SMASHBOX

**FUNZIONAMENTO GENERALE:**

SMASHBOX è un sistema che simula l’utilizzo delle classiche cassette di sicurezza della banca in modo più accessibile e smart rispetto allo scenario attuale.

Si compone di un N cassette di sicurezza (chiamate ACQUISITION) (il nostro prototipo ne comprende 2) e di una supplementare che funge da controllo per le altre (chiamata CENTRAL).

Ogni cassetta di sicurezza può essere registrata da un cliente tramite l’impronta digitale della mano, in modo da avere una chiave univoca e personale. Ogni cassetta ha la possibilità di tracciare lo stato interno della cassetta con dei parametri come temperatura, umidità, verificare se la cassetta è aperta o meno e se c’è stata una tentata manomissione. Il monitoring di questi parametri è utile per notificare il cliente in caso di tentativo di furto (manomissione) o di pericolo (aumento temperatura). Inoltre viene fornito un log con le diverse date e orari di apertura.

La notifica dei log e degli allarmi sfrutta un bot Telegram che viene registrato nel momento in cui il cliente acquista la possibilità di utilizzare una cassetta.

In caso di allarme il sistema si mette in quella che viene definita “SAFE-MODE”, ovvero tutte le cassette vengono bloccate, e solo un operatore può effettuare lo sblocco tramite un pulsante dedicato presente sul CENTRAL.

(FOTO SMASHBOX)

# MACRO COMPONENTI software

1. **LOCAL**: parte di acquisizione dei dati di monitoring e spedizione verso il BRIDGE.
2. **BRIDGE**: parte di recupero e inoltro verso il CLOUD.
3. **CLOUD**: gestione in cloud dei Digital Twin.
4. **CLIENT**: agenti esterni che possono accedere al CLOUD tramite interfaccia

# DETTAGLIO COMPONENTI

**LOCAL**

Questa sezione è divisa in due parti:

* Acquisition: la board scelta per leggere i dati dai sensori. Gestisce strettamente l’acquisizione dei dati dai sensori e l’inoltro tramite un canale (seriale o Wi-Fi) verso il Bridge.
* Central: Un Arduino dedicato gestisce il riconoscimento dell’impronta digitale. Centralizzata perché ci sarà un unico riconoscitore per tutte le cassette.

**ACQUISITION**

Questo componente è composto da

* Arduino
* Sensore di temperatura e umidità (Adafruit\_AHTX0)
* Accelerometro (ADXL345)
* 2 sensori di presenza (sportello e presenza oggetto)
* Un elettromagnete sostiene il blocco dello sportello quando è in modalità lock.

**CENTRAL:**

L’altra parte del LOCAL, quella centralizzata che effettua sia le registrazioni che le richieste di sblocco da parte dell’utente.

Questa è composta da:

* Sensore di impronte (Adafruit\_Fingerprint)
* Display di servizio
* Un pulsante per resettare la safe-mode (v. dopo)

**BRIDGE**

Il Bridge è uno script Python che ha due endpoint che lavorano in sinergia: uno lato LOCAL e uno lato CLOUD. L’idea di base è che il BRIDGE passi continuamente i dati che arrivano da LOCAL verso CLOUD (e viceversa). Il pacchetto di monitoring arriverà in un'unica trasmissione gestita da un protocollo (v. dopo). Il BRIDGE spacchetta tramite un Parser i dati ed effettua i comandi di update (HTTP/MQTT) verso il CLOUD in maniera separata.

Il BRIDGE è anche il punto di comunicazione tra CENTRAL e le varie ACQUISITIONS, dal momento che ci sono casi in cui il CENTRAL deve comunicare un informazione alle possibili N boards che sono le varie cassette.

**CLOUD (THINGSBOARD)**

La piattaforma scelta per il Cloud è Thingsboard. Qui avverrà il monitoring dei dati nel corrispondente Digital Twin e la gestione dei trigger scaturiti sulla base dei dati che arrivano dal Bridge.

Sostanzialmente raccoglie di ogni cassetta e comunica a tutte le altre di intervenire in caso di Safe Mode.

**TELEGRAM BOT**

E’ un endpoint in cui l’utente viene notificato di aperture della propria cassetta, di eventuali allarmi. Inoltre viene data la possibilità al cliente tramite comando al bot di capire quali sono stati i vari log della propria cassetta registrata.

