Tracing von großen Personengruppen unter Beachtung der Privatsphäre und des Datenschutzes im Kontext der Coronakriese

Leander Ludwig Günther Dreier
Professionelle Textsatzsysteme
Technische Hochschule Ingolstadt
Ingolstadt, Deutschland

Erstprüfer: Dr.-Ing. Paul Spannaus
Zweitprüfer: M. Sc. Christoph Nebl
Abgabedatum: 6. Juli 2020

***Abstract*—Zur Einschränkung der Verbreitung des Coronavirus SARS-CoV-2 in der Corona-Pandemie 2020 werden unter anderem sogenannte Corona-Tracing-Apps eingesetzt. Diese Anwendungen sollen Infektionsketten nachverfolgen und schließlich brechen, indem sie gefährdete Nutzer informieren. Für die Akzeptanz einer solchen Anwendung unter den Bürgern ist es wichtig, dass Datenschutz- und Anonymitätsstandards eingehalten werden. Dieses Paper stellt drei unterschiedliche Tracing-Apps für den Gebrauch in Deutschland vor und geht auf die verschiedenen technischen Umsetzungen ein. Zudem wird die Erfüllung oben genannter Ziele untersucht und die Entscheidungen der Bundesregierung bezüglich der Einführung einer solchen Anwendung bewertet.**

***Index Terms*—tracing, privacy, anonymity**

1. Einleitung: Skepsis der Bevölkerung bezüglich neuer Corona-Tracing-Apps

Im Rahmen der aktuellen Corona-Pandemie wird auch im Bereich der Informatik geforscht, wie die aktuelle Situation verbessert, beziehungsweise die Ausbreitung der Krankheit eingedämmt werden kann. So wurde in Deutschland zum Beispiel vom 20. bis zum 22. März 2020 ein Hackathon mit dem Motto „WirVsVirus“ abgehalten [1]. Im Fokus von Medien und Politik sind hierbei aktuell besonders sogenannte *Corona-Tracing-Apps*. Diese sollen eine Verbreitung des Virus reduzieren, idem sie Nutzer der App warnen sollen, wenn diese über einen bestimmten Zeitraum mit einem nachträglich als SARS-CoV-2-positiv getesteten Menschen in Kontakt waren. Hierzu müssen Standortdaten der Nutzer erfasst werden, wodurch eine ungewollte Überwachung durch Bewegungsprofile denkbar ist. Hinzu kommt, dass bei geringen Installationszahlen kaum mit einem positiven Effekt zu rechnen ist. Es scheint daher notwendig, Klarheit über die Implementation einer Lösung zu schaffen, bevor diese von einer ausreichenden Nutzerzahl verwendet wird.

In dieser Arbeit werden daher verschiedene Ansätze zur Realisierung einer solchen Anwendung betrachtet. Ein Fokus wird hierbei auf Lösungen gelegt, die für den Gebrauch in Europa beziehungsweise Deutschland entwickelt wurden. Es sollen nicht nur technische Unterschiede aufgezeigt werden, sondern auch die unterschiedlichen Beziehungen zu ethisch-moralischen Vorstellungen. Hierdurch soll die Transparenz der Angebote in diesem möglichen Konflikt zwischen Gesundheit und Privatsphäre erhöht werden.

In Abschnitt 2 wird auf den aktuellen Stand verschiedener Anwendungen eingegangen, welche in Abschnitt 3 evaluiert werden. Darauf folgt in Abschnitt 4 eine Beurteilung der aktuellen Situation, welche in einen Ausblick in Abschnitt 5 übergeht. Abgeschlossen wird mit einem Fazit in Abschnitt 6.

2. Stand der Technik: Untersuchte Apps

In diesem Text sollen drei Apps miteinander verglichen werden. Zuerst wird die Lösung der Initiative *Pan-European Privacy-Preserving Proximity Tracing (PEPP-PT)* [2] und hierbei speziell die deutsche Variante betrachtet. Es folgt ein Überblick über das Protokoll *Decentralised Privacy-Preserving Proximity Tracing (DP3T)* [3], sowie über die sogenannte *Corona-Warn-App (CWA)*, entwickelt von SAP und der Deutschen Telekom AG [4].

2.1. PEPP-PT

Die deutsche Lösung von PEPP-PT baut auf einer Server-Client-Architektur aus drei Komponenten auf: einer Anwendung zur Installation auf den Smartphones der Nutzer, einem zentralen Backend, sowie einem Service für Push-Benachrichtigungen (PBS). Die App broadcastet unverschlüsselte, kurzlebige Pseudonyme via Bluetooth. Gleichzeitig werden solche Pseudonyme von anderen Instanzen der Anwendung empfangen und gespeichert. Diese unter den Clients gesendeten Tokens werden *EBIDs* genannt und vom Backend an die Nutzer verteilt. Somit kann im Backend eine Zuordnung von EBID zu Anwendungsinstanz hergestellt werden. Diese Zuordnung wird genutzt, um im Falle einer Infektionsmeldung durch einen Client zentral berechnen zu können, welche anderen Nutzer sich dadurch wahrscheinlich angesteckt haben und darüber informiert werden müssen. Solche Benachrichtigungen erfolgen nicht direkt über das Backend, sondern über den PBS. Kommunikation zwischen Nutzer und Backend beziehungsweise PBS erfolgt verschlüsselt via TLS 1.2. Anstalten des Gesundheitswesens kommunizieren mit dem Backend über eine TLS-verschlüsselte Verbindung und mit den Nutzern über Wege außerhalb des Anwendungsscopes. Letzteres dient der Nutzerauthentifizierung, welche vor einem Upload von Daten nach einer Ansteckung notwendig ist [2, Abs. 5.1].

