

## CONTINUIAMO MQTT:

Ci Sono 3 lvl di Qualità del Servizio (AFFIDABILITÀ) diversi  
tutto Va Bene

da NON Confondere con la SICUREZZA

Tutto ciò che può Andare Storto Per Causa  
di Operazioni Malevoli Umane

MQTT Basato Su Lvl Trasporto (che è Già Affidabile)

Affidabilità HTTP, Mai Sentita? No Perché Mi Basavo Su TCP e Quindi  
non Avevo Nessun Problema.

MQTT Parlo di Affidabilità per un Motivo Ben Specifico, Ovvero:  
che Qui Abbiamo 3 Processi e non più Solo 2 e Quindi va  
ad Aggiungere Complessità.

Potrebbe Essere L'END TO END il Problema, Non Le Singole Transazione.

All' UTENTE Importa il Passaggio di INFORMAZIONI Corretto e non  
il BROKER (interessa Solamente ai TECNICI)

↳ Oggetto Intermedio a Livello Applicativo (come RANGE EXTENDER A)  
LVL FISICO

Se Va giù il BROKER (che è lui che crea Problemi di Affidabilità)

Dopo PUBLISH connessione Viene Chiusa Rispetto al SUBSCRIBER che Rimane Aperta in Attesa in una NOTIFY da Parte del BROKER

→ Heart/Keep Alive per non Essere Rimossa dal NAT

Se Provo a fare PUBLISH e Broker è Già Giù allora me ne Accorgo, ma Se ho fatto publish, chiudo Connessione e poi BROKER Va giù, allora Nessuno Se ne Accorge

**QoS Lvl 0 (at most one)** ⇒ Significa che non c'è controllo e c'è Affidabilità Nulla.

A Lvl. END TO END il Dato Arriva al Massimo una Volta...Quindi può non Arrivare. Non Arriva / può Essere DUPLICATO, Quindi non Viene ATRASMESSE.

→ devo Poi Capire Quando Viene Ritrasmesso Per Distinguere i PACHETTI DUPLICATI da Quelli "NORMALI" (Anche Perché io Potrei Voler Inviarti 2 Volte Lo Stesso Dato)

SUBACK/connack .....

In MQTT Sono Stati Inscritti degli **ACK/Scambi di Messaggi** di Servizio che Viaggiano Su TCP per Garantire Certi livelli di Affid.

**QoS Lvl 1 (At least One)** ha Bisogno di più Scambi, Sforzo Maggiore Per La RETE (Ogni Messaggio MQTT Genererà degli

ACK TCP, Quindi ha un EFFETTO VALANGA)


Qui DEVE Arrivare e può Arrivare più di una Volta, devo Quindi Aggiungere un Qualcosa Per distinguere i Messaggi duplicati come ad Esempio può Essere un TIMESTAMP.

❖ QoS Lvl 2 (Esattamente 1 Volta)  $\Rightarrow$  Proprio Come TCP che Effettua Ritrasmisione fino a che non Viene Scambiato Messaggio Con Successo 1 Volta e Basta.

Così facendo ARRIVERÀ SEMPRE e SOLAMENTE 1 Volta.

Si fa Riferimento a Messaggi END TO END perché Tra Singoli Scambi c'è Già TCP.

Nella Sicurezza Si Applica Stessa Cosa, Posso Mettere TLS/SSL sotto MQTT e così diventa sicuro

Funziona Solo su TCP 

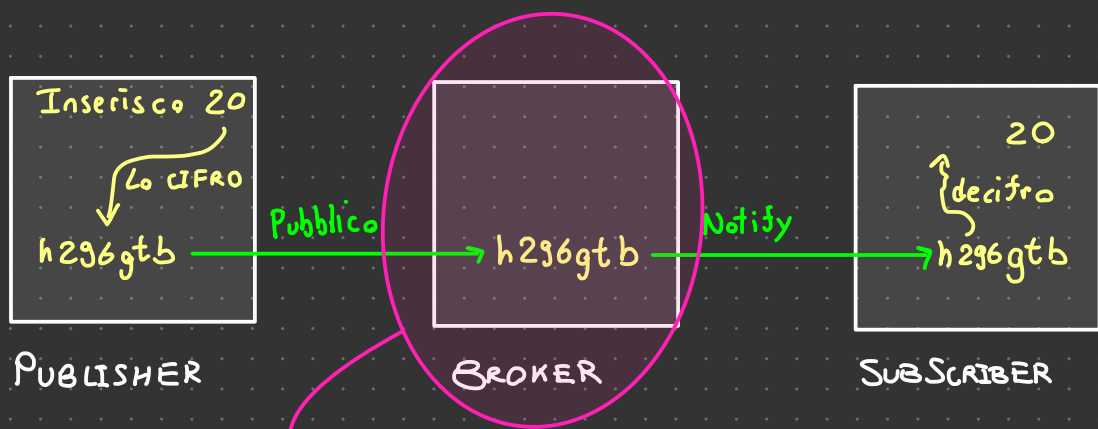
Non MODIFICO nulla a LVL Applicativo, chiamo libreria e WRAPPO con TLS