***/\* MQTT SERVER(MQTT)***

*C’è un ulteriore blocco, che sarà un broker MQTT sul server dove appoggiarsi per scambiare dati tra BRIDGE e THINGS. L’idea è che i trigger di allarme siano mandati su questo server (test.mosquitto.org) e recuperati dal BRIDGE sullo stesso. Questo perché Thingsboard nella sua versione free possiede soltanto la versione per scambiare i dati verso il server e non dal server.*

***SERVER CUSTOMERS (FIREBASE)***

*L’idea di accedere ad un server per visualizzare il log e di poter sottoscrivere per una nuova cassetta potrebbe essere fatta tramite una app che fa delle richieste ad un Server dedicato in cui sono contenuti i log delle cassette. Lo stesso server contiene anche la lista dei clienti registrati che tentano di accedere allo sblocco. Questo non solo può essere usato per il log, ma anche come backup dei clienti registrati sul central per accedere alle cassette. Tutto questo sarà gestito con un Real Time Database di Firebase.*

*Lo schema generale potrebbe essere quello linkato al repository: \Doc\Blocks \*/*

# FUNZIONAMENTO

**ANALISI GENERALE DEL SISTEMA:**

Il sistema è composto da tre componenti principali: due cassette di sicurezza (ACQUISITION), una cassetta centralizzata (CENTRAL), un bridge software (scritto in Python) e un bot Telegram.  
Quando viene acceso, le cassette ACQ e la centralina CNTRL entrano in una fase di attesa in cui si mettono in comunicazione con il bridge. Il bridge identifica correttamente le unità e abilita la centralina a iniziare il processo di registrazione delle impronte digitali.  
In questa fase, il sistema associa un'identità a ogni cassetta (simulando la registrazione di un utente) e inizia la comunicazione tra le parti.

Una volta assegnati gli ID, le cassette iniziano a rilevare vari parametri ambientali e di sicurezza. I dati vengono inviati regolarmente al bridge, che li inoltra al server. Tali cassette sono sbloccabili tramite il riconoscimento dell’impronta digitale precedentemente registrata.

Il sistema può funzionare in due modalità: normale (IDLE) e di allarme (SAFE MODE). La modalità SAFE MODE viene attivata se vengono rilevati eventi sospetti come urti, aperture non autorizzate, o valori anomali di temperatura/umidità. In tal caso, il sistema reagisce localmente (con LED e blocchi fisici) e invia una notifica anche al bot Telegram.

**ANALISI TENICA DEL SISTEMA:**

Il sistema avviene su fasi che si succedono una dietro l’altra.

**RECOGNITION**

Allo startup, il BRIDGE effettua una scansione delle seriali che sono collegate al sistema. L’obiettivo è capire quali delle seriali rappresenta ACQ e quale rappresenta CNTRL.

Per farlo, CNRTL manda sulle seriale un pacchetto identificativo (7 byte) di tutti 0 e come ACQ si mettono in “ascolto” del BRIDGE stesso. Valutando quale serial ha inviato il pacchetto identificativo, BRIDGE riesce a capire quali sono le seriali delle cassette, e quale quella di CNTRL.

*from central\_proto.ino, in setup ()*

  /\* Sending a "0" to let the bridge recognize Central \*/

Serial.write(idle\_packet\_init, sizeof(idle\_packet\_init));

A questo punto, la parte CNTRL che era in attesa viene risvegliata dal bridge tramite l’invio sulla seriale (quella appunto riconosciuta) un carattere identificativo per procedere (“@”) . Le ACQ restano ancora in attesa.

Se Central viene alimentato prima dell’avvio del bridge stesso, verrà visualizzato il messaggio seguente sul display di servizio:

“Wait for bridge…”

**ENROLL**

Quello che le ACQ stanno attendendo è l’ID. Poiché si sta simulando il primo power-on del sistema, le cassette fin quando non sono registrate, non possono partire ad acquisire i dati.

*from acquisition.ino, in setup ()*

Immagine che contiene testo, schermata, Carattere, software

Il contenuto generato dall'IA potrebbe non essere corretto.

Come si nota, ogni singola ACQ aspetta un pacchetto identificativo di 10 byte da parte del BRIDGE, con il primo valore (buffer[0]) contenente l’ID stesso. Fino a quando non arriva questo pacchetto, il sensore non è partito.

Il CNTRL, risvegliato, entra in fase di enroll, dove aspetta due registrazioni di due cliente differenti (come le cassette del sistema). La registrazione avviene tramite impronta digitale, che viene scannerizzata in due passaggi per accertarsi una corretta cattura dell’immagine.