Die Nutzerregistrierung erfolgt bei PEPP-PT unter Einsatz eines Proof-of-Work-Verfahrens in Verbindung mit einem Captcha, um Spamaccounts und Denial-of-Service-Angriffe einzuschränken. Nach einer erfolgreichen Registrierung erhält das Backend ein eindeutiges 128-bit Pseudonym des Nutzers, die *PUID*. Dem Backend ist die Zuordnung der PUID eines Nutzers zu seinen verschiedenen EBIDs möglich, da die EBIDs durch Verschlüsselung der PUID mit zufällig generierten Schlüsseln *BK* erzeugt werden [2, Abs. 5.2.2]:

$$EBID_t(PUID) = \mathrm{AES}(BK_t, PUID) \tag{1}$$

Da die Schlüssel BK nur auf dem Server gespeichert werden, ist es Außenstehenden, welche EBIDs empfangen, nicht möglich, daraus die PUIDs herzuleiten. Die PUID ändert sich auf einem Smartphone nur, falls die Anwendung von dem Gerät gelöscht und neu installiert wird. Im Gegenzug erhält der Client bei Registrierung OAuth2-Credentials, welche er für zukünftige Kommunikation mit dem Backend nutzt. Wie die Registrierung im Detail abläuft, wird hier nicht näher erläutert [2, Abs. 5.2.1].

Wird ein Nutzer als SARS-CoV-2-positiv getestet, kann er empfangene EBIDs in Verbindung mit Zeitstempeln an das Backend schicken. Darauffolgende Berechnungen, welche anderen Nutzer ein erhöhtes Risiko haben, sich bei dem Patienten angesteckt zu haben, werden nun ausschließlich auf dem Backend ausgeführt. Eine grafische Darstellung des Grundprinzips dieser zentralen Risikobewertung ist in Abbildung 1a zu sehen. Am Start dieser Berechnungen steht das Entschlüsseln der EBIDs, um PUIDs daraus zu erhalten, welche dann bei Bedarf vom PBS benachrichtigt werden können. Diese Push-Benachrichtigungen beinhalten nicht die eigentliche Nachricht, sondern nur einen Hash davon oder eine zufällige Nummer. Außerdem werden nicht nur gefährdete Nutzer in dieser Form benachrichtigt, sondern auch eine große Zahl zufälliger Nutzer. Die benachrichtigten Anwendungen kontaktieren dann das Backend und fragen dieses, ob zu ihrer empfangenen Nummer eine Nachricht für sie existiert. Da so von dem PBS auch viele Noise-Nachrichten versandt werden, wird es Angreifern erschwert, auf Grund von Datenverkehr mit dem PBS auf Nutzer mit erhöhter Ansteckungsgefahr zu schließen [2, Abs. 5.2.3].

2.2. DP3T

Im Whitepaper zum DP3T werden zwei verschiedene Designvorschläge sowie zwei Ziele, verbunden mit unterschiedlich großer Nutzerbeteiligung, beschrieben. Bei den Designvorschlägen handelt es sich erstens um eine kostengünstigere Lösung zum dezentralen Contact-Tracing und zweitens um eine etwas komplexere Lösung, welche die Privatsphäre für ihre Nutzer verbessert. Die beiden Ziele sind einerseits die schnelle Benachrichtigung von Nutzern, welche sich möglicherweise mit SARS-CoV-2 angesteckt haben und andererseits die Zusammenarbeit mit Epidemiologen, um diese bei der Analyse des Ausbreitungsverhaltens des Virus mit Daten zu unterstützen [3].

Erster Designvorschlag. Hier versenden die Geräte kurzlebige IDs ($EphID$s) mit Hilfe von Bluetooth Low Energy Advertisements. Andere Smartphones empfangen diese Signale und speichern die IDs zusammen mit dem groben Zeitpunkt sowie dem Zeitrahmen in welchem sie es empfangen haben. Bei DP3T wird wie bei PEPP-PT zudem ein Backend-Server eingesetzt. Bei DP3T hängt jedoch die Sicherheit des Nutzers nicht von der Sicherheit des Servers ab. Dies bedeutet, dass auch im Falle einer Übernahme des Backends die Privatsphäre der Nutzer gewahrt bleibt. Sollte sich ein Nutzer infizieren, sendet dieser eine kompakte Repräsentation seiner EphIDs an das Backend, von welchem andere Nutzer diese dann abfragen können. Stellt ein Gerät fest, dass es Kontakt mit einer der empfangenen IDs hatte, wird auf dem Gerät das Risiko für den Benutzer ermittelt und dieser bei Überschreitung eines Schwellwerts darüber informiert.

Im Detail passiert dies wie folgt: Das Gerät des erkrankten Patienten sendet dem Server seinen Schlüssel SK_t, zusammen mit dem Tag t, der dem ersten Tag entspricht, an dem der Patient ansteckend war. Dieses Datenpaar wird dann an alle anderen Geräte verteilt und ermöglicht diesen eine Berechnung aller vom Patienten verwendeten EphIDs ab dem Tag t. Folgender mathematischer Zusammenhang liegt dem zugrunde: Ein Schlüssel SK_t wird, außer bei expliziter Änderung, durch

$$SK_t = \mathrm{H}(SK_{t-1}) \tag{2}$$

gebildet, wobei H für eine kryptographische Hashfunktion steht. Am Beginn eines Tages wird mithilfe dieses Schlüssels eine Reihe von EphIDs für diesen Tag

$$EphID_1 || \ldots || EphID_n = \mathrm{PRG}(\mathrm{PRF}(SK_t, \text{“bc key”})) \tag{3}$$

bestimmt. PRF steht dabei für eine pseudozufällige Funktion, wie HMAC-SHA256, bc key für einen fixen, öffentlichen String und PRG für eine Stromverschlüsselung wie AES im Zählermodus. Wie anhand dieser beiden Gleichungen zu sehen ist, lassen sich allein mithilfe des öffentlich gemachten (SK_t, t)-Wertepaares alle erforderlichen $EphID_i$-Werte durch Berechnung ermitteln. Die Speicherung der notwendigen Patientendaten auf dem Backend ist damit sehr speicherplatzeffizient [3, S. 10 f.].

