
Projectvoorstel: Sensorio - Enhanced Security Edition

1. Inleiding en Kerndoelstelling

Dit project is een doorontwikkeling van het oorspronkelijke "Sensorio" concept. Het primaire doel is de realisatie van een **significant verbeterd, hoog-beveiligd IoT-systeem** dat niet enkel omgevingsdata monitort, maar ook robuuste **fysieke beveiliging** en **manipulatiedetectie** integreert. We streven naar een systeem dat:

1. **Betrouwbaar** omgevingsdata verzamelt (temperatuur, vochtigheid, CO2, etc.).
2. Deze data **veilig en direct** communiceert via een end-to-end versleutelde verbinding.
3. De **hardware zelf beschermt** tegen uitlezen en ongeautoriseerde firmware.
4. **Fysieke toegang** tot de behuizing controleert en beveiligt.
5. Pogingen tot **manipulatie of diefstal actief detecteert** en hierop reageert.
6. Data opslaat voor analyse en visualiseert via een **veilig platform**.
7. Draait op een **serveromgeving die proactief wordt beveiligd**.

De kern van deze verbeterde editie ligt in de implementatie van geavanceerde security-mechanismen op zowel hardware-, software- als netwerkniveau, en de toevoeging van fysieke beveiligingscomponenten.

2. Kernvernieuwingen en Verbeteringen (Het "Wat" en "Waarom")

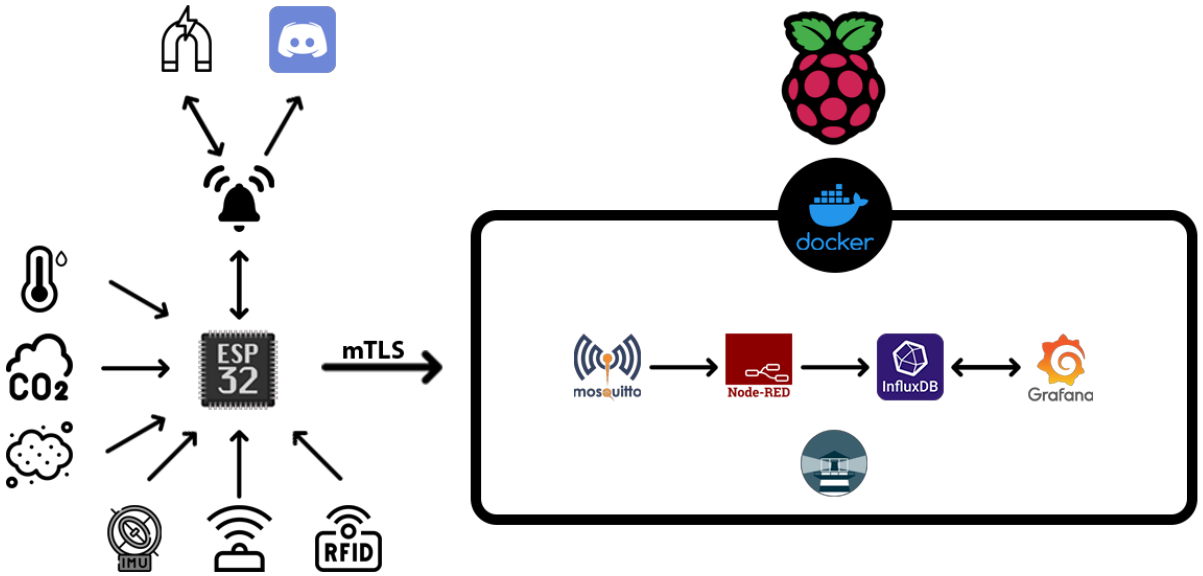
Ten opzichte van de initiële versie introduceren we de volgende cruciale vernieuwingen:

