Progetto di IoT

INDUSTRIAL
SECURE
CHAT

GASPAROLLOMARTIN

a.a. 2021/2022

Introduzione

Siamo due studenti che frequentano la facoltà di IoT, Big Data & Web presso l'Università degli Studi di Udine; questo progetto è nato fondendo le conoscenze che abbiamo acquisito grazie ai corsi curati dai docenti e alle realtà aziendali che tramite il tirocinio e il lavoro abbiamo visitato.

Nella stragrande maggioranza dei casi ciò che fa diventare un' azienda il punto di riferimento per le altre nel suo settore: sono le informazioni che lei possiede.
Per informazione intendiamo quel gruppo di dati che vanno a costituire una ricetta nel caso della pasticceria oppure un part program nel settore della lavorazione dei materiali.

Tutte le aziende sono capaci (tramite un cospicuo investimento di denaro) di acquistare le stesse attrezzature della azienda master nel settore, però non tutte sono capaci di utilizzare le strumentazioni al meglio per poter rendere il prodotto finale il migliore sul mercato.
Queste informazioni sono il carburante dell'azienda e in molti casi non gli viene attribuito il giusto peso, facendole transitare dall'ufficio di progettazione fino al reparto di produzione mediante l'utilizzo di un postit oppure tramite e-mail, esponendole al mondo esterno con una tranquillità invidiabile.

Con questo progetto abbiamo cercato di rendere il più sicuro possibile il sistema che veicola queste informazioni da un lato all'altro dell'azienda.

Suddivisione dello Sviluppo

Fondamentalmente entrambi abbiamo toccato ogni ambito dello sviluppo del progetto al 50%, cercando di realizzare un prodotto più valido e supervisionato in ogni sua sfaccettatura da più persone.

Un esempio concreto

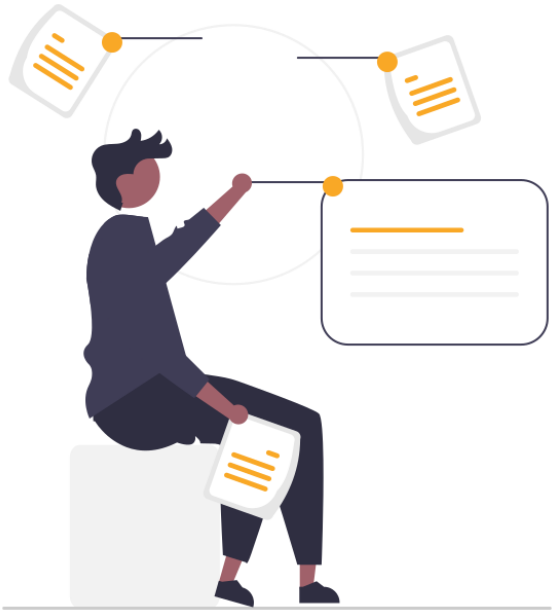
Immaginiamo di trovarci nella migliore azienda che lavora l'acciaio; essa è diventata leader del settore probabilmente perché :

- Ha un piano pubblicitario molto valido
- Ha dei prezzi buoni rispetto alle altre (poco probabile)
- Offre un servizio che le altre aziende non offrono
- Offre un prodotto finito di una qualità superiore

Quest'ultima probabilmente sarà una delle caratteristiche che ha portato l'azienda in cima alla classifica.

Per poter ottenere questo risultato, l'azienda è riuscita ha trovare il giusto compromesso tra:

- scelta del materiale da lavorare
- temperatura di riscaldo da impostare nel forno
- tempo di riscaldo del pezzo in lavorazione nel forno
- forza corretta da applicare nella battuta tramite il bilanciere
- ecc…



Molte aziende potranno acquistare lo stesso modello di forno e di bilanciere ma difficilmente riusciranno a reperire le informazioni sopra elencate.

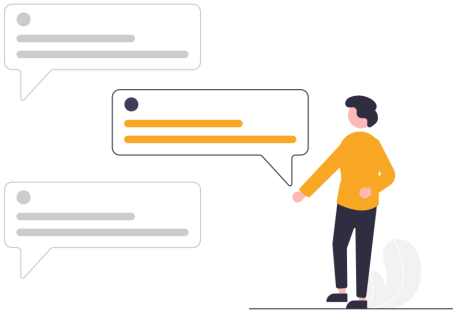
Il nostro prodotto

Il prodotto che dovrebbe risolvere questo grosso problema lo abbiamo soprannominato ISC, che è l'acronimo di Industrial Secure Chat, ed è un piccolo portachiavi che ospita all'interno di esso una WebChat totalmente scollegata da Internet.
Le informazioni transitano, codificate, dal dispositivo direttamente al portachiavi senza passare per server posizionati in zone a noi sconosciute.
Per comunicare basterà con il proprio dispositivo connettersi alla rete Wi-Fi del portachiavi e collegarsi tramite il browser alla "WebApp".