Zusätzlich zu der bereits beschriebenen Funktionalität haben Nutzer weiter die Möglichkeit, Daten mit Epidemiologen zu teilen, falls sie mit einem erkrankten Nutzer in Kontakt kommen. Die Teilnahme hieran ist freiwillig und schränkt nicht die oben genannten Warnungen bei Kontakt mit Infizierten ein. Erklärt sich ein Nutzer zu diesem Programm bereit, werden einem ausgewählten Forschungszentrum regelmäßig Daten über Kontakte mit infizierten Nutzern geschickt. Diese Daten beinhalten die empfangenen Schlüssel SK_t der anderen Instanzen, ein boolesches Flag, welches angibt, ob der Nutzer selbst positiv getestet wurde, sowie Metadaten. Die Metadaten beinhalten, wann und wie oft die beiden Personen Kontakt hatten. Die Zeitinformationen werden dabei relativ zu t und nicht absolut übergeben. Auch Standortinformationen werden nicht geteilt [3, S. 13].

Zweiter Designvorschlag. Dieser Vorschlag legt einen höheren Wert auf die Privatsphäre, was auf der anderen Seite zu einem höheren Ressourcenverbrauch und höheren Kosten führt. Auch hier werden EphIDs von Geräten gene-

riert und an andere Geräte ausgesendet. Diese IDs bleiben in einer sogenannten Epoche i konstant und werden durch

$$EphID_i = \mathrm{TRUNCATE128}(\mathrm{H}(seed_i)) \tag{4}$$

gebildet. H bezeichnet hierbei eine kryptographische Hashfunktion, $seed_i$ ist ein 32-Byte-Zufallswert, der sich für jede Epoche ändert, TRUNCATE128 kürzt die Ausgabe auf 128 Bit.

Ein Gerät, welches eine EphID von einem anderen empfängt, speichert diese hier nicht wie im ersten Ansatz direkt ab, sondern bildet den Hashwert

$$\mathrm{H}(EphID||i). \tag{5}$$

Dieser Wert wird zusammen mit der Entfernung zum anderen Gerät, der Dauer des Kontaktes und einem groben Zeitpunkt des Kontakts gespeichert. Das Konkatenieren von i zu dem zu hashenden String unterbindet Replay-Angriffen, die außerhalb der Epoche ablaufen.

Die Repräsentation der EphIDs, die im Falle einer nachgewiesenen Infektion an das Backend übermittelt werden, wird durch die Menge

$$\{(i, seed_i)\} \tag{6}$$

gebildet. Anders als bei der kostengünstigeren Variante ist es der nutzenden Person möglich auszuwählen, von welchen Epochen i sie ihre Daten veröffentlichen möchte. Der Grund dafür ist, dass die verteilten IDs auf Basis verschiedener, unabhängiger Seeds gebildet werden.

Das Backend verwaltet einen Cuckoo-Filter F, dem es regelmäßig neue Einträge aus empfangenen Daten hinzufügt. Diese Einträge werden berechnet durch

$$\mathrm{H}(\mathrm{TRUNCATE128}(\mathrm{H}(seed_i))||i) \tag{7}$$

oder kurz

$$\mathrm{H}(EphID_i||i) \tag{8}$$

wobei $EphID_i$ wie in Gleichung 4 beschrieben bestimmt wird. Der Filter wird regelmäßig an die Smartphones verteilt, welche dann selbstständig überprüfen können, ob ihre Aufzeichnungen in F enthalten sind, um dann gegebenenfalls das Infektionsrisiko ihres Nutzers zu berechnen. Cuckoo-Filter können False Positives liefern, allerdings wurden die Filterparameter für DP3T laut Troncoso, Payer, Hubaux u. a. so eingestellt, dass dieser Fall bei einer Million Nutzer in einem Zeitraum von fünf Jahren nur einmal auftreten sollte. Die zusätzliche Anonymität der infizierten Nutzer wird dadurch erhöht, dass mit dieser Methode nicht mehr zuordnungsfähige IDs gespeichert und miteinander verglichen werden, sondern lediglich eine gehashte Repräsentation davon [3, S. 16].

2.3. Die Corona-Warn-App

Die Anwendung, die nach aktuellem Stand (26.06.2020) von der Bundesregierung Deutschlands unterstützt und empfohlen wird, entstammt einer Zusammenarbeit der Deutschen Telekom AG und der SAP SE und heißt schlicht Corona-Warn-App. Die Deutsche Telekom stellt hierfür die Infrastruktur zur Verfügung, während sich SAP auf die Anwendungs- und Backend-Entwicklung spezialisiert. Bereits in der Einleitung des README-Dokuments auf der offiziellen GitHub-Seite [4] wird auf die starke Anlehnung an das bereits beschriebene DP3T hingewiesen. Auch die Entwicklung rund um PEPP-PT wird als wertvoll, wenn auch weniger maßgeblich, für die CWA angesehen. Es handelt sich um eine Open-Source-Anwendung, welche unter Apache-2.0 lizenziert ist [4]. Die CWA arbeitet mithilfe eines neuen Frameworks namens *Exposure Notification Framework* (ENF), das von Apple und Google für deren Betriebssysteme iOS und Android gemeinsam bereitgestellt wurde. ENF arbeitet mit Bluetooth Low Energy. Wie bereits bei den anderen Ansätzen strahlen die Smartphones wechselnde Identifikationsnummern aus – hier *Rolling Proximity Identifier* (RPI) genannt – und empfangen diese gleichzeitig von anderen Geräten in der Nähe. Diese RPIs werden aus einem sich täglich ändernden *Temporary Exposure Key* (TEK) abgeleitet. Sobald ein TEK einem positiven Testergebnis zugeordnet wird, wird er *Diagnosis Key* genannt.