Ogni volta che viene registrata una cassetta, da CNTRL viene inviato verso BRIDGE un pacchetto con di 2byte, uno che identifica l’header (PACKET\_ENROLL) e l’altro byte l’ID stesso. Questo ID è generato casualmente all’interno del CNTRL in un range compreso tra 1 e 128.

Il bridge riceve questo pacchetto con l’ID, lo estrae e lo invia nel pacchetto di 10 byte (tutti 0 tranne il meno significativo contenente proprio l’ID) ad una delle box non ancora registrate. Si traccia quali sono registrate e quali no, tramite un dizionario Python in RAM.

Successivamente, viene ripetuta la registrazione di una nuova impronta in modo da replicare il processo per il secondo cliente/cassetta.

A questo punto ACQ riceve i 10 byte, assegna l’ID e termine la sua routine di setup().

Anche CNTRL termina la sua prima parte della routine di setup().

Immagine che contiene testo, diagramma, schermata, linea

Il contenuto generato dall'IA potrebbe non essere corretto.

**3.2 SAFE MODE**

Prima di descrivere cosa fanno nella loro parte applicativa le cassette ACQ e il CNTRL, è necessario definire in quali modalità il sistema (nell’intero) può ritrovarsi:

* IDLE (funzionamento normale)
* SAFE MODE (caso di allarme)

Il secondo caso viene triggerato dal server: quando i dati di ACQ giungono a questo, vengono elaborati. In particolare, se viene effettuata una manomissione (accelerometro) oppure temperatura e umidità superano alcune soglie, tramite un broker MQTT viene inviato al BRIDGE un messaggio, il quale notifica l’intera parte “local”: sia CNTRL che ACQ.

In particolare, questo avviene mandando da BRIDGE a LOCAL un carattere speciale (uno per CNTRL che sarà “\_” e uno per ACQ che sarà “\*”) che nei vari main loop degli applicativi verra catturato e farà in modo che si comportino nella modalità SAFE MODE (dettagli nelle sezioni successive, descritti all’interno dei main loop).

A livello generale, si può dire che una volta ricevuto un allarme, l’intero sistema (ACQ e CNTRL) entrano entrambe in SAFE MODE, con comportamenti differenti.

CNTRL rimane in tale modalità, con l’unico task di mostrare sul proprio display di servizio il messaggio di un intervento manuale richiesto:

MANUAL OP REQUIRED!

Tale operazione richiesta, è la pressione di un pulsante da parte di un operatore (previo codice identificativo, che non è stato concepito ma si può aggiungere) che notifica il sistema della risoluzione dei problemi che hanno indotto a scaturire l’allarme.

Per quanto riguarda le ACQ, anch’esse in modalità SAFE MODE, notificano l’utente tramite il lampeggio dei led e modificano il contenuto del pacchetto di 7 byte che ciclicamente inviano.

Un messaggio di allarme viene inviato anche al BOTTG nel momento in cui viene triggerato un allarme da parte del server.

Immagine che contiene testo, diagramma, schermata, linea

Il contenuto generato dall'IA potrebbe non essere corretto.

**MAIN LOOP ACQUISTION:**

Una volta che il CNTRL ha assegnato l’ID, ogni singola ACQ rileva in real-time i vari dati dai sensori installati all’interno delle cassette. In particolare:

* Temperatura + Umidità
* Accelerometro
* Presenza oggetto
* Rilevamento sportello aperto o no

I dati vengono mandati sulla seriale ad ogni ciclo al BRIDGE in un pacchetto di 7 byte:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID (1 byte)** | **Presence (1 byte)** | **Temperature (1 byte)** | **Humidity (1 byte)** | **Infra (1 byte)** | **Lock (1byte)** | **Open (1byte)** |

Il main loop di acquisizione è continuo, alla fine di ogni ciclo verranno comunque spediti al BRIDGE i 7 byte. Quello che può cambiare è il contenuto di questo pacchetto, sia nella rilevazione (i dati cambiano in maniera naturale sulla base di quello che devono rilevare), oppure se si è in modalità SAFE MODE.

Nel caso normale, i 7 byte saranno quelli che descriveranno cosa è stato acquisito durante la rilevazione dei sensori. Al BRIDGE poi il compito di girarli verso il server.