3 strati di sicurezza

"hw propietario"
l'HW è completamente in tuo possesso e ti permette di avere il pieno controllo.
Nel caso in cui si verifichi qualche tentativo di intromissione nella rete puoi spegnere il dispositivo e fare in modo che la rete e il server siano temporaneamente fuori uso.

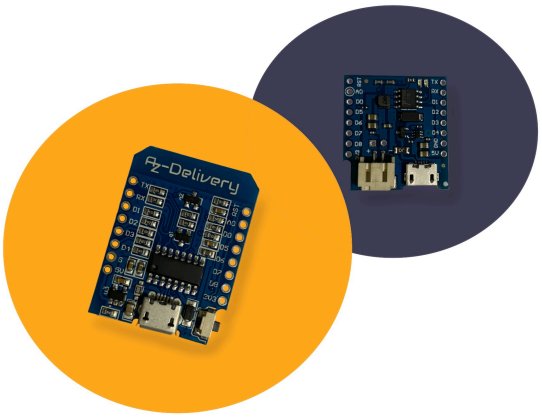
SSID non in broadcast e password
Una volta acceso il portachiavi, non sarà possibile trovare immediatamente la rete facendo una ricerca rapida dalle impostazioni perché l'SSID non è impostato in Broadcast, quindi bisognerà conoscerlo ed aggiungere la rete manualmente; inoltre la rete Wi-Fi è protetta da password.



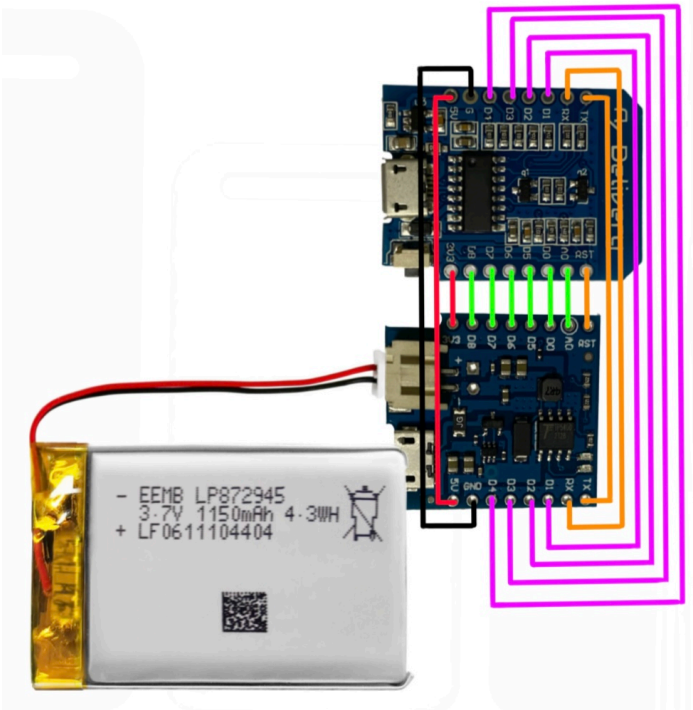
Key di Codifica e Decodifica
Una volta effettuata la corretta connessione al portachiavi bisognerà essere a conoscenza della chiave con cui vengono codificati i messaggi.
Per aumentare ulteriormente la sicurezza abbiamo scelto di introdurre una codifica simmetrica lato client (con scambio della chiave a livello vocale); questo sistema permette di tenere lontana la chat (che è memorizzata nel server) dalla chiave di decodifica.
In questo modo anche se qualcuno si impossessasse del HW e riuscisse a fare un DUMP della memoria non troverebbe alcuna traccia della chiave.

Come è realizzato il portachiavi

Fondamentalmente il progetto è costituito da un Server Web equipaggiato ad una NodeMCU che tramite GET e POST permette lo scambio di informazioni (codificate lato client) tra il server stesso e i clients che si collegano.
Alla connessione, il server, restituisce un interfaccia Web (sui browser dei client) che permette di fruire del servizio.



Relaizzazione HW



Realizzazione SW

```
45    WiFi.mode(WIFI_AP);
46    WiFi.softAPConfig(apIP, apIP, IPAddress(255, 255, 255, 0));
47    WiFi.softAP(wifiName, pwd, 1, true, 7);
48    dnsServer.start(DNS_PORT, "*", apIP);
49    webServer.begin();
```

```
81  void HandleSendMessage()
82  {
83    if (webServer.hasArg("message"))
84    {
85      String message = webServer.arg("message");
86      fileWrite(messagesFile, message + "\n", "a+");
87      webServer.sendHeader("Access-Control-Allow-Methods", "POST");
88      webServer.sendHeader("Access-Control-Allow-Origin", "*");
89      webServer.send(200, "text/plain", "Message Sent");
90    }
91  }
```

