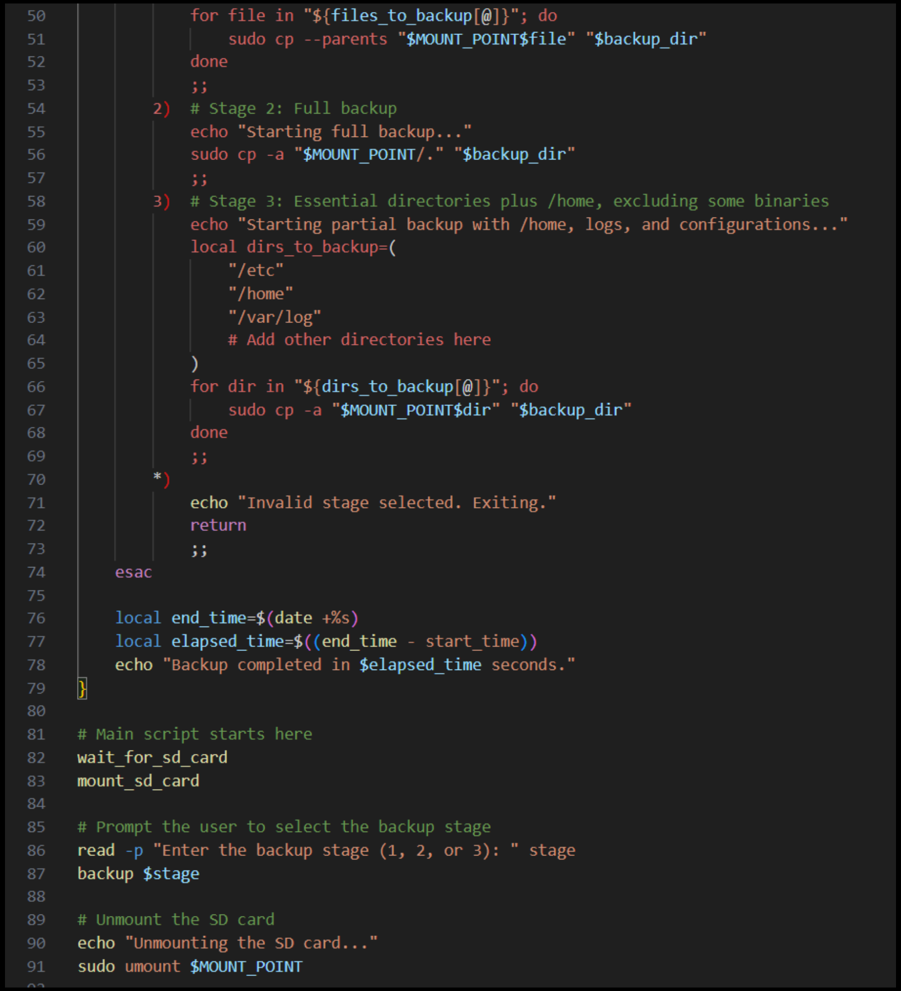
5) Use Clone

Backend Systems

Some Password Changing Script Fu



```bash
#!/bin/bash

# Define the backup directory on the host system
BACKUP_ROOT_DIR="/path/to/backup/directory"
MOUNT_POINT="/mnt/rpi"

# Function to wait for the SD card insertion
wait_for_sd_card() {
    echo "Please insert the SD card..."
    while true; do
        SD_CARD_PARTITION=$(ls /dev | grep 'mmcblk0p2')
        if [ ! -z "$SD_CARD_PARTITION" ]; then
            echo "SD card detected."
            break
        else
            echo "Waiting for SD card..."
            sleep 1
        fi
    done
}

# Function to mount the SD card
mount_sd_card() {
    echo "Mounting the SD card..."
    sudo mkdir -p $MOUNT_POINT
    sudo mount /dev/$SD_CARD_PARTITION $MOUNT_POINT
}

# Function to perform the backup
backup() {
    local stage=$1
    local start_time=$(date +%s)
    local date_str=$(date +"%Y-%m-%d_%H-%M-%S")
    local backup_dir="$BACKUP_ROOT_DIR/$date_str"

    mkdir -p "$backup_dir"

    case $stage in
        1)  # Stage 1: Critical configs and credentials
            echo "Starting backup of critical configs and credentials..."
            local files_to_backup=(
                "/etc/wpa_supplicant/wpa_supplicant.conf"
                "/etc/network/interfaces"
                "/etc/ssh/sshd_config"
                "/etc/vnc/config.d"
                "/home/pi/.google_authenticator"
                "/etc/shadow"
                # Add other critical files here
            )
```



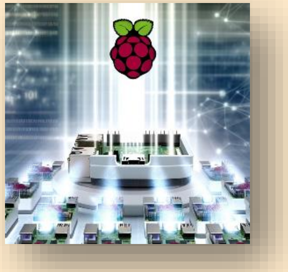
```bash
            for file in "${files_to_backup[@]}"; do
                sudo cp --parents "$MOUNT_POINT$file" "$backup_dir"
            done
            ;;
        2)  # Stage 2: Full backup
            echo "Starting full backup..."
            sudo cp -a "$MOUNT_POINT/." "$backup_dir"
            ;;
        3)  # Stage 3: Essential directories plus /home, excluding some binaries
            echo "Starting partial backup with /home, logs, and configurations..."
            local dirs_to_backup=(
                "/etc"
                "/home"
                "/var/log"
                # Add other directories here
            )
            for dir in "${dirs_to_backup[@]}"; do
                sudo cp -a "$MOUNT_POINT$dir" "$backup_dir"
            done
            ;;
        *)
            echo "Invalid stage selected. Exiting."
            return
            ;;
    esac

    local end_time=$(date +%s)
    local elapsed_time=$((end_time - start_time))
    echo "Backup completed in $elapsed_time seconds."
}

# Main script starts here
wait_for_sd_card
mount_sd_card

# Prompt the user to select the backup stage
read -p "Enter the backup stage (1, 2, or 3): " stage
backup $stage

# Unmount the SD card
echo "Unmounting the SD card..."
sudo umount $MOUNT_POINT
```