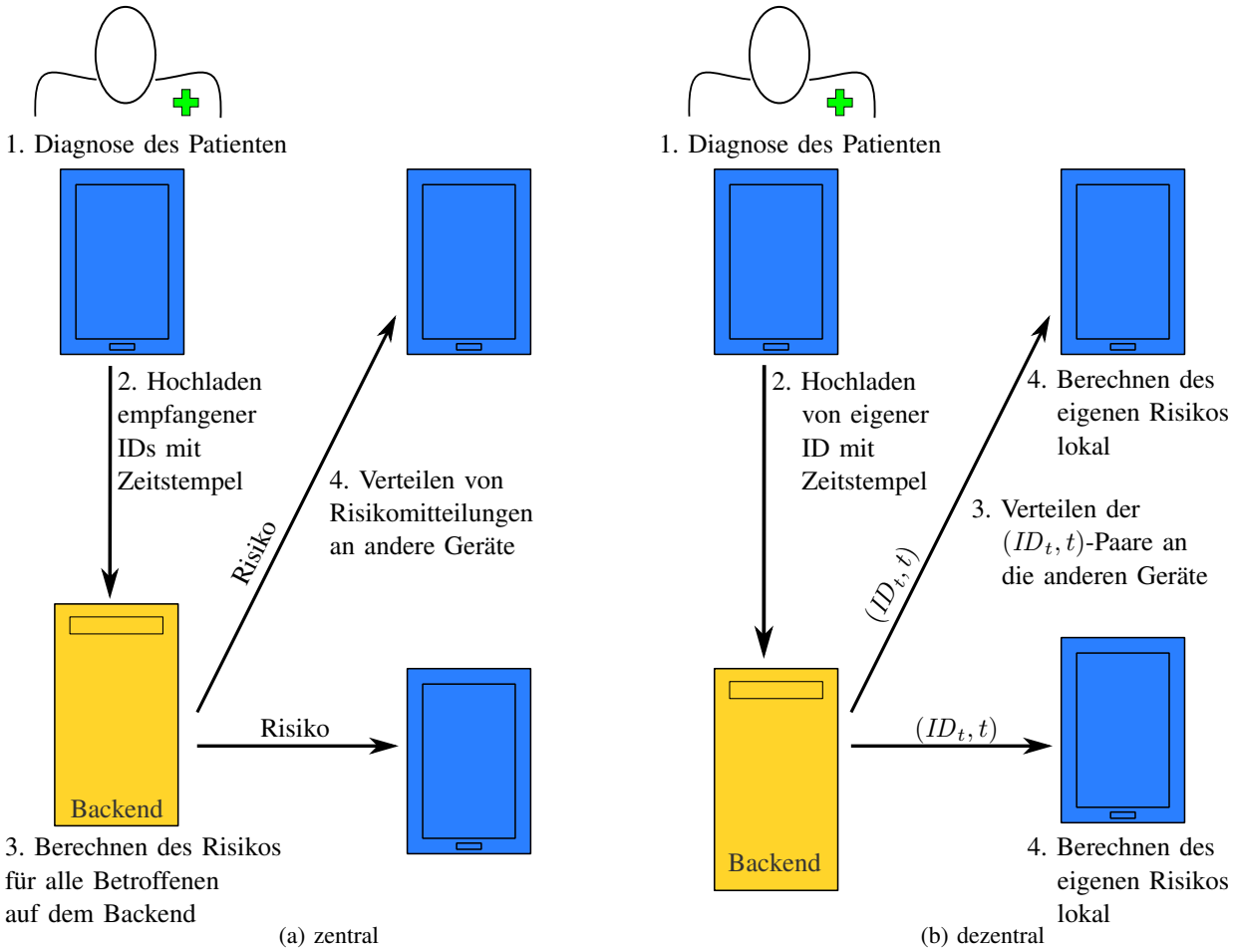
Aufbau des Backends.

- Der *Test Result Server* speichert Testergebnisse, welche global eindeutigen IDs (*GUIDs*) zugeordnet sind.
- Der *Corona-Warn-App-Server* speichert die Diagnosis Keys positiv getesteter Nutzer.
- Der *Verification Server* ist zuständig für einen Großteil der Kommunikation mit dem Nutzer und speziell für die Verifikation von Tokens.
- Der *Portal Server* wird zur Bestätigung von Testergebnissen verwendet, sollte sich der Nutzer gegen die Benutzung der elektronischen Testübermittlung durch CWA entscheiden.

Ablauf nach Test eines Nutzers auf SARS-CoV-2. Folgende Schritte laufen ab, wenn sich ein CWA-Nutzer testen lässt. Um die Übersichtlichkeit zu wahren wird dabei auf technische Details verzichtet.

1) Der Anwender erhält vom testenden Personal einen QR-Code, der eine GUID enthält, welchen er mit seiner Instanz der CWA einscannen kann. Tut er dies, so wird sein Gerät mit der GUID verknüpft und er erhält ein Registrierungstoken.
2) Die Proben werden zusammen mit einem Probenbegleitschein, dem QR-Code und verschiedenen Barcodes an das Labor übermittelt.
3) Sobald das Testergebnis bekannt ist, wird es zusammen mit einem Hash der GUID auf den Test Result Server hochgeladen und gespeichert.

4a) Hat der Nutzer in Schritt 1 seinen QR-Code gescannt, fragt sein Gerät den Verification Server regelmäßig nach Updates zu seinen Ergebnissen. Sind diese verfügbar, wird der Nutzer informiert. Sollte ein positives Testergebnis vorliegen, wird der Nutzer gebeten, seine TEK-Schlüssel freizugeben. Bei Zustimmung werden diese (jetzt Diagnosis Keys genannt) unter Verwendung einer TAN auf den CWA-Server hochgeladen und gespeichert. Die TAN wird dafür genutzt, die Gültigkeit der Daten zu überprüfen und unter Verwendung des Registrierungstokens angefragt.