Semplicemente controlla che ci sia un messaggio e lo scrive all'interno del file che funge da memoria.
Il messaggio viene inviato al server attraverso una chiamata POST.

```
93  void ShowMessages()
94  {
95    String messages = fileRead(messagesFile);
96    webServer.sendHeader("Access-Control-Allow-Methods", "GET");
97    webServer.sendHeader("Access-Control-Allow-Origin", "*");
98    webServer.send(200, "text/plain", messages);
99  }
```

La funzione *ShowMessages()* inoltra l'intero contenuto del file di memorizzazione (ovvero, i messaggi codificati) e li manda al client che provvederà a decodificarli e formattarli adeguatamente per l'impaginazione.

```
118    void PowerOff()
119    {
120      webServer.send(200, "text/plain", "Goodbye");
121      UsrAlertLed(5);
122      ESP.deepSleep(0);
123    }
```

La funzione di spegnimento del portachiavi è realizzata richiamando il metodo *PowerOff()* che, una volta richiamato, manda la scheda ESP in **deepSleep** e risponde al front-end con una conferma di spegnimento riducendo così i consumi.

```
109  void ShowStatus()
110  {
111    webServer.sendHeader("Access-Control-Allow-Methods", "GET");
112    webServer.sendHeader("Access-Control-Allow-Origin", "*");
113    batteryResult = (battery == 1024 ? ">3.3" : (battery < 100 ? "Usb" : String(GetVolt()))) + " Volt";
114    webServer.send(200, "text/plain", batteryResult);
115    UsrAlertLed(5);
116  }
```

Questa funzione di base restituisce il livello di batteria e se l'ESP è alimentato tramite USB oppure a batteria. Nel caso in cui sia alimentato a batteria restituisce il livello di batteria attraverso la funzione *GetVolt()*.

La *GetVolt()* semplicemente effettua una proporzione: viene costruita sapendo che il livello di batteria viene restituito con un valore da 0 a 1023 (valore analogico) e sapendo che il valore 967 (valore restituito) vale circa a 4.08V (valore misurati).

```
453   const Encode = salt => {
454     const textToChars = text => text.split('').map(c => c.charCodeAt(0));
455     const byteHex = n => ("0" + Number(n).toString(16)).substr(-2);
456     const applySaltToChar = code => textToChars(salt).reduce((a, b) => a ^ b, code);
457
458     return text => text.split('')
459       .map(textToChars)
460       .map(applySaltToChar)
461       .map(byteHex)
462       .join('');
463   }
```

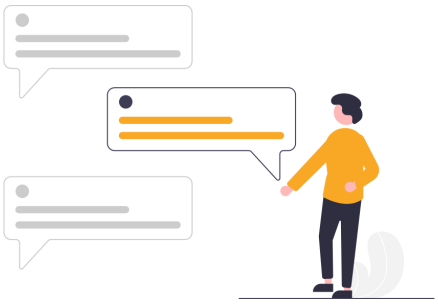
```
465   const Decode = salt => {
466     const textToChars = text => text.split('').map(c => c.charCodeAt(0));
467     const applySaltToChar = code => textToChars(salt).reduce((a, b) => a ^ b, code);
468     return encoded => encoded.match(/.{1,2}/g)
469       .map(hex => parseInt(hex, 16))
470       .map(applySaltToChar)
471       .map(charCode => String.fromCharCode(charCode))
472       .join('');
473   }
```

La codifica dei messaggi viene fatta con cifratura a chiave simmetrica con cifrario di Vigenere e salva i messaggi codificati all'interno di un file salvato nell'ESP che funge da memoria.
La chiave viene passata in fase di "login".
La fase di Encode procede nel seguente modo:

- prende il messaggio e lo converte nel corrispondente valore decimale (ASCII);
- applica XOR carattere per carattere;
- converto l'intera stringa in codifica esadecimale.

La decodifica del messaggio viene fatta applicando lo stesso processo inverso effettuando la XOR tra messaggio codificato e chiave.

Si è deciso di usare un cifrario di Vigenere per il fatto che al momento l'ESP non è in grado di collegarsi ad Internet o accettare file esterni.