Scenario evaluation

Local Attacker

Damage Scenario

FULL System COMPROMISE



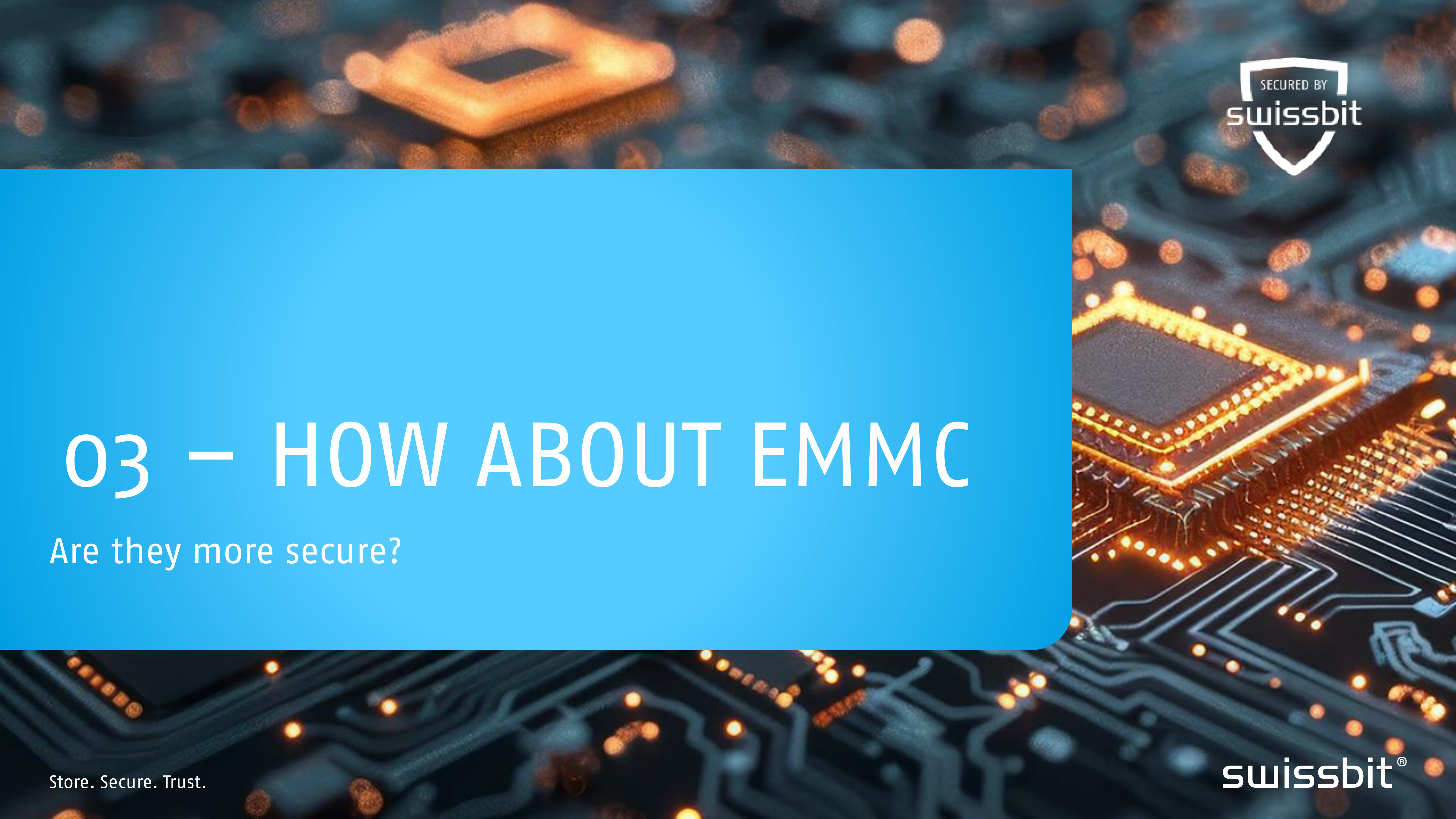
- All data on SD exposed
- All credentials cloned
- Loss of integrity (local)

Attacker Classification

Experience Level	low
Attack Vector	physical
Detectability	medium
Attack duration	minutes
Tools	< 50 €
Know How	Common knowledge

Classic Mitigation

- Physical Protection
- Monitoring



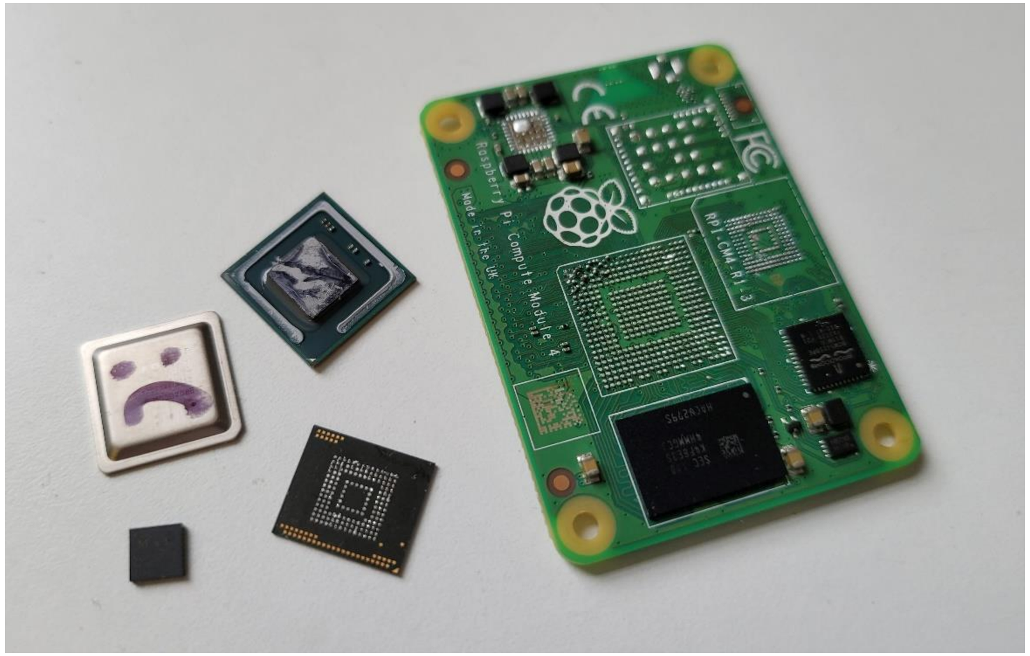
03 – HOW ABOUT EMMC

Are they more secure?

What about e.MMC

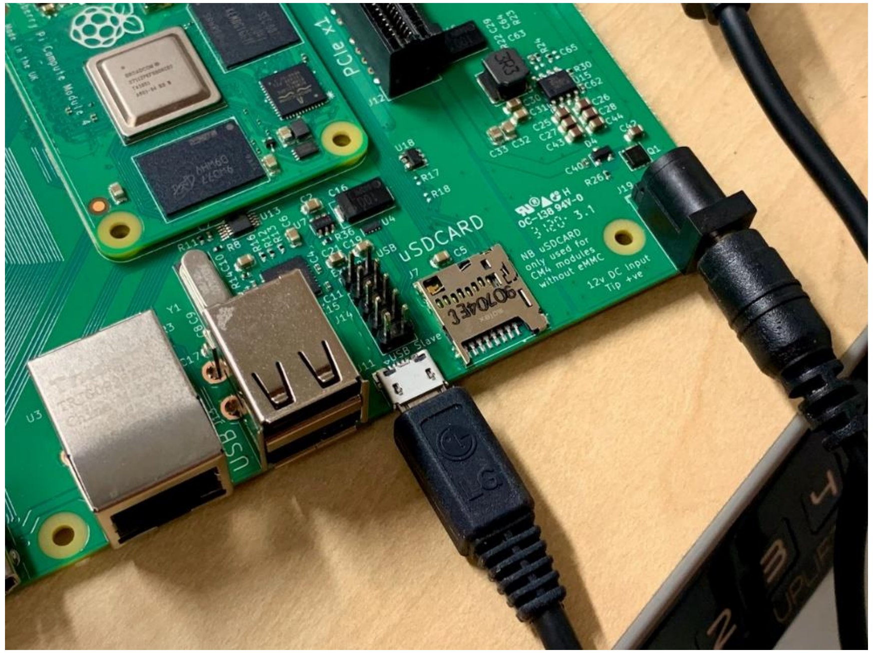
EMMCs are mainly soldered SD cards"

e.MMCs can also be removedv



https://github.com/jeffmakes/pi-data-recovery

Often the readout interfaces are just there



https://www.jeffgeerling.com/blog/2020/how-flash-raspberry-pi-os-compute-module-4-emmc-usbboot

04 – ASSETS

What you might have and should protect

Security Upgrade Kit ensures business continuity

Intellectual Property

Credentials

Identifiers (Device and people)

Digital Product-License

Algorithms / Firmware

Logfiles / Logdata

Location

Accounts

Process Data

Configuration (Files/Flags)

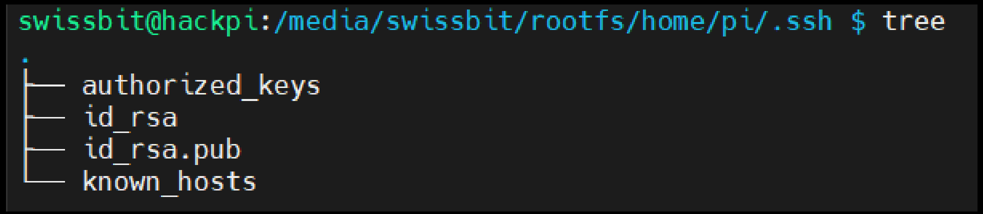
Safety Functions

Sensor Data

Privacy

Communication

SSH



```
swissbit@hackpi:/media/swissbit/rootfs/home/pi/.ssh $ tree
.
├── authorized_keys
├── id_rsa
├── id_rsa.pub
└── known_hosts
```