(a) zentral (b) dezentral

Abbildung 1: Funktionsweise der Risikobewertung, angelehnt an [3, S. 12, Fig. PT]

4b) Hat sich der Nutzer jedoch am Anfang gegen die digitale Übertragung seiner Ergebnisse durch die Anwendung entschieden oder wird diese Möglichkeit vom Labor nicht unterstützt, wird ihm das Resultat direkt vom Gesundheitsamt mitgeteilt. Auf dem selben Weg kann ihm auch eine sogenannte *teleTAN* mitgeteilt werden. Diese teleTAN ist am Backend registriert, weshalb der Anwender damit ein Registrierungstoken von selbigem anfragen kann. Mit diesem Token kann dann wie oben eine TAN angefragt und mit dieser können die Diagnosis Keys hochgeladen werden.

In beiden Fällen müssen die Diagnosis Keys in zwei Schritten hochgeladen werden. Während die Schlüssel für die vergangenen 14 Tage (die genaue Anzahl der Tage kann variieren) sofort hochgeladen werden können, sollten solche für den Tag des positiven Tests und für darauffolgende Tage erst am Ende des jeweiligen Tages geteilt werden. Ansonsten könnte ein Angreifer aus einem aktuell gültigen TEK zusätzliche Rolling Proximity Identifiers (RPIs) erzeugen, welche mit einem positiven Testergebnis verbunden wären.

Auf den CWA-Server hochgeladene Diagnosis Keys können von anderen Instanzen der App regelmäßig heruntergeladen werden. Zusätzlich werden noch Konfigurationsparameter für das Framework vom Backend angefragt. Die Risikoberechnung erfolgt dann auf Basis dieser Daten mithilfe des Exposure Notification Frameworks von Apple und Google lokal auf den Geräten der Nutzer [5]. Wie auch schon bei der Inspirationsquelle DP3T wird hier also auf einen dezentralen Ansatz vertraut. Vereinfacht dargestellt ist dieses Prinzip in Abbildung 1b zu sehen, worin auch der Unterschied zur zentralen Herangehensweise gut erkennbar ist.

3. Beurteilung

Der Chaos Computer Club (CCC) veröffentlichte am 6. Apr. 2020 einen Post mit dem Titel *10 Prüfsteine für die Beurteilung von „Contact Tracing"-Apps*. In diesem wird das sogenannte Contact-Tracing, welches im Rahmen von Corona-Apps eingesetzt wird, als „Risikotechnologie" [6] beschrieben. Das angesprochene Risiko bezieht sich dabei auf möglichen Einschränkungen in der Privatsphäre der Nutzer. Es wird jedoch eingestanden, dass es durchaus möglich ist durch „„Privacy-by-Design'-Konzepte" [6] sichere und hilfreiche Anwendungen zu entwickeln. Um dies zu unterstützen, stellte der CCC zehn Kriterien auf, aufgeteilt in gesellschaftliche und technische Anforderungen. Anhand dieser Punkte sollen nun die oben vorgestellten Lösungsansätze betrachtet und bewertet werden.

3.1. Keine zentrale Entität, der vertraut werden muss

Die Privatsphäre des Nutzers sollte nicht von der Kooperation eines zentralen Servers abhängen. Das bedeutet, dass die Vertraulichkeit der persönlichen Daten auch im Falle eines übernommenen oder bösartigen Backend-Servers gewährleistet ist. Auch zufriedenstellend für diesen Punkt ist eine vollständig dezentrale Lösung ohne dedizierte Server.

Bei DP3T wird zwar ein zentraler Server eingesetzt, diesem werden allerdings per Design keine persönlichen Daten anvertraut [3, S. 9]. Da bei DP3T zwei mögliche Implementationsmöglichkeiten diskutiert wurden, wird im Folgenden bei Unterschieden zwischen den beiden getrennt auf sie eingegangen. Im ersten Designvorschlag hält das Backend dieselben Informationen, welche auch an alle Smartphones verteilt wird, nämlich eine Menge der Schlüssel-Zeit-Paare (SK_t, t). Dadurch, dass der Server keine zusätzlichen Informationen über die Nutzer besitzt, muss ihm in diesem Fall kein Vertrauen entgegengebracht werden. Angriffe, die zur Folge haben, dass der Server zusätzliche, nicht alle notwendigen, oder falsche Datenpaare verteilt, werden hier nicht näher behandelt, da sie die Privatsphäre der Nutzer nicht beeinträchtigen. Beim zweiten Designvorschlag unterscheiden sich die Datenpakete die das Backend erhält (Mengen von $\{(i, seed_i)\}$-Mengen) von denen, die es verteilt (einen Cuckoo-Filter F). Somit wäre es dem Backend möglich, die EphIDs der Nutzer zu berechnen, was den Endgeräten anhand des Filters jedoch nicht möglich ist. Durch dieses Ungleichgewicht entsteht jedoch keine Verschlechterung zum ersten Vorschlag, da es dort ohnehin jedem Gerät möglich ist, die EphIDs der infizierten Nutzer zu berechnen; es handelt sich im Gegenteil um eine Verbesserung. Darüber hinaus generiert die Anwendung auf den Smartphones diese nur für eine Epoche gültigen IDs selbstständig. Auch die Risikoberechnung wird durch die Nutzer selbst und nicht vom Backend durchgeführt. Dadurch muss der Server keine Informationen über die Kontaktbeziehungen zwischen Geräten speichern [3].