- **Overstap naar ESP-IDF Framework:**
    - **Wat:** We vervangen de Arduino-omgeving door het Espressif IoT Development Framework (ESP-IDF).
    - **Waarom:** ESP-IDF biedt low-level toegang tot de ESP32-hardware, wat essentieel is voor het implementeren van geavanceerde security features zoals Secure Boot en Flash Encryption, die niet (eenvoudig) beschikbaar zijn via de standaard Arduino core. Dit geeft ons de controle die nodig is voor een robuuste beveiliging.
- **Directe mTLS Communicatie vanaf ESP32:**
    - **Wat:** De ESP32 communiceert *rechtstreeks* met de Mosquitto MQTT broker via MQTT over een **Mutual TLS (mTLS)** verbinding. De tussenliggende Raspberry Pi voor seriële dataoverdracht wordt geëlimineerd.
    - **Waarom:** mTLS zorgt voor twee cruciale zaken: 1) **Encryptie:** Alle data tussen ESP32 en server is versleuteld en onleesbaar voor afluisteraars. 2) **Wederzijdse Authenticatie:** Zowel de ESP32 als de server *moeten* zich met geldige certificaten identificeren. Dit voorkomt dat een ongeautoriseerd apparaat zich voordoet als onze ESP32 (en data injecteert) of dat de ESP32 data naar een malafide server stuurt. Door de Pi4 te elimineren, verkleinen we het aanvalsoppervlak en de complexiteit.

- **ESP32 Hardware Tamper-Proofing:**
    - **Wat:** Implementatie van **Secure Boot V2** en **Flash Encryption** via ESP-IDF configuratie.
    - **Waarom:**
        - **Secure Boot:** Garandeert dat alleen door ons cryptografisch ondertekende firmware op de ESP32 kan draaien. Een aanvaller kan geen eigen, mogelijk kwaadaardige, firmware flashen.
        - **Flash Encryption:** Versleutelt de *volledige* inhoud van de ESP32 flash-opslag (firmware, configuratie, en vooral de gevoelige mTLS private key en geautoriseerde RFID-tags). Zelfs als een aanvaller fysieke toegang krijgt en de flashchip probeert uit te lezen, zijn de data onbruikbaar zonder de unieke, hardware-gebonden encryptiesleutel. Dit beschermt onze intellectuele eigendom en kritieke security credentials.
- **Geïntegreerde Fysieke Toegangscontrole:**
    - **Wat:** Toevoeging van een **RFID-lezer** en een **elektromagnetisch slot**. Eventueel wordt ook een batterij backup toegevoegd.
    - **Waarom:** Dit transformeert het systeem van enkel monitoring naar actieve fysieke beveiliging. Toegang tot de (inhoud van de) behuizing is enkel mogelijk na succesvolle authenticatie via een geautoriseerde RFID-tag. Het slot fungeert als fysieke barrière. Dit voorkomt eenvoudige fysieke toegang om sensoren te manipuleren of de ESP32 te bereiken. De geautoriseerde RFID ID's worden veilig opgeslagen dankzij Flash Encryption. De batterij backup zorgt ervoor dat het slot vergrendeld blijft en het alarm kan functioneren, zelfs bij stroomuitval, wat de fysieke beveiliging significant verhoogt tegen sabotage van de stroomtoevoer.
- **Geavanceerde Manipulatiedetectie (Tampering):**
    - **Wat:** Toevoeging van een **6DoF IMU (Inertial Measurement Unit)** sensor en een **Afstandssensor**.
    - **Waarom:** Deze sensoren detecteren fysieke interactie:
        - **IMU:** Detecteert beweging, schokken, of kantelen. Dit signaleert of het apparaat wordt verplaatst, opengebroken of gemanipuleerd.
        - **Afstandssensor:** Monitort de afstand tot het deksel/deur. Detecteert of de behuizing (ongeautoriseerd) wordt geopend, zelfs als het slot geforceerd wordt.
- **Actieve Alarm- en Notificatiefuncties:**
    - **Wat:** Integratie van een **lokale speaker/buzzer** en **Discord Webhook** notificaties.
    - **Waarom:** Bij detectie van een ongeautoriseerde toegangspoging (bv. foute RFID), tampering (IMU/afstandssensor), of het openen zonder deactiveren:
        - **Speaker:** Genereert een lokaal akoestisch alarm als afschrikmiddel en directe waarschuwing.
        - **Discord:** Stuurt *onmiddellijk* een bericht naar een vooraf ingesteld kanaal, zodat beheerders op afstand real-time geïnformeerd worden over het incident.

- **Proactieve Server Beveiliging (Automatische Updates):**
    - **Wat:** Inzet van een tool zoals **Watchtower** binnen de Docker-omgeving op de Raspberry Pi.
    - **Waarom:** Zorgt ervoor dat de draaiende containers (Mosquitto, Node-RED, InfluxDB, Grafana) automatisch worden bijgewerkt naar de laatste stabiele/veilige versie. Dit minimaliseert de blootstelling aan recent ontdekte kwetsbaarheden in de serversoftware, zonder handmatige interventie.