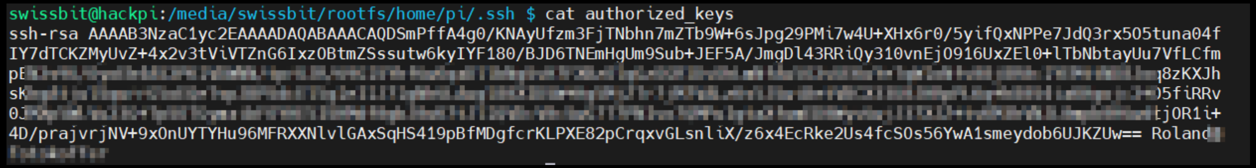
🗁 /etc/ssh/sshd_config

Config that can allow root login, Ports, PubKey or PW ...

I can enable root access, or PW-less login...

🗁 /home/USRNAME/.ssh/authorized_keys

Bunch of all public keys and constraints for login via SSH using public keys

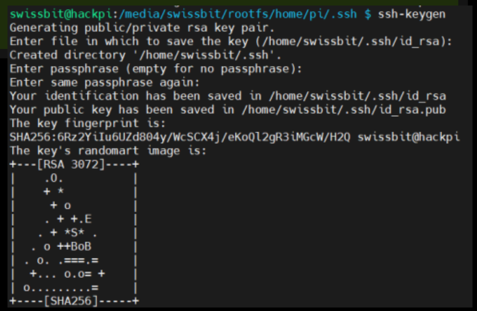


```
swissbit@hackpi:/media/swissbit/rootfs/home/pi/.ssh $ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDSmPffA4g0/KNAyUfzm3FjTNbhn7mZTb9W+6sJpg29PMi7w4U+XHx6r0/5yifQxNPPe7JdQ3rx505tuna04f
IY7dTCKZMyUvZ+4x2v3tViVTZnG6Ixz0BtmZSssutw6kyIYF180/BJD6TNEmHgUm9Sub+JEF5A/JmgDl43RRiQy310vnEj0916UxZEl0+lTbNbtayUu7VfLCfm
pE[redacted]q8zKXJh
sK[redacted]05fiRRv
0J[redacted]tjOR1i+
4D/prajvrjNV+9x0nUYTYHu96MFRXXNlvlGAxSqHS419pBfMDgfcrKLPXE82pCrqxvGLsnliX/z6x4EcRke2Us4fcS0s56YwA1smeydob6UJKZUw== Roland[redacted]
```

Allowlist who may login!
Human key won't see a fake key ;-)

🗁 /home/USRNAME/.ssh/id_rsa // or other names

Private keys for logging into other systems

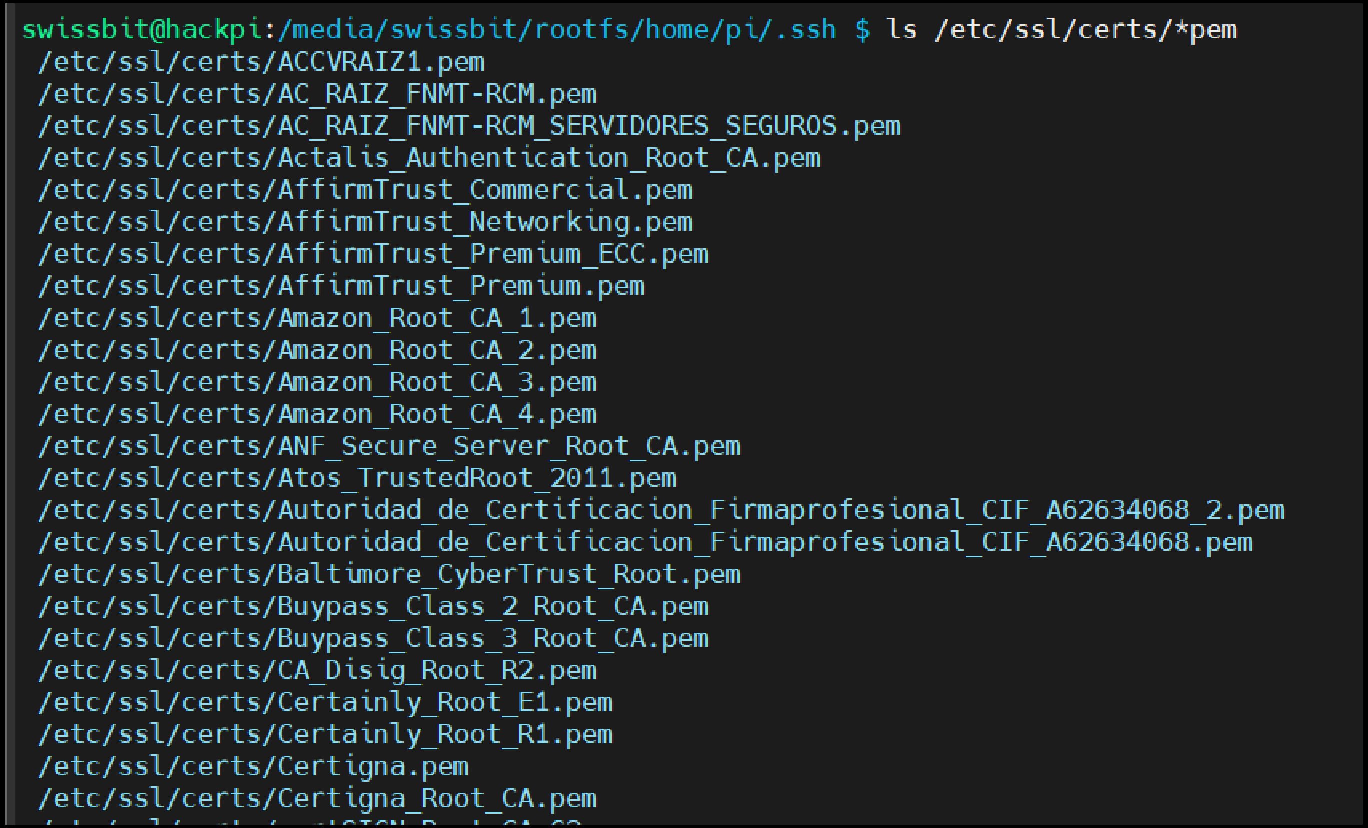


```
swissbit@hackpi:/media/swissbit/rootfs/home/pi/.ssh $ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/swissbit/.ssh/id_rsa):
Created directory '/home/swissbit/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/swissbit/.ssh/id_rsa
Your public key has been saved in /home/swissbit/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:6Rz2YiIu6UZd804y/WcSCX4j/eKoQl2gR3iMGcW/H2Q swissbit@hackpi
The key's randomart image is:
+---[RSA 3072]----+
|     .O.         |
|     + *         |
|      + o        |
|     . + +.E     |
|    . + *S* .    |
|   . o ++BoB     |
|  . o. .===.=    |
|   +... o.o= +   |
|  o.........=    |
+----[SHA256]-----+
```

Keys without passphrase can directly be used

Keys with a weak passphrase can be exported and remotely bruteforced!

