Contact Tracing su Blockchain

rev 0.1

Alessio Ferone, Antonio Della Porta, Luigi Borchia,
Giovanni Volpicelli
parthenopeblockchaingroup@gmail.com

24 aprile 2020

Per contrastrare l'epidemia da SARS-CoV-2 c'è bisogno di velocizzare e semplificare il processo di notifica a persone potenzialmente a rischio contagio. A questo fine sempre più sistemi di **contact tracing** vengono adottati e richiesti dalle nazioni che combattono il contagio. L'idea di base del contact tracing digitale è di tracciare il contatto tra individui attraverso l'utilizzo della tecnologia BLE (Bluetooth Low Energy) per stimare la vicinanza fisica. I dati così raccolti, verranno processati in modo decentralizzato e utilizzati al solo scopo di notificare le persone che sono state a contatto con il virus. Il contact tracing non ha nei suoi obiettivi la rivelazione dell'identità dell'infetto ed il suo tracciamento.

Molti dei modelli presentati fin ora (DP-3T, PEPP-PT, il sistema Apple/Google) utilizzano come intermediario tra i vari peer del sistema un sistema backend centralizzato. In questo documento viene presentato un modello che utilizza come tecnologia di backend un sistema blockchain sfruttando il concetto di **smart contracts**. Crediamo che un tale sistema possa aprire la strada verso una unificazione della frammentazione che si verrà a creare nel caso di nascita di molteplici applicativi per il contact tracing.

Ogni contributo al documento o al progetto è ben accetto.

Sommario

In questo documento verrà presentato un modello di contact tracing che ha le seguenti caratteristiche:

- **Tutela della privacy** dell'individuo contagiato e di chi è stato a contatto con lui
- **Trasparenza del backend** e possibilità di auditing del software che si occupa della notifica di possibili contagiati.

Unico obiettivo del sistema è la notifica ad individui che si siano trovati in contatto con persone che sono state provate positive da parte del **sistema sanitario**.

Tra gli obiettivi del sistema **non** ci sono invece:

- Il tracciamento del contagiato
- La ricostruzione delle sue interazioni sociali

L'adesione al sistema qui proposto è da intendersi a titolo volontario da parte dei cittadini sani o infetti che siano.

Panoramica del sistema

Il sistema proposto è un sistema che rivela le minime informazioni necessarie al pubblico. Tutte le informazioni vengono memorizzate su piattaforme decentralizzate, in cui sono meno probabili attacchi che si possono verificare con un sistema di notifica e memorizzazione centralizzato. Il protocollo si divide in 3 fasi:

(1) **Broadcast e ricezione di ID temporanei**: gli utenti generano ID temporanei che vengono comunicati ai device circostanti grazie al BLE. Parallelamente gli utenti restano in ascolto di ID temporanei nelle vicinanze, che memorizzano su un database locale per la successiva verifica di contatti potenzialmente contagiosi.

(2) **Notifica di contagio**: un utente a cui viene diagnosticato il virus, riceverà un codice di conferma grazie al quale potrà pubblicare su blockchain una **forma compatta** che rappresenta il set degli ID temporanei utilizzati durante il periodo in cui era contagioso.

(3) **Ricezione di eventi dalla blockchain**: gli utenti resteranno in ascolto di eventi innescati dalla blockchain, i quali indicheranno la presenza di nuovi contagi. A quel punto, sarà necessario scaricare la forma compatta messa a disposizione dall'utente contagiato e controllare nel proprio database per trovare eventuali corrispondenze tra i propri ID e quelli segnalati dall'utente contagiato.

Il sistema può essere diviso in due parti:

- **Tracciamento dei contatti (off-chain)**, che si occupa del tracciamento dei contatti degli utenti. Questa parte del sistema consiste in un'applicazione per smartphone che sfrutta la tecnologia BLE (Bluetooth Low Energy) e alcune sue caratteristiche per il broadcast e la ricezione di identificativi anonimi.

- **Smart contract per la notifica di contagio (on-chain)**, che si occupa della ricezione dei dati degli infetti e della notifica dei dati agli utenti del sistema. Questa parte del sistema consiste in *smart contracts* rilasciati su una sistemi blockchain basati su EVM (Ethereum, Hyperledger Burrow, Quorum).

Di seguito verranno descritte le tecnologie utilizzate dal sistema ed i loro dettagli.