Ähnlich verhält es sich mit der CWA. Es existieren Backend-Server; diese speichern allerdings keine persönlichen Daten. Ein Angriff auf den Corona-Warn-App-Server könnte zum Beispiel nur Diagnosis Keys offenlegen, welche ohnehin publiziert und ohne Zuordnung zu einer Person oder einem Gerät gespeichert werden. Der Test Result Server speichert zwar Testergebnisse, diese werden jedoch nur dem Hashwert einer GUID zugeordnet, von welchem nicht ohne weiteres auf eine Person geschlossen werden kann. Es wäre allerdings denkbar, dass ein Angreifer sowohl Zugriff auf den QR-Code einer getesteten Person als auch auf die gespeicherten Daten dieses Servers hat. In diesem Fall könnte er selbst den Hash der GUID in dem QR-Code bilden und wüsste somit das Testergebnis der entsprechenden Person. Auch bei der CWA wird die Risikobewertung lokal auf den Geräten der Nutzer durchgeführt. Damit ist es einem bösartigen Backend nicht möglich, falsche Risikos an die Geräte zu übermitteln und ein solcher Datenaustausch kann nicht belauscht werden [5].

Im Gegensatz dazu steht der Ansatz von PEPP-PT, bei welchem die Risikos der einzelnen Nutzer auf einem zentralen Server berechnet werden. Damit verbunden ist, dass hier die Nutzer die zur Risikobewertung notwendigen Daten an das Backend schicken und nicht andersherum wie bei DP3T. Zwar gibt es auch hier IDs für die Anwender, welche sich regelmäßig ändern (die $EBID$s), allerdings werden diese von einer $PUID$ abgeleitet, welche im Backend gespeichert bleibt und sich zur Lebenszeit der Anwendung auf dem Smartphone nicht ändert. Die Nutzung dieser IDs kann zwar noch als Form der Pseudonymisierung angesehen werden – laut dem Informationsdokument von PEPP-PT ist auch einem bösartigen Backend-Admin eine Deanonymisierung der Pseudonyme nicht möglich [2, Abs. 2.2.2 A6] –, jedoch sollte diese Implementation eine solche Zurückverfolgung durch das Backend deutlich vereinfachen. Diese Meinung wird auch in einem offenen Brief von D64 – Zentrum für digitalen Forschritt e.V., LOAD e.V. – Verein für liberale Netzpolitik, Forum InformatikerInnen für Frieden und gesellschaftliche Verantwortung e.V. u. a. an die Bundesregierung vertreten [7].

3.2. Anonymität

Der CCC schreibt zu diesem Punkt: „[D]ie Nutzung des Systems [muss] möglich sein, ohne dass persönliche Daten jedweder Art erfasst werden oder abgeleitet werden können. Diese Anforderung verbietet eindeutige Nutzerkennungen.“ [6, Punkt 7] Weiter wird in diesem Punkt gefordert, dass verwendete IDs zum drahtlosen Tracing (in unseren Beispielen die $EBID$s beziehungsweise die $EphID$s oder die RPIs) nicht auf die Nutzer zurückführbar sein dürfen und sich häufig ändern müssen [6, Punkt 7]. Die Abgrenzung zum vorherigen Punkt ergibt sich aus dem Wegfallen eines zentralen Backends. Stattdessen wird hier allgemein die Möglichkeit einer Deanonymisierung untersagt, unabhängig von der handelnden Partei.

Auch in diesem Punkt schneidet die Implementierung von PEPP-PT aufgrund ihrer Verwendung eindeutiger PUIDs schlecht ab. Zwar ändern sich die via Bluetooth verschickten EBIDs regelmäßig (in dem Informationsschreiben dazu wird lediglich als Beispiel eine Gültigkeit von einer Stunde genannt [2, S. 14]). Alle diese Token werden jedoch durch Verschlüsselung einer PUID erzeugt, welche eindeutig einer spezifischen Installation der Anwendung zugeordnet wird und nur durch Neuinstallation jener geändert werden kann. Dadurch, dass das Backend die Schlüssel speichert, welche zur Generierung der IDs benutzt werden, ist diesem eine Zuordnung von EBID zu PUID jederzeit möglich. Die spezielle Implementierung von PEPP-PT setzt dieses Wissen sogar voraus, da es dem Server sonst nicht möglich wäre, die Risikowerte der einzelnen Nutzer zu berechnen. Es verstößt also per Definition gegen die Vorgaben der Anonymität.

Eine gewisse Verbesserung hierzu stellt die erste Variante von DP3T dar. Hier werden die über Bluetooth verteilten IDs *lokal* aus Schlüsseln SK_i abgeleitet. Dieser Schlüssel ändert sich zwar täglich, allerdings ist es möglich und gewollt, aus einem Schlüssel alle Nachfolgerschlüssel bis zu einer expliziten Generierung eines neuen Schlüssels zu bestimmen. Sollte sich ein Nutzer im Anschluss einer Infektion dazu entschließen, seinen Schlüssel für einem bestimmten Tag t zu teilen, ist es ihm nicht möglich, EphIDs von t bis zum aktuellen Tag geheim zu

halten. Der Vorteil gegenüber dem Modell von PEPP-PT ergibt sich trotzdem daraus, dass der Risikofaktor der Nutzer auf deren eigenen Geräten berechnet und nicht geteilt wird und dass keine eindeutigen, zurückverfolgbaren IDs verwendet werden. Anderen Geräten ist es nicht möglich, aus SK_t oder den daraus berechenbaren IDs dieses persönliche Datum zu ermitteln. Eine weitere Verbesserung ist bei Anwendung des zweiten Vorschlags im DP3T-Whitepaper zu beobachten: Zum einen werden hier die EphIDs von Seeds abgeleitet, welche nicht voneinander abhängig sind. Dadurch kann der Nutzer die Zeiträume, zu welchen er im Falle einer Infektion seine Informationen teilen möchte, genauer bestimmen. Zum anderen können andere Nutzer durch Verwendung eines Cuckoo-Filters und mehrerer Hash-Vorgänge (siehe Gleichung 5) die IDs der Infizierten nicht wiederherstellen.

