Distributed Systems 2: Corona Contact Tracing

Wannes Vermeiren, Wout Deleu, Toon Eeraerts

December 2022

1 Implementatiekeuzes

1.1 Tokens en timestamps

Bezoeken aan een facility worden opgeslagen in het Visit object. Dit wordt zowel gebruikt door de gebruiker om lokaal al zijn bezoeken op te slaan als om zijn bezochte plaatsen door te geven aan de Mixing Proxy en eventueel aan de dokter. Een afwijking van de paper is dat er per bezoek, per gebruiker slechts één token wordt opgeslagen. Bij die ene token kunnen zoveel timestamps als we willen horen. Als er slechts één timestamp wordt opgegeven en de gebruiker dus vergeet aan te duiden dat hij de bar verlaten heeft, gaan we uit van een standaardduur van 5 uur.

1.2 Mixingproxy

De Mixing Proxy wordt hoofdzakelijk gebruikt als tussenstap in de communicatie en dus al cache. De `flushCache()` methode kan hierbij worden uitgevoerd om de Mixing Proxy volledig te ledigen, en alles permanent op te slaan in de Matching Service, waar alle logica wordt uitgevoerd. Bij het binnenkomen van logs via de dokter, zal de Matching Service automatisch aan de Mixing Proxy vragen om al zijn data te flushen.

1.3 Matching Service

In de Matching Service worden alle capsules omgezet in Entry-objecten, verwijzend naar een data entry in een databank. Alle logs die binnenkomen worden geverifieerd door de signature van de dokter te checken en te kijken of de R_i waarden echt zijn. Dit kan door de nyms op te vragen aan de Registrar en zo zelf een hash te maken. Deze hash moet dan overeenkomen met de hash in de log afkomstig van de dokter en origineel van de gebruiker. Als dit niet het geval is worden de logs buiten beschouwing gelaten bij het aanduiden van al dan niet besmette entries in de databank.

2 SWOT analyse

2.1 Sterke punten

- Privacy is de essentie van het ontwerp en wordt ook in onze implementatie gerespecteerd door het volgen van de structuur die aangegeven wordt in de paper. Alle

gegevens van de gebruikers worden niet vrijgegeven buiten het systeem. Enkel wanneer een gebruiker geïnfecteerd geraakt zullen de gegevens noodzakelijkerwijs door het systeem worden vrijgegeven.

- Door het handtekenen van heel wat zaken, zal het insturen van gewijzigde data snel gedetecteerd worden. Andere data om in te dienen zit dan weer meestal verborgen in hashstructuren en is dus moeilijk te achterhalen.

- Zeker niet alle code in dit project is even overzichtelijk, maar onze klasse Methods draagt hier wel sterk toe bij. Hierin staan voor drie categoriën de meest voorkomende methodes geïmplementeerd, zodat we deze overal in het project op dezelfde manier kunnen gebruiken. Enkele voorbeelden van de cryptografische functies zijn `hash()`, `getKeyPair()`, `KDF()` en `checkSignature()`. Ook bevat de klasse allerlei conversies van `byte[]` naar Strings e.d. Tot slot werd ook het connecteren naar elk van onze drie servers in een aparte methode gegoten.

2.2 Zwaktes

- De robuustheid van het systeem kan nog veel verbeterd worden. Uiteraard bevat onze code hier en daar nog wat kleine foutjes die bijgeschaafd kunnen worden. Verder zullen bij het uitvallen van één van de servers, de andere servers ook stoppen met werken, door de RemoteException afkomstig van de RMI-connectie.

- Internet connectie is noodzakelijk, doordat er verbinding met de servers gelegd moet worden. Bars die toevallig in een gebied liggen zonder bereik zullen geen gebruik kunnen maken van het systeem. Onze implementatie houdt momenteel geen rekening met het cachen van gegevens in afwachting tot een internetconnectie.

2.3 Kansen

- Dit project bevat veel technologieën waarmee we voor het eerst konden experimenteren. De RMI-technologie die ons toeliet om de verbinding te leggen tussen servers en clients. En ook de crypto protocollen die de communicatie versleutelen.

- Dit project liet toe om de onderliggende theorie en technologiën goed onder de knie te krijgen. Het kan een goede basis vormen om van te leren en van over te stappen op een uitvoering die geschikt is voor productie. De grootste extra stappen hierbij zouden zijn om de GUI naar een veel hoger niveau te tillen, gebruik te maken van echte, permanente serveropslag en het encrypteren van de communicatiekanalen.

2.4 Bedreigingen

- Het overzicht behouden van alle code en toevoegingen van ieder lid overzichtelijk houden was moeilijk. Vaak was er onduidelijkheid over de manier waarop zaken werden geïmplementeerd. Een duidelijke planning en een toevoeging van algemene methoden was hier de oplossing. De code blijft echter relatief chaotisch, waardoor aanpassingen/uitbreidingen niet zo simpel zijn.

3 Code

De code van het project kan gevonden worden op Github via: `https://github.com/WannesVermeire/Corona_ContactTracing.git`