ID temporanei e BLE (Bluetooth Low Energy)

Sono state recentemente pubblicate le specifiche del protocollo presentato da Apple e Google. Il framework messo a disposizione non permette di cambiare le dinamiche del gestore delle chiavi e crediamo perciò che esso diventerà presto lo standard per il contact tracing su dispositivi mobili.
Il protocollo introduce il concetto di ID pseudocasuali chiamati *Rolling Proximity Identifiers* [2]. Il device che esegue il protocollo avvierà un broadcast degli RPI precedentemente citati utilizzando la tecnologia BLE (Bluetooth Low Energy) [1] e contemporaneamente resterà in ascolto per ricevere gli RPI dei device nelle vicinanze, creando uno storico dei contatti avuti. I RPI vengono distribuiti come un payload di 20 byte in cui i primi 16 byte rappresentano l'ID [1].

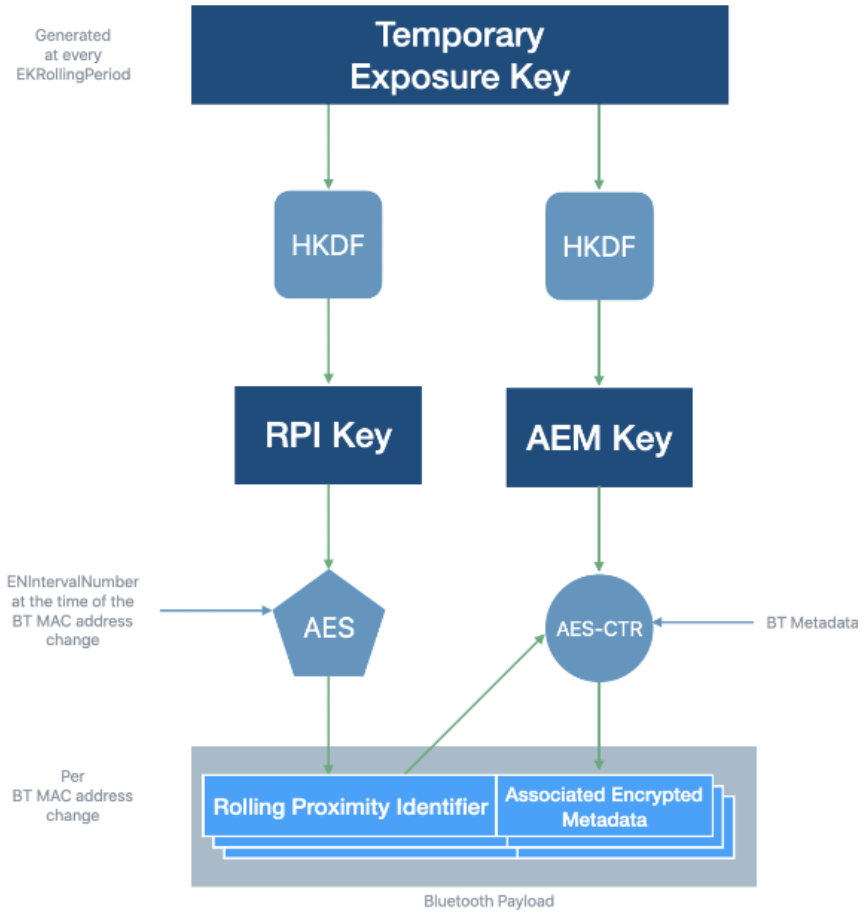

Figura 1: Gestione delle chiavi nel protocollo proposto da Apple e Google [2]

Cuckoo Filters

Un *cuckoo filter* è una struttura dati probabilistica che consente di verificare l'appartenenza di un elemento ad un set. È possibile avere dei falsi positivi, ma non dei falsi negativi. Esso è una hash table in cui le collisioni sono risolte tramite un meccanismo chiamato *partial-key cuckoo hashing* basato sul metodo *cuckoo hashing*. È possibile mantenere un tasso di falsi positivi

stabile al 0.03% fissando un numero di bit per entrata nella tabella di 7 bit e un carico della tabella del 95.5%, mentre è possibile mantenere un tasso $\approx 0.001\%$ utilizzando 16 bit per entrata nella tabella [4] . Grazie ai *cuckoo filters* è possibile avere una rappresentazione compatta del set di RPI generati da un contagiato.