PKI / Root CAs



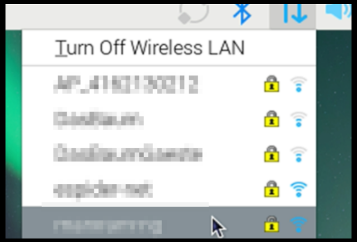
```
swissbit@hackpi:/media/swissbit/rootfs/home/pi/.ssh $ ls /etc/ssl/certs/*pem
 /etc/ssl/certs/ACCVRAIZ1.pem
 /etc/ssl/certs/AC_RAIZ_FNMT-RCM.pem
 /etc/ssl/certs/AC_RAIZ_FNMT-RCM_SERVIDORES_SEGUROS.pem
 /etc/ssl/certs/Actalis_Authentication_Root_CA.pem
 /etc/ssl/certs/AffirmTrust_Commercial.pem
 /etc/ssl/certs/AffirmTrust_Networking.pem
 /etc/ssl/certs/AffirmTrust_Premium_ECC.pem
 /etc/ssl/certs/AffirmTrust_Premium.pem
 /etc/ssl/certs/Amazon_Root_CA_1.pem
 /etc/ssl/certs/Amazon_Root_CA_2.pem
 /etc/ssl/certs/Amazon_Root_CA_3.pem
 /etc/ssl/certs/Amazon_Root_CA_4.pem
 /etc/ssl/certs/ANF_Secure_Server_Root_CA.pem
 /etc/ssl/certs/Atos_TrustedRoot_2011.pem
 /etc/ssl/certs/Autoridad_de_Certificacion_Firmaprofesional_CIF_A62634068_2.pem
 /etc/ssl/certs/Autoridad_de_Certificacion_Firmaprofesional_CIF_A62634068.pem
 /etc/ssl/certs/Baltimore_CyberTrust_Root.pem
 /etc/ssl/certs/Buypass_Class_2_Root_CA.pem
 /etc/ssl/certs/Buypass_Class_3_Root_CA.pem
 /etc/ssl/certs/CA_Disig_Root_R2.pem
 /etc/ssl/certs/Certainly_Root_E1.pem
 /etc/ssl/certs/Certainly_Root_R1.pem
 /etc/ssl/certs/Certigna.pem
 /etc/ssl/certs/Certigna_Root_CA.pem
```

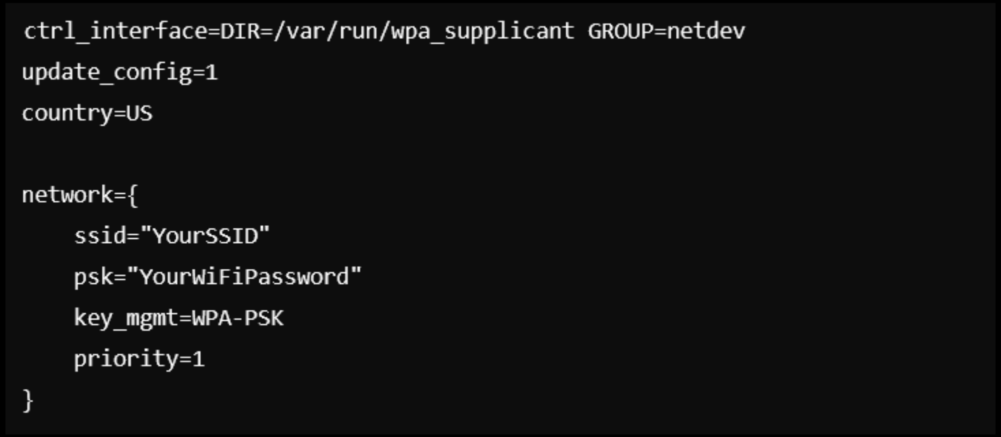
If you add your own CA here, the system will trust it and all derived certificates

Good example for Integrity protection!

WIFI Configs (WPA Supplicant)

WiFi Client Config





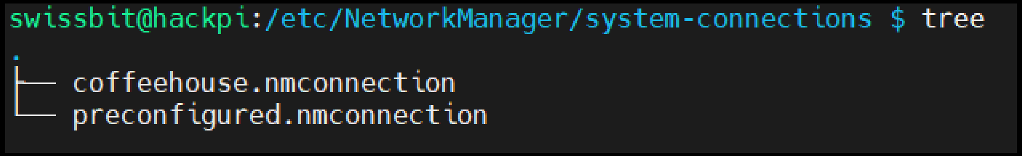
```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=US

network={
    ssid="YourSSID"
    psk="YourWiFiPassword"
    key_mgmt=WPA-PSK
    priority=1
}
```

WiFi Passwords for existing Stations are stored as clear text!

/etc/wpa_supplicant/wpa_supplicant.conf

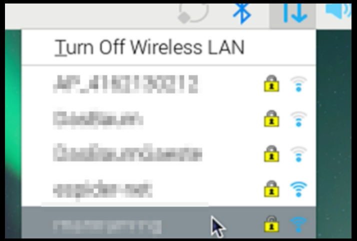
Storage for WiFi Settings

WIFI Configs (Networkmanager)

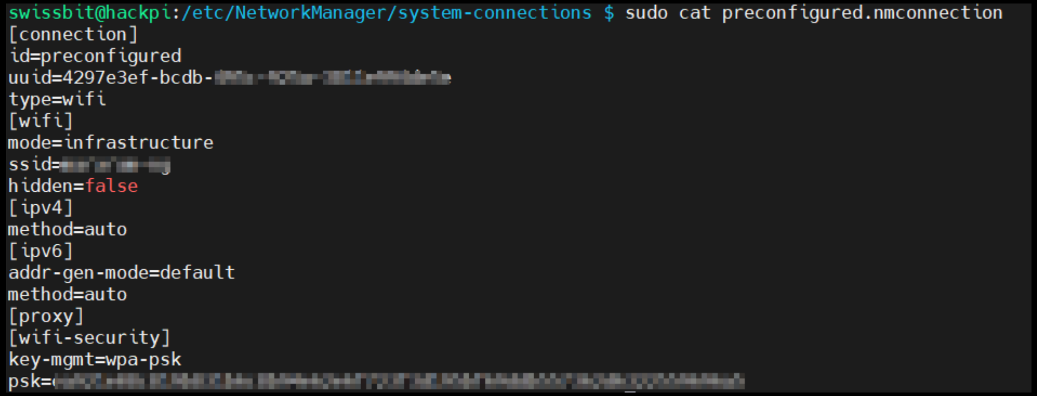


```
swissbit@hackpi:/etc/NetworkManager/system-connections $ tree
.
├── coffeehouse.nmconnection
└── preconfigured.nmconnection
```

WiFi Client Config



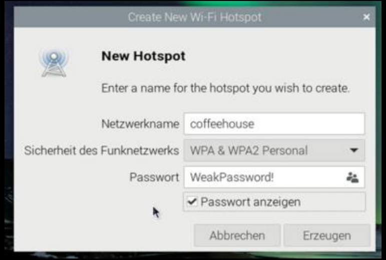
[redacted]



```
swissbit@hackpi:/etc/NetworkManager/system-connections $ sudo cat preconfigured.nmconnection
[connection]
id=preconfigured
uuid=4297e3ef-bcdb-[redacted]
type=wifi
[wifi]
mode=infrastructure
ssid=[redacted]
hidden=false
[ipv4]
method=auto
[ipv6]
addr-gen-mode=default
method=auto
[proxy]
[wifi-security]
key-mgmt=wpa-psk
psk=[redacted]
```

WiFi Passwords for existing Stations are stored as clear text!