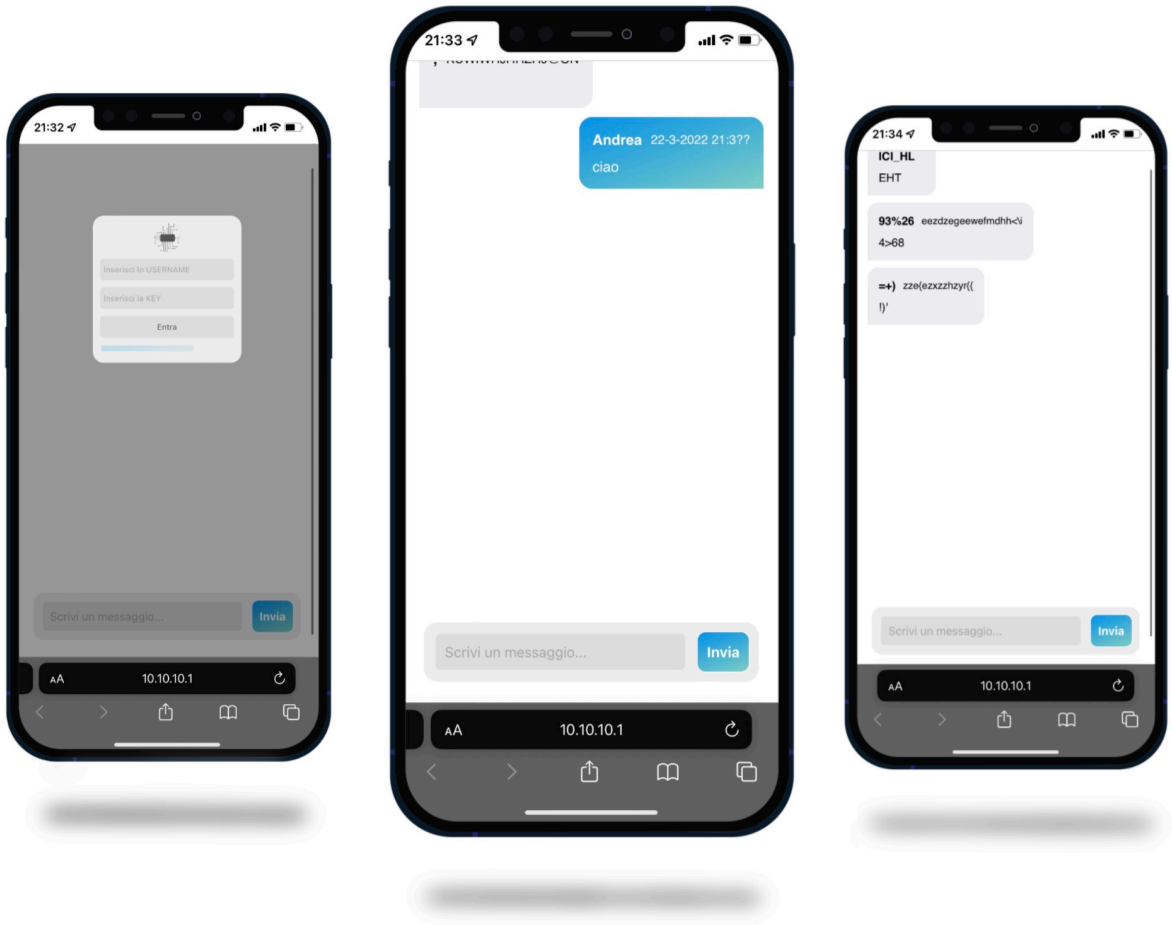


Manuale del SW

Per usufruire del servizio bisogna conoscere l'SSID della rete Wi-Fi della scheda e la relativa password.
Conoscendo questi 2 dati si può procedere aggiungendo manualmente la connessione alla lista delle reti Wi-Fi del proprio dispositivo; una volta connessi potrebbe venire richiesto di restare connessi a una rete senza Internet.
Una volta connessi alla rete basterà recarsi via Browser all'indirizzo 10.10.10.1 ed effettuare il login inserendo il nome Utente e la Key di Cifratura concordata con gli altri partecipanti alla Chat.

ERRORI comuni

Se non visualizzate la schermata di Login della Chat probabilmente il dispositivo si sarà scollegato a causa della mancanza di Internet; se invece visualizzate correttamente la Piattaforma ma leggete dei messaggi incomprensibili, probabilmente avrete inserito la Key di Cifratura NON Corretta.



Limiti e Conclusioni

Ammettiamo che la realtà dei fatti viene leggermente estremizzata per cercare di aumentare la comprensione del pericolo e che il prodotto finale di questo progetto ha grosse limitazioni dettate dalla distanza, però volevamo cercare di porre soluzione in maniera semplice e appetibile a tutte le tipologie di utenza ad una possibile problematica aziendale riscontrata nella realtà ed unire competenze acquisite tramite lo studio a competenze acquisite mediante il lavoro o il tirocinio.

Future Espansioni

In futuro vorremmo sostituire l'attuale algoritmo di cifratura (simil cifrario di Vigenere) con AES, ottimizzare i consumi della batteria, ridurre le dimensioni dell'hardware e implementare il riconoscimento di file (tipo css e js) esterni alla pagina.

Studenti : Gasparollo / Martin
Matricola : 143225 / 147800
Facoltà : IoT, Big Data & Web

Università degli Studi di Udine
a.a. 2021/2022
Corso : IoT