Auch bei dem CWA-Modell werden keinerlei persönliche Daten erfasst. Die im Falle einer Infektion geteilten TEKs werden zumindest täglich erneuert und laut der *Corona-Warn-App Solution Architecture* durch Kryptografie auf den Smartphones erzeugt. Nähere Informationen zu diesem Ableitungsprozess gehen aus dem Dokument leider nicht hervor. Es wird jedoch darauf geachtet, dass nacheinander auf dem Corona-Warn-App-Server eingehende Schlüssel nicht gruppierbar sind, indem sie in zufälliger Reihenfolge aus der Datenbank abgefragt werden.

3.3. Unbeobachtbarkeit der Kommunikation

Es ist wichtig, dass ein Außenstehender durch Beobachten der Kommunikation nicht darauf schließen kann, ob ein Nutzer infiziert ist oder Kontakt zu infizierten Personen hatte. *Außenstehende* sind dabei andere Nutzer, Infrastruktur- und Netzbetreiber und Angreifer mit Einblick in das System. *Kommunikation* beschreibt die Übermittlung von Nachrichten innerhalb des Systems [6].

In dem Whitepaper zu PEPP-PT wird das hierzu relevante Ziel *NF-REQ 9* formuliert: „Infizierte Nutzer sollten nur von der Gesundheitsbehörde persönlich identifizierbar sein.“ [2, S. 4] Um dem näher zu kommen, werden zum Beispiel Noise-Nachrichten bei der Benachrichtigung möglicherweise infizierter Nutzer versandt. Leider wird jedoch gegen Ende der Schrift, in der Diskussion der Sicherheitsziele, festgestellt, dass das Ziel NF-REQ 9 nicht erreicht wurde. So ist es einem Angreifer durch einen Lauschangriff möglich, große Datenpakete festzustellen, welche von einem Smartphone an das Backend verschickt werden. Treten solche Datenströme auf, handelt es sich bei dem Sender um einen infizierten Nutzer, der gerade die notwendigen Daten an einen Server hochlädt [2, S. 22]. Die besondere Größe der versendeten Datenpakete ist darauf zurückzuführen, dass im Infektionsfall alle vom Gerät empfangenen EBIDs eines gewissen Zeitraums an das Backend versandt werden müssen. Es wird zugegeben, dass der Angriff sowohl von Netzbetreibern, als auch von Hackern im Netzwerk durchgeführt werden kann. Als Lösung des Problems wird die Verwendung des Tor-Netzwerks vorgeschlagen, um eine Zuordnung der Datenpakete zu einer Person zu vermeiden. Diese Sicherheitsmaßnahme ist allerdings optional und auf Wunsch durch den Nutzer selbst vorzunehmen.

Bei der dezentralen DP3T-Lösung werden im Falle eines positiven Testergebnisses nur geringe Datenmengen ((SK_t, t)-Wertepaare beziehungsweise $\{(i, seed_i)\}$-Wertepaarmengen) an das Backend verschickt. Da jedoch ausschließlich bei einer Infektion Daten an einen Server verschickt werden, ist es auch hier möglich, durch Belauschen der Verbindung auf einen Infektionsfall bei dem Nutzer zu schließen. Dies wird in [3] auf Seite 31 bestätigt.

Eine mögliche Lösung dieses Problems durch die Anwendung wäre das regelmäßige Senden von Daten zum und vom Backend. Diese Daten können entweder relevante Daten, wie Informationen über eine Infektion des Nutzers beziehungsweise anderer Nutzer oder nicht verwertbare Platzhalterdaten derselben Größe sein. Durch Verschlüsselung wären gültige nicht von ungültigen Daten zu unterscheiden. Der erhöhte Übermittlungsverkehr sollte nicht zu Speicherplatzproblemen führen, da die zusätzlichen Pseudo-Daten nicht gespeichert werden müssen, er führt jedoch zu erhöhter Netzlast und zu einer erhöhten Last für die Server.

Dieser Ansatz wird von der Deutschen Telekom AG und der SAP SE in ihrer Corona-Warn-App umgesetzt: Hier werden regelmäßig Datenpakete an den CWA-Server geschickt, welche im Falle einer aktuellen Infektion die Diagnosis Keys und andernfalls zufällige Daten enthalten. Dem Server ist es problemlos möglich, zwischen den beiden zu unterscheiden, einem Lauscher jedoch auf Grund der verschlüsselten Übertragung nicht. Weiter erfolgt die Übermittlung der Testergebnisse nicht durch das Gerät des Nutzers, sondern durch das Labor, wodurch auch hier keine Zuordnung anhand von Verbindungsdaten möglich ist. Die weitere Kommunikation zwischen Anwendung und Backend variiert für die verschiedenen möglichen Zustände des Patienten nicht.

4. Diskussion

Gerade in Zeiten von Fake News und verbreiteter Skepsis gegenüber der medizinischen Forschung rund um Corona ist es wichtig, eine möglichst attraktive und wirkungsvolle Anwendung zur Ausbreitung des Virus zu schaffen. Die Wirksamkeit des Einsatzes von den hier angesprochenen Anwendungen ist ohnehin umstritten. So ist der IT-Security-Spezialist Bruce Schneier überzeugt, dass ein effektives Tracing via App nicht möglich ist. Er begründet dies mit dem, laut ihm, hohen Risiko für False Positives und False Negatives, welche man sich mit der Wahl von GPS oder Bluetooth als Tracing-Technologie inhärent einkaufe [8]. Ein tatsächlicher Nutzen lässt sich aber üblicherweise erst durch ausführliche Tests feststellen. Um jedoch überhaupt eine Chance zu haben, ist eine ausreichende Verbreitung der Anwendung ausschlaggebend, denn nur wenn beide Parteien in einer Konfrontation die gleiche App in Benutzung haben, kann eine mögliche Ansteckungsgefahr erkannt werden. Für den Erfolg der Anwendung sollte daher dringend vermieden werden, dass sich potentielle Nutzer Sorgen um ihre Privatsphäre beziehungsweise die Sicherheit ihrer Daten machen müssen.