WiFi Accesspoint Config



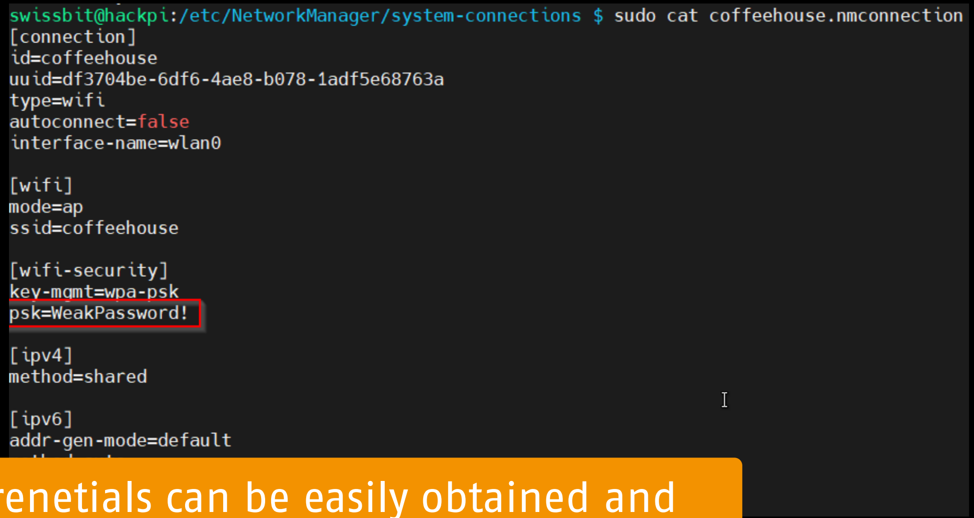
Enter a name for the hotspot you wish to create.

Netzwerkname: coffeehouse

Sicherheit des Funknetzwerks: WPA & WPA2 Personal

Passwort: WeakPassword!

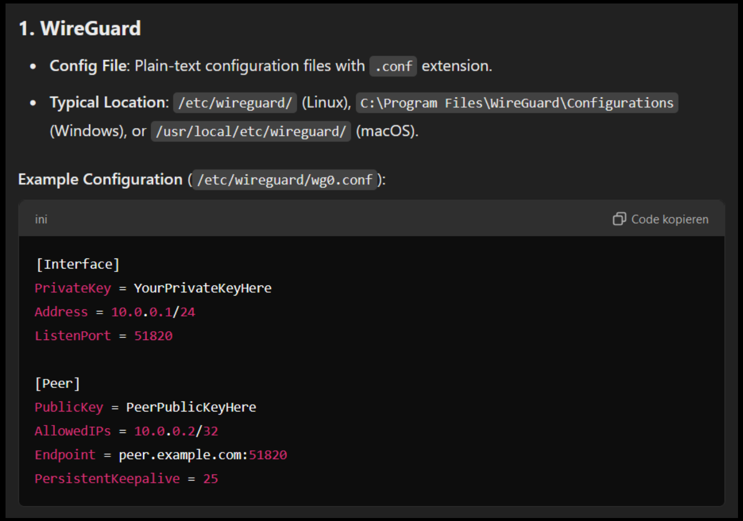
☑ Passwort anzeigen



```
swissbit@hackpi:/etc/NetworkManager/system-connections $ sudo cat coffeehouse.nmconnection
[connection]
id=coffeehouse
uuid=df3704be-6df6-4ae8-b078-1adf5e68763a
type=wifi
autoconnect=false
interface-name=wlan0

[wifi]
mode=ap
ssid=coffeehouse

[wifi-security]
key-mgmt=wpa-psk
psk=WeakPassword!

[ipv4]
method=shared

[ipv6]
addr-gen-mode=default
```

Also AP Crenetials can be easily obtained and allow persitent access via WiFi!

VPN config examples



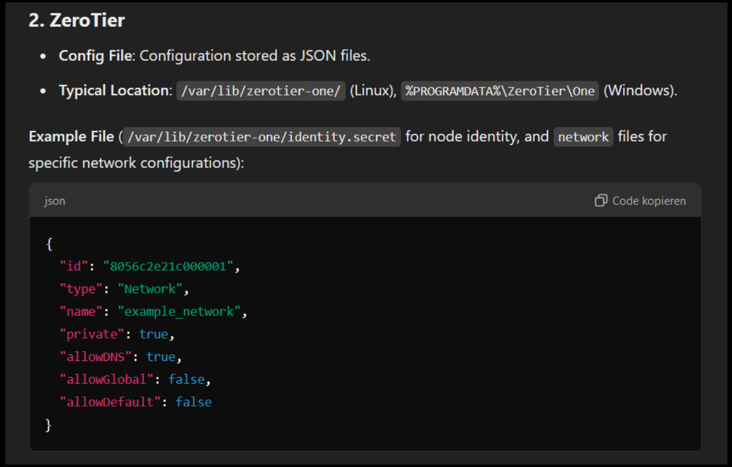
1. WireGuard

- **Config File**: Plain-text configuration files with `.conf` extension.
- **Typical Location**: `/etc/wireguard/` (Linux), `C:\Program Files\WireGuard\Configurations` (Windows), or `/usr/local/etc/wireguard/` (macOS).

Example Configuration (`/etc/wireguard/wg0.conf`):

ini Code kopieren

```ini
[Interface]
PrivateKey = YourPrivateKeyHere
Address = 10.0.0.1/24
ListenPort = 51820

[Peer]
PublicKey = PeerPublicKeyHere
AllowedIPs = 10.0.0.2/32
Endpoint = peer.example.com:51820
PersistentKeepalive = 25
```



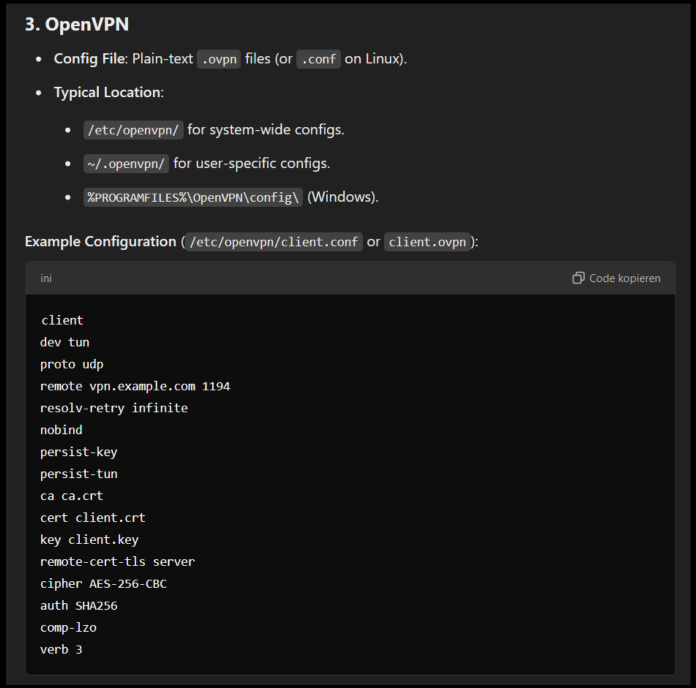
2. ZeroTier

- **Config File**: Configuration stored as JSON files.
- **Typical Location**: `/var/lib/zerotier-one/` (Linux), `%PROGRAMDATA%\ZeroTier\One` (Windows).

Example File (`/var/lib/zerotier-one/identity.secret` for node identity, and `network` files for specific network configurations):

json Code kopieren

```json
{
  "id": "8056c2e21c000001",
  "type": "Network",
  "name": "example_network",
  "private": true,
  "allowDNS": true,
  "allowGlobal": false,
  "allowDefault": false
}
```



3. OpenVPN

- **Config File**: Plain-text `.ovpn` files (or `.conf` on Linux).
- **Typical Location**:
  - `/etc/openvpn/` for system-wide configs.
  - `~/.openvpn/` for user-specific configs.
  - `%PROGRAMFILES%\OpenVPN\config\` (Windows).

Example Configuration (`/etc/openvpn/client.conf` or `client.ovpn`):

ini Code kopieren

```ini
client
dev tun
proto udp
remote vpn.example.com 1194
resolv-retry infinite
nobind
persist-key
persist-tun
ca ca.crt
cert client.crt
key client.key
remote-cert-tls server
cipher AES-256-CBC
auth SHA256
comp-lzo
verb 3
```

If the VPN settings are plaintext on the storage and
not specially protected on a secure element, TPM etc.
they can be cloned and used to authenticate

Backend cannot differentiate who is who!?
If two devices login.
Only Metadata like IPs, etc. could help

More ideas of assets?