Smart Contracts

Gli *smart contracts* consentono la definizione di transazioni condizionali e la memorizzazione di dati strutturati su blockchain. Essi vengono eseguiti in modo distribuito dalla rete offerta dal sistema blockchain sul quale vengono definiti. Inoltre sono ispezionabili pubblicamente nel codice e nei dati in essi memorizzati.
Gli *smart contracts* ci offrono quindi uno strumento per costruire un servizio di backend **pubblico** e **trasparente**. Nel modello proposto le forme compatte delle chiavi notificate da un infetto vengono memorizzate da uno smart contract che avrà il compito di emettere eventi in cui vengono specificati i dettagli per i nuovi contagi.

Dimensione dei *cuckoo filters* e *gasLimit*

Un limite che presenta l'utilizzo di *smart contract* come servizio di backend è la dimensione massima dei dati trasferibili tramite una transazione, che viene dettata dal *gasLimit*. I costi per il trasferimento di dati tramite transazioni sono così definiti [9]:

- 68 *gas* per ogni byte diverso da 0
- 4 *gas* per ogni byte uguale a 0

Al momento della scrittura di questo documento, il *gasLimit* per la rete pubblica di Ethereum è 10 milioni, che consentirebbe quindi il trasferimento di 146 kB. L'utilizzo di una struttura dati come i *cuckoo filters* permette di rappresentare molti RPI efficientemente. Sarebbe infatti possibile rappresentare ≈ 20000 RPI utilizzando 65 kB.

Dettagli del protocollo

Tracciamento dei contatti Il protocollo di tracciamento dei contatti avviene in modo distribuito attraverso una applicazione che utilizzerà il framework messo a disposizione da Apple e Google. Come descritto in precedenza i RPI generati dall'utente verranno emessi in broadcast tramite BLE (Bluetooth Low Energy), impostando come UUID del dispositivo il RPI corrispondente. L'UUID verrà riscritto allo scadere dell'RPI temporaneo, quando verrà emesso il RPI temporaneo successivo.
Allo stesso tempo gli utenti resteranno in ascolto di dispositivi nelle vicinanze dai quali cattureranno gli RPI comunicati. Questi ultimi verranno memorizzati in un database locale in cui resteranno per 14 giorni.

Notifica di contagio Nel momento in cui un utente si sottopone al test epidemiologico gli verrà rilasciato un codice r composto da 32 byte randomici. L'hash $KEC256(r)$ verrà poi segnalato allo *smart contract* che lo terrà inattivo fino alla richiesta di attivazione dello stesso.
Nel caso in cui il test risultasse positivo, l'utente verrà notificato dall'ente predisposto che avrà attivato la possibilità di utilizzare il codice precedentemente concordato. Qualora l'utente decidesse di notificare alla rete la positività, l'applicazione costruirà un *cuckoo filter* inserendo ognuno dei RPI da lui generati nel periodo infettivo. Il filtro così costruito verrà serializzato e memorizzato dallo *smart contract* utilizzando come codice di conferma il codice r. L'applicazione dovrà quindi eseguire una transazione che invoca lo *smart contract* con gli argomenti `(filter, r)`.

Ricezione di eventi generati dalla blockchain All'arrivo di una nuova segnalazione di positività, lo *smart contract* emetterà un evento serializzato come $(timestamp, filter)$, che verrà scritto all'interno del blocco in cui la transazione verrà inclusa. L'evento conterrà il timestamp del momento in cui è avvenuta la segnalazione e la forma serializzata del *cuckoo filter*.
I dispositivi dovranno interrogare la blockchain per ricevere queste informazioni tramite gli eventi e successivamente ricostruire i relativi *cuckoo filters* dalla loro rappresentazione serializzata, grazie ai quali potranno verificare la presenza nel database locale di eventuali incontri a rischio. In caso di riscontro positivo, l'utente riceverà una notifica.

Considerazioni sull'infrastruttura

Il modello definito in questo documento si basa sull'utilizzo di un sistema blockchain. Quest'ultimo potrebbe essere rappresentato dalla rete principale Ethereum o da una sua instanza consorziata in ambito privato. Svariati progetti basati su Ethereum permettono l'implementazione di un sistema blockchain permissioned (Quorum, Hyperledger Burrow, Besu) [8, 6, 3].
L'utilizzo di *smart contracts* per la raccolta e la notifica di nuovi contagi pone la necessità di distribuire moneta nativa del sistema blockchain per l'esecuzione di transazioni verso gli smart contract del sistema, rendendo impraticabile l'adesione al sistema senza i fondi necessari. È possibile eludere questo limite attraverso l'utilizzo di **GSN** (Gas Station Network) [5]. L'utilizzo di questo sistema permette di delegare il pagamento delle transazioni ad una terza parte (chiamata *Relayer*) con una soluzione decentralizzata che permette ai partecipanti al protocollo di controllare la sottomissione di transazioni alla rete.