Diventa Sicuro nei Singoli Viaggi ma NON END TO END.

Tra PUB e BROKER è Sicuro Sì Ma poi dopo Quando il dato lo ha in PANCIA il BROKER lì cade il tutto perché Se Attacchi il BROKER Non Sarei Protetto.

Una Soluzione Sarebbe Potrebbe Essere Quella di Negoziare una Chiave per Cifrare il DATO per RENDERLO un DATO Senza Senso e Solo Quando Arriva al SUBSCRIBER (che sa la KEY)

Viene DECIFRATO.



→ Se VENISSE attaccato non Riescono a Capire che Cosa Sia Veramente Quel DATO, Solo PUBLISHER & SUBSCRIBER Lo Sanno Interpretare

Questo è un ESEMPIO, MQTT ha Standardizzato un AFFIDAB. ma non la Sicurezza.

Io Sono AUTENTICATO al BROKER, e Anche il BROKER Verso di noi ma NON però END TO END, Quindi non so Chi Sarà Che legge i Miei PUBLISH.

KAFKA ad Esempio può Impostare DIRITTI Su Alcuni TOPIC.

**CONNESSIONI BREVI per PUBLISHER e LUNGHE Per SUBSCRIBER**

Se SUBSCRIBER è dietro al NAT Nessun Problema Perché è lui che si Attacca al BROKER. ⇒ NAT COMPLIANT

## STRUTTURA DEL TOPIC:

Sono Stringhe, Meglio Non Usare gli Spazi

Possò Sfruttare TOPIC Gerarchici Implementati Tramite '/' che mi Permette di Costruire degli Alberi di TOPIC

Nei TOPIC Possò Usare dei caratteri '#' o '+' per:

/// '#' Fare SUBSCRIBE Di tutti i Sotto-TOPIC (Messo Sulle foglie) [come \* sul PATH di Linux] lvl1/lvl2/#

/// '+' Va Messo Sulle RADICI +/lvl1/lvl2

ESEMPIO ESAME Mostrare la differenza Con Chat Web Socket

PUB-SUB



Devo Implementarmi Tutto



Devo fargli Scaricare App

Ho Browser



GUI è facilitata



Non devo far Scaricare nessuna App



**SICUREZZA:** ⇒ Ricordarsi i 5 Punt:

**CRITTOGRAFIA:** Implementazione

Vuole Rendere SICURA L'INFORMAZIONE in modo che un UTENTE NON autorizzato NON possa **DECIFRARE** L'INFORMAZIONE e Comprenderla

↳ Le mie **Stringhe di BYTE**

**CRITTOANALISI** Invece è il Contrario, Quindi Cercare di Aggirare o Superare le PROTEZIONI CRITTOGRAFICHE accedendo alle INFO PROTETTE

Ci Sono **2 Aspetti Ben SEPARATI:**

❗ **ALGORITMO** → Brutta Pratica, NON Posso Sapere se c'è un BUCO Non Me Ne Accorgo, NO SEGRETEZZA ALGORITMO  
La forza dell'Algoritmo deve Stare Nella Segretezza della CHIAVE

❗ **CHIAVE** → Deve Essere Tenuta Segreta

Quando BYTE Sono cifrati Prende il Nome di CIPHER TEXT  
anche se NON è per FORZA Testo

C'è un ALGORITMO che decifra (S2. Inversa della Cifatura) anche  
detto INVERSO, Parametrizzata Sulle CHIAVE