Browser Password

Browser History

.bash_history

Auth logs → Can be deleted?!
Need for WORM

Routing & DNS settings

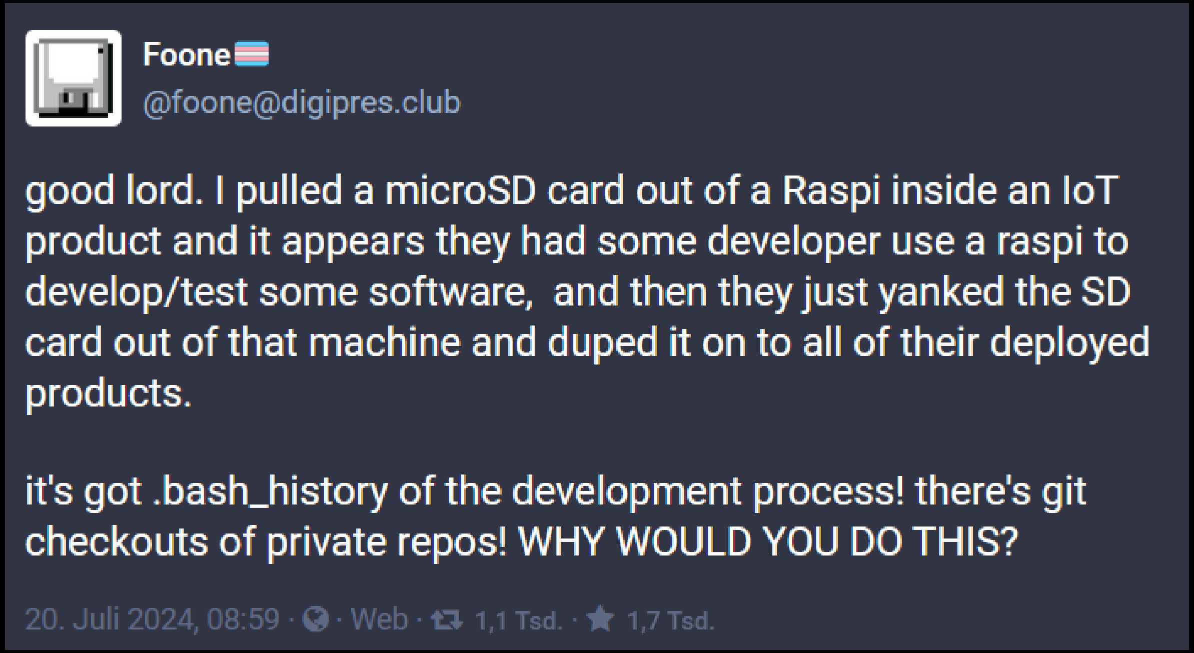
Firewall Settings

Cloud Connection Credentials

AI Model files

DB Credentials

Is this a real thing?



Foone🏳️‍⚧️
@foone@digipres.club

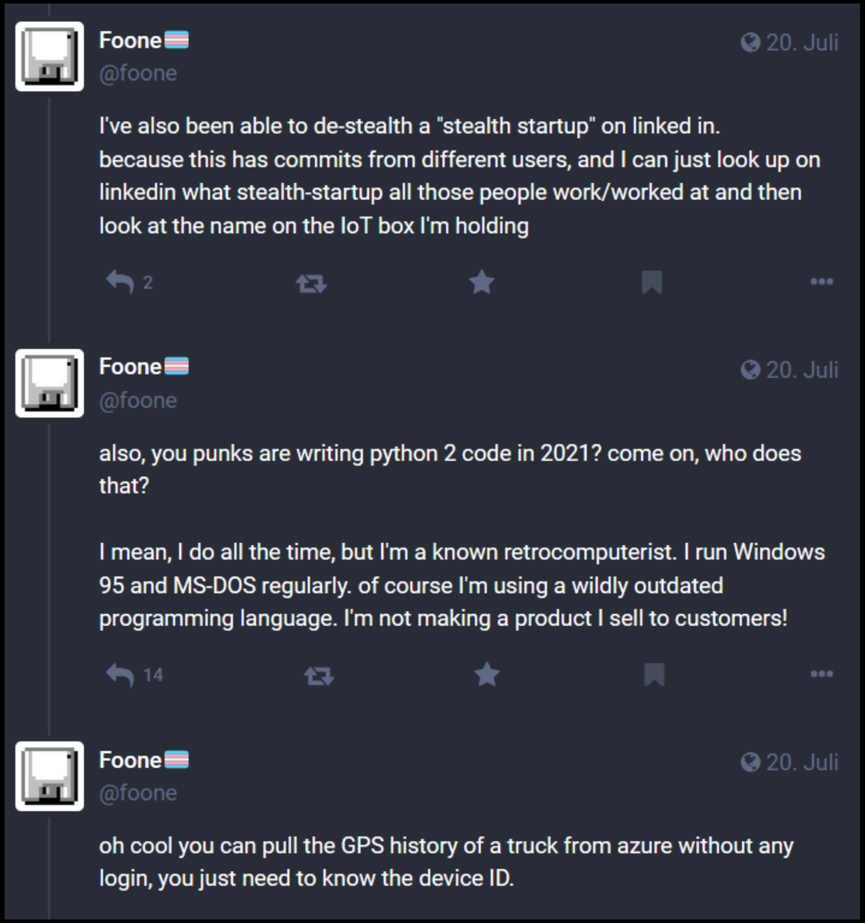
good lord. I pulled a microSD card out of a Raspi inside an IoT product and it appears they had some developer use a raspi to develop/test some software, and then they just yanked the SD card out of that machine and duped it on to all of their deployed products.

it's got .bash_history of the development process! there's git checkouts of private repos! WHY WOULD YOU DO THIS?

20. Juli 2024, 08:59 · 🌐 · Web · 🔁 1,1 Tsd. · ★ 1,7 Tsd.

https://digipres.club/@foone/112817523308786223

→ No security circumvented! There was none!



Foone🏳️‍⚧️ 🌐 20. Juli
@foone

I've also been able to de-stealth a "stealth startup" on linked in. because this has commits from different users, and I can just look up on linkedin what stealth-startup all those people work/worked at and then look at the name on the IoT box I'm holding

↩ 2

Foone🏳️‍⚧️ 🌐 20. Juli
@foone

also, you punks are writing python 2 code in 2021? come on, who does that?

I mean, I do all the time, but I'm a known retrocomputerist. I run Windows 95 and MS-DOS regularly. of course I'm using a wildly outdated programming language. I'm not making a product I sell to customers!

↩ 14

Foone🏳️‍⚧️ 🌐 20. Juli
@foone

oh cool you can pull the GPS history of a truck from azure without any login, you just need to know the device ID.

05 – Security Retrofit

WARNING:

Contains Placement of cool product!