Se altrimenti la modalità è SAFE MODE, i 7 byte saranno tutti a 0, eccetto per il campo “Lock” che avrà valore 1. Fin quando tale modalità non viene rilasciata, ciclicamente il BRIDGE riceverà questo tipo di pacchetto.

**0000010**

**GESTIONE DEL CAMPO INFR:**

Nella simulazione dell’evento di manomissione (si scuote la scatola) la ACQ che rileva tale infrazione(accelerometro), settasse il campo “Infra”, che una volta elaborato dal server triggerava il pacchetto che serve per entrare in SAFE MODE.

Essendo il ciclo di ACQ molto più veloce del tempo necessario per settare il flag *safe\_mode*, quello che accadeva era che il BRIDGE vedesse arrivare in ricezione più pacchetti con il campo “Infra” settato a 1, con conseguente invio di due pacchetti di allarmi consecutivi.

Per evitare tutto ciò, alla prima rilevazione di manomissione da parte di ACQ, viene settato un flag booleano (*active\_infr*) che permette di effettuare o meno all’interno del loop di acquisition, la acquisizione dei valori dell’accelerometro:

Come si nota, a riga 97, dopo la spedizione del pacchetto ciclico, si analizza il valore del campo di infr:

Nel momento in cui viene rilevato una manomissione, si setta il booleano.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Una volta tornati nel loop principale, l’update dell’accelerometro avverrà soltanto se tale flag non è stato appunto settato in precedenza.

Immagine che contiene testo, schermata, software, Software multimediale

Il contenuto generato dall'IA potrebbe non essere corretto.

A completare questo ciclo, l’operazione (che in realtà è la prima di tutte) che si effettua (update\_lock\_field) ha lo scopo di mettersi in ascolto del BRIDGE. In particolare, ACQ aspetta 1 singolo carattere, sulla base del quale verranno prese diverse azioni:

* CHAR\_UNLOCK (‘&’): E’ il carattere che permette lo “sblocco” da parte dell’utente dopo che è stato riconosciuta una impronta digitale. Si rilascia il magnete (porta apribile, flag *lock = 0*) e si accende il led di servizio della cassetta.
* CHAR\_SAFE\_MODE (‘\*’): Carattere inviato nel caso il server triggeri un allarme legato o alla soglia di temperatura o alla rilevazione di una manomissione. Si attiva il magnete (porta apribile) e si accende il led di servizio della cassetta. In questo momento, viene settato il flag *safe\_mode e lock,* che serviranno poi nel loop principale per decidere di mandare il pacchetto dedicato a questa modalità.
* CHAR\_RE\_SAFE (‘/’): per rilevazione di uno sblocco manuale da parte di un operatore dalla modalità SAFE\_MODE. Viene resettato (false) il booleano per l’infrangment (*active\_infr = false),* così come resettato *safe\_mode.* Il magnete viene attivato in modo che la cassetta una volta appoggiatta si possa lockare.

**MAIN LOOP CENTRAL:**

Il CNTRL gestisce più operazioni all’interno del main loop. Possiamo dividere tali operazioni (sostanzialmente di richiesta di azione da parte di CNTRL verso il BRIDGE) in 3 possibilità (descritte più sotto). Nel caso però in cui, nessuna di queste 3 azioni venga richiesta, CNTRL notifica ciclicamente il bridge di essere nello stato IDLE, con un pacchetto di 2 byte. *(d’ora in avanti questi 2 byte di CNTRL verranno definiti come ACTION\_PACKET, che contengono 1 byte per l’header dell’operazione richiesta e 1 byte per i dati che servono per quell’operazione)*

Nel caso IDLE, l’header è identificativo della modalità (PACKET\_IDLE) e il campo dati 0.

Le altre 3 modalità (di richiesta), sono invece le seguenti

* Enroll Mode: questa modalità in realtà è l’ultimo step a conclusione della routine di setup(). Sostanzialmente il sistema è in attesa di 2 registrazioni di impronte digitali che simulino due clienti. Se non vengono registrate, l’applicativo di CNTRL non parte.

Sul display:

Waiting enroll...

Nel caso in cui c’è una corretta registrazione:

"Enrolled!"

Questo è collegato a quanto già citato nella sezione ENROLL per l’ACQUISITION. E’ il momento in cui c’è una registrazione completata con successo, con conseguente invio verso il BRIDGE con un ACTION\_PACKET (2 byte) contenente l’header identificativo dell’azione in corso (PACKET\_ENROLL) e l’id generato randomicamente in quel momento nel campo dati. Il BRIDGE, dopo averlo ricevuto, lo girerà poi in uno da 10 byte verso ACQ.