↳ Che Specializza l'Algoritmo  
e NON è nota la chiave

### ESEMPI STUPIDI:

❧ Aggiungi  $x$  ad Ogni BYTE  $\Rightarrow$  Chiave è  $x$  e NON So di  
Quanto INCREMENTO

❧ Sposto di  $x$  Lettere  $\Rightarrow$  Non So di Quanto Sposto

**CIFRATURA SIMMETRICA:** Chiavi Uguali per CIFRATURA e  
DECIFRATURA (come la PORTA di Casa) [esempio della Somma]

**CIFRATURA ASIMMETRICA:** Una è PUBBLICA e l'Altra è  
privata (Nascono Assieme e Sono legati da una S2. Matematica)

Sono INTERSCAMBIABILI, NON hanno Ruolo Specifico ma So  
Solo che se una viene usata Per CIFRATURA NON va BENE  
per DECIFRARE e Viceversa.

CHIAVE PUBBLICA la do a TUTTI mentre Quella PRIVATA la Tengo  
io Ma Salgo io a Caso Quale Voglio CONDIVIDERE Perché

a lui di Caratteristiche Sono Uguali (DIPENDE L'USO CHE NE FACCIO)

Non devo Scambiarle Poi

Devo RENDERE il più difficile Possibile, ma Si può Sempre fare il BRUTEFORCING:

/// ho 8 Bit? faccio  $2^8$  Prove e Risolvo.

SICUREZZA PRATICA  $\Rightarrow$  Sforzo per fare la VIOLAZIONE deve ESSERE disEconomico, Quindi deve Essere Svantaggioso.

Quasi IMPOSSIBILE Se ho il concetto di "SESSIONE" e Quindi ho un PROTOCOLLO che RINNOVA la chiave con Tempo  $<$  Che ci mette la Tecnologia a Fare forza Bruta.

Protocollo Usa la Chiave, ma ciò che è IMPOSSIBILE Oggi, DOMANI Sarà Possibile ed ESSENDO BRUTE-FORCING e Quindi:

/// Si inventa NUOVO Protocollo

/// Oppure si Allunga DIMENSIONE chiave

SIMMETRICA ha cifratura più leggera per chi ha la Chiave e la Posso Pagare Con lo Scaricarsi della BATERIA o Energia.

Ma il Problema è COME MI SCAMBIO la chiave in Modo Sicuro Senza Però Usare la chiave

$\rightarrow$  Usa la CIFRATURA ASIMMETRICA (che non Necessita Sambio di Chiave)

Per Scambiarsi La Chiave SIMMETRICA Per  
Poi Abbandonare L'ASIMMETRICA.

---

### CIFRARIO DI CESARE:

Chiave è #Shift, Si Applica a TUTTO, Non Solo al TESTO ma anche a JPEG Ad Esempio.

L'Alfabeto per la Macchina è 255 Visto che Viene Considerato il SINGOLO BYTE

1 Byte = 8 Bit

$2^8 - 1$  Possibili Chiavi

ALFABETO In INFORMATICA Rappresenta tutti i Simboli che Riesco a Creare Partendo Dall'UNITÀ BASE

Per BRUTEFORCING devo Provare TUTTE LE POSSIBILI Permutazioni di una STRINGA  $\Rightarrow$  NEL NOSTRO CASO  $\Rightarrow 255!$

UPPER BOUND è il BRUTEFORCING

Cifratura monoALFABETICA si Presta ad un Attacco sulle frequenze della COMPARSA dei Simboli.

Questo Capita anche Nel CODIFICARE un PDF oppure Anche Nelle IMG .Jpg.

## CIFRATURA A BLOCCHI:

Quello che Viene Usato Oggi

Dati  $x$  Bit i Possibili  $2^k$  ingressi Vengono Permutati

## CIFRATURA ASIMMETRICA:

Sono 2 Chiavi FORMALMENTE UGUALI

Quello che cifro con una Chiave Posso decifrarla con l'Altra e Viceversa.

Conosco una Chiave, ma Quella NON dà Nessuna INFO Per la Privata

Non c'è Entità Terza per lo SCAMBIO della Chiave

Ai fini della CONFIDENZIALITÀ Basta che si SCAMBINO le Chiavi

Ma Sono Sicuro che Sto Parlando con la PERSONA giusta?

Ruolo della BLOCKCHAIN Cruciale per la FIDUCIA Rispetto a fidarsi Nella Catena