Ensuring secure embedded Linux systems

Protecting data e.g. IP, configurations and credentials from being stolen, copied or manipulated on any embedded Linux system

© WAGO AG

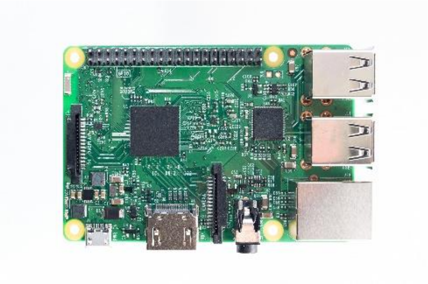
Boot drive protection

Ensure data confidentiality of externally stored data like security logs, privacy or configurations on any embedded Linux system

© WAGO AG

External storage protection

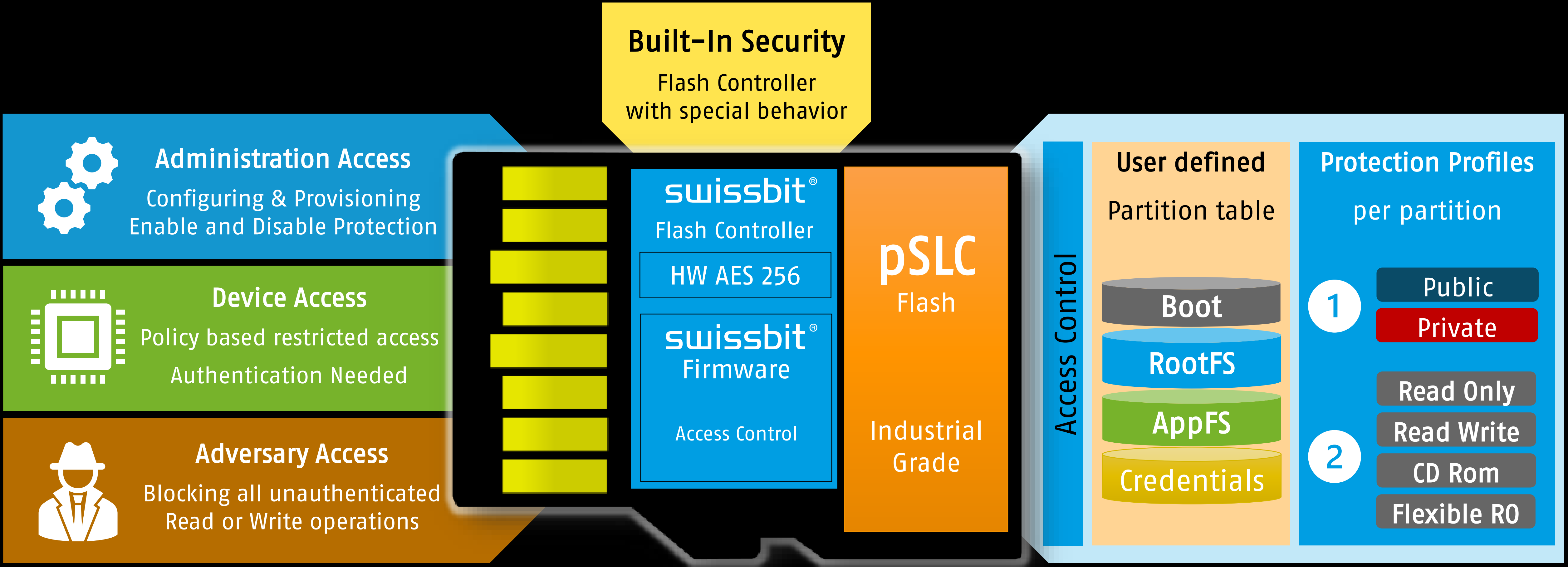
Ensure system integrity of the operating system and applications on many embedded Linux systems



Secure boot

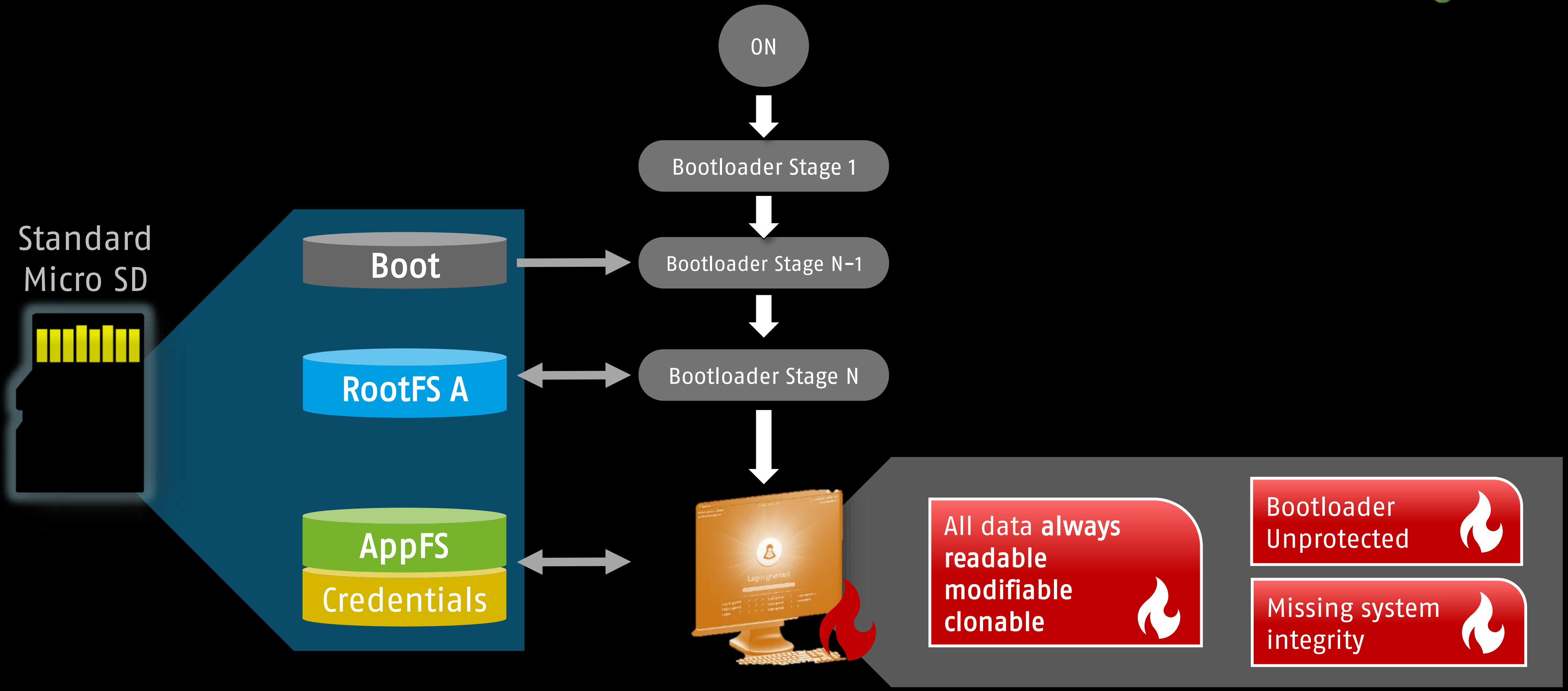
Security Upgrade Kit

How does it work?