* Fingerprint Mode: È la modalità in cui CNTRL passa la maggior parte del tempo. E’ l’attesa per cui si mette in ascolto di un riconoscimento di una impronta digitale.

Sul display:

Waiting finger..

* Check Mode: quando un utente appoggia il dito, si procede con la fase di riconoscimento. Se c’è match tra quelle che sono state registrate, allora si entra nella modalità sotto descritta. In quest’ultimo caso, viene inviato un pacchetto ACTION\_PACKET contenente l’header identificativo dell’azione in corso (PACKET\_CHECK) e l’id della cassetta che è stata riconosciuta, ovvero che si vuole sbloccare.

Sul display verrà visualizzato per un istante l’ID della cassetta sbloccata.

**CLOUD:**

Per la fase di recupero e monitoraggio dei dati provenienti dal bridge, ho scelto di utilizzare la piattaforma ThingsBoard in modalità "Live Demo". Questa scelta ha permesso di sfruttare un ambiente già predisposto per l’acquisizione in tempo reale, senza la necessità di configurazioni server complesse.

ThingsBoard si è rivelato particolarmente utile per visualizzare i dati raccolti, applicare soglie in modo dinamico e inviare automaticamente un messaggio MQTT quando tali soglie venivano superate. In questo modo si è potuto testare e validare rapidamente il meccanismo di notifica e controllo, riducendo i tempi di sviluppo e mantenendo un elevato livello di flessibilità.

Per la fase di prototipazione, sono stati creati due dispositivi virtuali sulla piattaforma, ciascuno configurato con il proprio access token per poter comunicare con il broker MQTT integrato. Questi dispositivi inviano periodicamente dati simulati al broker, rappresentando il flusso proveniente da sensori reali.

È stata realizzata una Dashboard personalizzata per la visualizzazione in tempo reale dei dati raccolti, facilitando il monitoraggio dello stato dei dispositivi anche in fase di Debug.

Immagine che contiene testo, schermata, software, Software multimediale

Il contenuto generato dall'IA potrebbe non essere corretto.

Infine grazie ad una Rule Chain, è stato possibile definire le condizioni di trigger in base al superamento di soglie preimpostate. Quando una condizione di allarme viene rilevata, la Rule Chain genera un messaggio MQTT in uscita, inviato a un broker esterno (Mosquitto).

Il bridge, iscritto come subscriber a tale broker, riceve questi messaggi e agisce di conseguenza, completando così il ciclo di notifica e risposta descritto precedentemente.

Immagine che contiene testo, schermata, diagramma, mappa

Il contenuto generato dall'IA potrebbe non essere corretto.

# SVILUPPI FUTURI

**HARDWARE:**

* Trasferimento da scheda filata a PCB
  + PCB CAD designer
  + Da Arduino a un microcontrollore/ASIC
* Comunicazione Wireless: tramite transceiver Wifi effettuare la comunicazione CNTRL-ACQ senza l’utilizzo dei cavi fisici, impossibile in una produzione reale del prodotto.
* Efficienza energetica: soluzioni di risparmio energetico (cassetta non è in modalità sleep)

**SOFTWARE:**

* Refactoring dell’architettura sulla base di HAL
  + Protezione dall’obsolescenza
  + Flessibilità e modularità fortemente aumentata

L’obiettivo è rendersi indipendente dalla piattaforma. In un contesto di mercato in cui l’obsolescenza dei prodotto elettronici (non solo microcontrollori) è all’ordine del giorno, l’idea è di cercare di avere un software più flessibile e modulare per cercare di raggiungere il massimo del riutilizzo del codice.

Attualmente il codice dell’applicativo e di basso livello è tutto appartenente ad Arduino. Se Arduino (o il microcontrollore in generale) viene dismesso dalla produzione, si è costretti a riscrivere tutto il codice.

Passando ad una architettura Multilayer, l’applicativo utilizzare un interfaccia comune (INT) che viene concretizzata in file appartenenti alla piattaforma, con i suoi driver specifici. Se cambia la piattaforma, la parte di applicativo non cambia, rimane invariata. Verrà solo riscritta soltanto la parte di basso livello legata alla nuova macchina.

Immagine che contiene testo, schermata, Rettangolo, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

ESEMPIO RAPIDO:

(vedi ppt)

* Fw update da remoto