3. Architectuur en Componenten



4. Beveiligingsstrategie Gelaagd Uitgelegd

Onze beveiliging is opgebouwd uit meerdere lagen:

1. **Hardware Integriteit (ESP32):**
    - Mechanisme: Secure Boot V2 + Flash Encryption.
    - Beschermt tegen: Ongeautoriseerde firmware, reverse engineering, diefstal van code en credentials (mTLS keys, Wi-Fi pass, RFID keys) via fysieke toegang tot de chip.
    - Hoe: Firmware moet digitaal ondertekend zijn. Flash is versleuteld met een unieke hardware sleutel.
2. **Communicatiekanaal Beveiliging:**
    - Mechanisme: MQTT over mTLS (direct ESP32 -> Broker).
    - Beschermt tegen: Afluisteren (man-in-the-middle), data manipulatie onderweg, ongeautoriseerde apparaten die data sturen (spoofing), verbinding met malafide servers.
    - Hoe: TLS zorgt voor encryptie. Wederzijdse authenticatie via certificaten valideert *beide* eindpunten.

3. **Fysieke Toegangscontrole & Detectie:**
    - Mechanisme: RFID authenticatie + Elektromagnetisch slot + IMU + Afstandssensor (+ Optionele Batterij Backup).
    - Beschermt tegen: Ongeautoriseerde fysieke toegang tot de behuizing, diefstal van het apparaat, sabotage van sensoren of ESP32, omzeiling door stroomonderbreking.
    - Hoe: Slot vergrendelt fysiek. Alleen geldige RFID opent/deactiveert. IMU detecteert beweging/schok. Afstandssensor detecteert opening. Batterij zorgt voor continue werking slot/alarm bij stroomuitval.
4. **Authenticatie & Autorisatie:**
    - Mechanisme: mTLS (device-server), RFID (user-device).
    - Beschermt tegen: Ongeautoriseerde systemen of gebruikers die toegang krijgen tot data of controlefuncties.
    - Hoe: Certificaten valideren apparaat/server. RFID ID's valideren gebruiker voor fysieke toegang.
5. **Alarm & Notificatie:**
    - Mechanisme: Lokale speaker + Discord Webhooks.
    - Beschermt tegen: Onopgemerkte incidenten. Werkt als afschrikmiddel en maakt snelle respons mogelijk.
    - Hoe: Directe lokale en remote signalering bij detectie van security events.
6. **Server Omgeving Hardening:**
    - Mechanisme: Docker containerisatie + Geautomatiseerde updates (Watchtower) + Database authenticatie + Netwerkisolatie.
    - Beschermt tegen: Compromittering van de server door kwetsbaarheden in software, cross-container aanvallen, ongeautoriseerde databasetoegang.
    - Hoe: Isolatie beperkt schade. Updates patchen kwetsbaarheden. Authenticatie en firewall regels beperken toegang.

5. Verwachte Uitdagingen

- Correcte implementatie van Secure Boot V2 en Flash Encryption (kritieke, onomkeerbare stappen).
- Beheer van X.509 certificaten voor mTLS (creatie, distributie, revocatie).
- Robuuste implementatie van RFID-authenticatie en veilig sleutelbeheer.
- Kalibratie van IMU en afstandssensor om valse alarmen te voorkomen.
- Integratie van alle componenten binnen ESP-IDF (vereist dieper begrip dan Arduino).
- Configuratie van de Docker-omgeving met mTLS en auto-updates.
- Implementatie en beheer van de noodstroomvoorziening (selectie componenten, laadbeheer, omschakeling, batterijmonitoring), indien toegevoegd.

6. Conclusie

Dit voorstel transformeert "Sensorio" van een basis IoT-monitoringsysteem naar een **robuust beveiligd systeem** met zowel cyber- als fysieke beschermingslagen. Door gebruik te maken van ESP-IDF's geavanceerde security features (Secure Boot, Flash Encryption), end-to-end mTLS communicatie, en de toevoeging van fysieke toegangscontrole en manipulatiedetectie (met eventuele batterij backup voor verhoogde veerkracht), pakken we de security-uitdagingen frontaal aan. De combinatie van deze maatregelen creëert een systeem dat significant beter bestand is tegen diefstal, uitlezen, en ongeautoriseerde toegang of manipulatie, en voldoet aan de gestelde eisen voor een uitzonderlijk beveiligd eindproject.