Standard boot flow of a RPi

No Integrity protection and Security



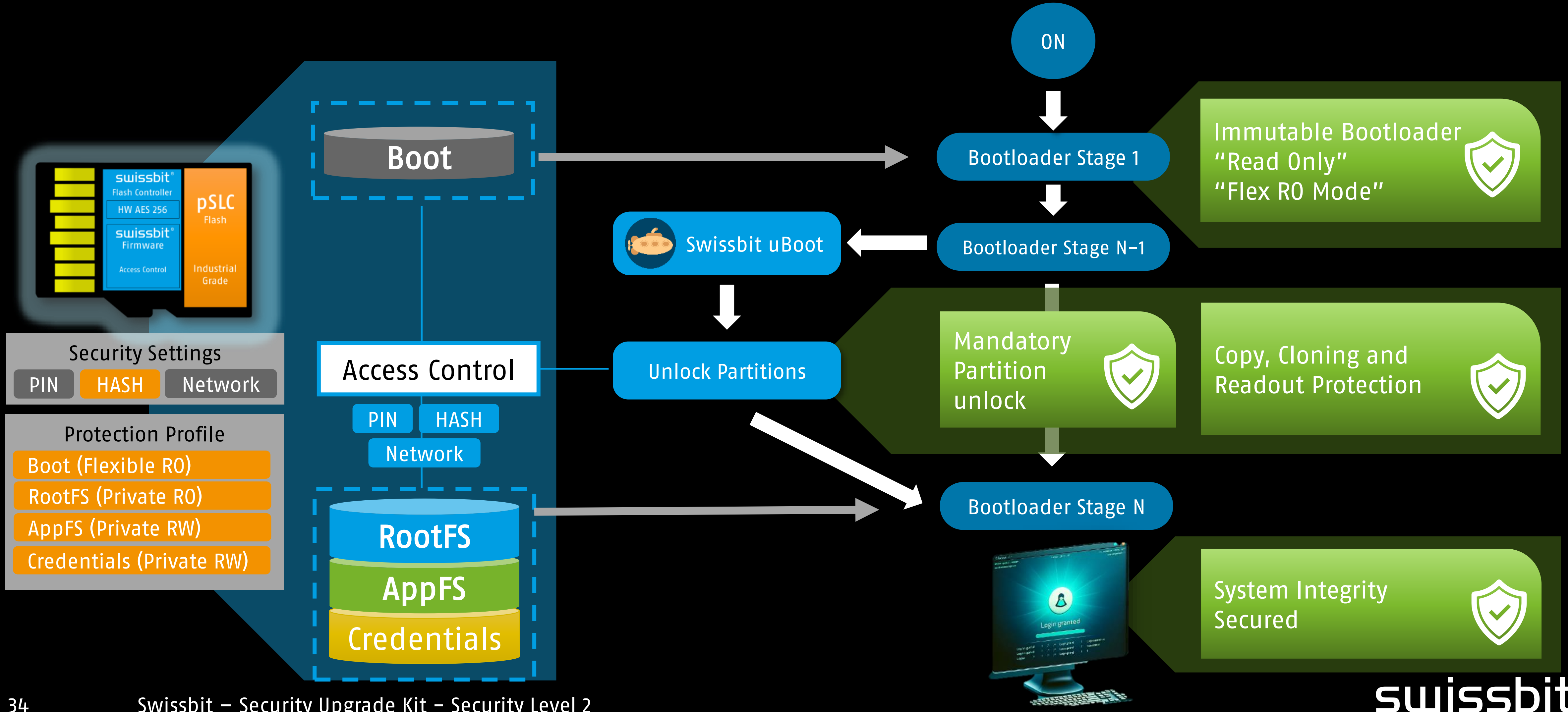
All data **always readable modifiable clonable**

Bootloader Unprotected

Missing system integrity

Secure External Drive

Reference Implementation



Security Settings

PIN | HASH | Network

Protection Profile

- Boot (Flexible RO)
- RootFS (Private RO)
- AppFS (Private RW)
- Credentials (Private RW)

Boot

Access Control

PIN | HASH | Network

RootFS

AppFS

Credentials

ON

Bootloader Stage 1

Bootloader Stage N-1

Swissbit uBoot

Unlock Partitions

Bootloader Stage N

Immutable Bootloader
"Read Only"
"Flex RO Mode"

Mandatory Partition unlock

Copy, Cloning and Readout Protection

System Integrity Secured

Physical and Cyber Protection for existing products

Physical Attacks

Remote Attacks

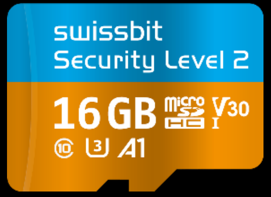
Existing Protection Options

- Monitoring
- Physical Protection
- Software Hardening
- Attack Surface Reduction

swissbit®

Security Level 2 Extra Protection

- Hardware Based Access Control
- Partition specific protection profiles
- Data Encryption
- Readout, Cloning & Tamper Protection
- System Integrity / Secure Boot
- Privacy Protection



Assets to protect on Embedded Systems

- Apps
- Configs
- Credentials
- Safety
- Functionality

Raspberry Pi as reference system for embedded IoT/OT platforms

Link

- Keys
- Certs

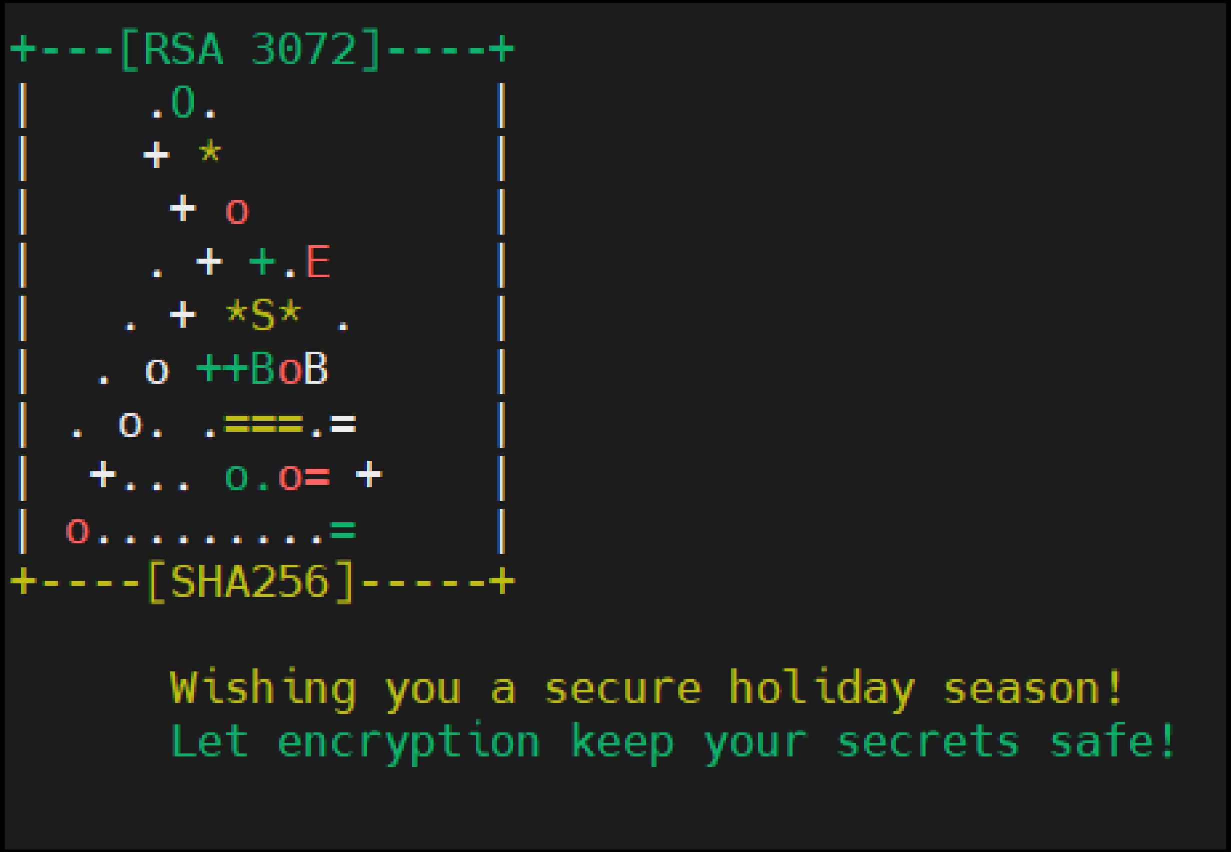
- Data
- Services

Thanks for watching

Maker, Hacker, Enthusiast

linkedin.com/in/**marxram/**

roland.marx@swissbit.com



```
+---[RSA 3072]----+
|    .O.          |
|    + *          |
|     + o         |
|    . + +.E      |
|   . + *S* .     |
|  . o ++BoB      |
| . o. .===.=     |
|  +... o.o= +    |
| o.........=     |
+----[SHA256]-----+

      Wishing you a secure holiday season!
      Let encryption keep your secrets safe!
```