Analisi del protocollo

Sicurezza

Di seguito verranno esaminate e discusse le fasi che compongono il protocollo:

(1) Durante la fase di **tracciamento dei contatti**, le vulnerabilità che possono presentarsi riguardano tutte la possibilità di raccolta di RPI temporanei da parte di un attore malevolo tramite strumenti adatti ad effettuare uno scan dell'area in cui si trova.

(2) Durante la fase di **notifica del contagio**, un attore malevolo potrebbe entrare in possesso del codice r e notificare informazioni errate o ingannevoli. Attacchi a forza bruta contro il codice r sono limitati dalle proprietà dell'algoritmo di hashing scelto.

(3) Durante la fase di **ricezione di eventi generati dalla blockchain** l'utente potrebbe scaricare dati ingannevoli dovuti ad un codice r finito nelle mani di un attore malevolo. È possibile mitigare l'attacco legando il codice r ad un indirizzo Ethereum definito ed appartenente all'utente.

(4) Un attore malevolo potrebbe raccogliere informazioni riguardo le interazioni tra un utente e lo smart contract (o il *Relayer*) attraverso tecniche di analisi del traffico. È possibile tuttavia mitigare questo attacco tramite l'utilizzo di proxy e *dummy packets* per anonimizzare le richieste [7] .

Utilizzo dei *Rolling Proximity Identifiers* Nel modello presentato vengono utilizzati i RPI piuttosto che le *Temporary Exposure Keys* introdotte dal modello Apple/Google [2] per non esporre l'utente a possibili analisi basate sui RPI da lui generati. Tuttavia un ulteriore approfondimento su questo tema è ancora in via di discussione.

Scalabilità

La natura decentralizzata del meccanismo di tracciamento dei contatti tramite BLE rende lo stesso scalabile nel complesso poiché permette di non avere bottlenecks.
Tuttavia ci sono considerazioni da fare per ogni singolo individuo che utilizza l'applicazione in termini di spazio occupato in memoria e utilizzo della rete:

- Durante il tracciamento dei contatti, ogni record registrato occupa 16 bytes.
- Per la segnalazione o la ricezione di un nuovo contagio i dati utilizzati dipendono dalla dimensione dei *cuckoo filters* generati. Tuttavia i *cuckoo filters* consentono di rappresentare i RPI generati dall'utente mantenendo una dimensione contenuta.

Riferimenti bibliografici

[1] Google Apple. *Exposure Notification - Bluetooth Specification*. 2020. URL: https://covid19-static.cdn-apple.com/applications/covid19/current/static/contact-tracing/pdf/ExposureNotification-BluetoothSpecificationv1.1.pdf.

[2] Google Apple. *Exposure Notification - Cryptography Specification*. 2020. URL: https://covid19-static.cdn-apple.com/applications/covid19/current/static/contact-tracing/pdf/ExposureNotification-CryptographySpecificationv1.1.pdf.

[3] *Besu*. URL: https://www.hyperledger.org/projects/besu.

[4] Bin Fan et al. "Cuckoo Filter: Practically Better Than Bloom". In: *Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies*. CoNEXT '14. Sydney, Australia: Association for Computing Machinery, 2014, pp. 75–88. ISBN: 9781450332798. DOI: 10.1145/2674005.2674994. URL: https://doi.org/10.1145/2674005.2674994.

[5] *GSN - Gas Station Network*. URL: https://www.opengsn.org/.

[6] *Hyperledger Burrow*. URL: https://www.hyperledger.org/projects/hyperledger-burrow.

[7] The DP-3T Project. *Privacy and Security Evaluation of Digital Proximity Tracing Systems*. URL: https://github.com/DP-3T/documents/blob/master/Security%20analysis/Privacy%20and%20Security%20Attacks%20on%20Digital%20Proximity%20Tracing%20Systems.pdf.

[8] *Quorum*. URL: https://www.goquorum.com/.

[9] Dr. Gavin Wood. "Ethereum: A secure decentralized generalised transaction ledger". In: (2019). URL: https://ethereum.github.io/yellowpaper/paper.pdf.