Mitte April plante das Bundesgesundheitsministerium die Einführung einer Anwendung auf Basis der PEPP-PT-Initiative. Wie in Punkt 3 aufgeschlüsselt und in Tabelle 1 übersichtlich dargestellt, wird hier jedoch keines der analysierten Kriterien erfüllt. So existiert zum Beispiel ein zentraler Server, der eindeutig zu Nutzern zuordenbare

Richtlinie	Technologie			
	PEPP-PT	DP3T Variante 1	DP3T Variante 2	CWA
Kein zentraler Vertrauenspunkt	nein	ja	ja	ja
Lokale Berechnung der Risikofaktoren	nein	ja	ja	ja
Keine eindeutige Nutzerkennungen	nein	ja	ja	ja
Unabhängigkeit versendeter Kennungen	nein	nein	ja	ja
Unbeobachtbarkeit der Kommunikation	nein	nein	nein	ja

Tabelle 1: Beurteilungsübersicht

IDs speichert, und es ist mittels Lauschangriff feststellbar, welche Nutzer infiziert sind. Dies resultierte in einem offenen Brief an den Herrn Bundesminister Spahn und den Herrn Kanzlerminister Braun, angestoßen vom Chaos Computer Club. In diesem werden die Empfänger gebeten, die Forderungen der Experten, wie beispielsweise die Prüfsteine des CCC [6], ernst und von PEPP-PT Abstand zu nehmen [7].

Die beiden Ansätze um DP3T verbessern viele der Schwachstellen von PEPP-PT. Es hängt zum Beispiel die Anonymität nicht mehr von dem Vertrauen zu einem zentralen Server ab und Ansteckungsrisiken werden lokal auf dem Smartphone berechnet. Die beiden vorgestellten Varianten haben jeweils ihre Vor- und Nachteile, so priorisiert die erste Effizienz und die zweite Sicherheit, wobei das Sicherheitsniveau beider gut ist. Auch ist der zweite Ansatz von DP3T der einzige hier besprochene, bei dem der Nutzer im Falle einer Infektion Informationen für ausgewählte Zeiträume zurückhalten kann. Ein großer Nachteil von DP3T bleibt jedoch bestehen: Ein Lauschangriff, welcher zum Ziel hat, infizierte Personen durch bestimmte verschickte Datenpakete zu identifizieren, ist weiterhin möglich.

Die CWA nimmt sich dieses Problems an und übernimmt gleichzeitig einen Großteil der Vorzüge der Technologie. Durch Senden von Platzhalter-Datenpaketen werden Lauschangriffe unterbunden. Zudem sind keine Bottlenecks in der Infrastruktur zu vermuten, da das Backend in der Telekom-Cloud betrieben wird [4]. Es ist deshalb als sehr positiv zu betrachten, dass sich die Bundesregierung tatsächlich von der Idee PEPP-PT zu verwenden hat abbringen lassen und am 16.06.2020 die Corona-Warn-App veröffentlicht wurde [9].

5. Ausblick

Innerhalb des ersten Tages wurde die Corona-Warn-App bereits über sechs Millionen [9], innerhalb der ersten 48 Stunden etwa acht Millionen Mal heruntergeladen [10]. Die Anwendung hatte also einen guten Start, wenn auch nicht alle daran teilhaben konnten. Auf Smartphones von Apple funktioniert sie nämlich erst ab iOS-Version 13.5 und damit nur auf dem iPhone 6s oder neueren Modellen [11]. Diese Einschränkung liegt wohl dem Fehlen von technischen Spezifikationen älterer Hardware zugrunde.

Jetzt bleibt abzuwarten, wie wirksam die App ist und ob ihre Kritiker in der Befürchtung, dass die eingesetzten Technologien viel zu ungenau sind, Recht haben oder ob sich doch effektiv Infektionsketten brechen lassen. Der Grad des Erfolges hängt dabei natürlich von der Verbreitung der Anwendung ab und es kann nur gehofft werden, dass sich ein Großteil der Bevölkerung für einen Download entscheiden wird. Auch ein eventueller Erfolg sollte aber nicht Anlass dazu geben, andere Sicherheitsmaßnahmen zur Bekämpfung des Virus, wie der Schutz durch Masken oder das regelmäßige Händewaschen, zu vernachlässigen. Es ist zu hoffen, dass durch den gemeinsamen Einsatz verschiedener Mittel eine zweite große Welle der Corona-Pandemie verhindert werden kann.

6. Conclusion

In diesem Paper wurden drei Apps zum Corona-Tracing unter Verwendung verschiedener Prüfsteine untersucht. Als wichtigste Eigenschaft einer solchen Anwendung wurde dabei ein dezentraler Aufbau herausgearbeitet. Aus diesem Grund wurde die Entscheidung der Bundesregierung, von der ursprünglich geplanten, voreiligen Einführung von PEPP-PT abzusehen und stattdessen mehr Ressourcen in die Entwicklung einer überarbeiteten Lösung zu investieren, stark befürwortet. Die nun auf den Markt gebrachte Corona-Warn-App beurteile ich aus Datenschutzsicht als unbedenklich und rate zu deren Verwendung.

Danksagung

Vielen Dank an Dominik Okwieka für das Lektorieren meiner Arbeit trotz Prüfungsstress.

Literatur

[1] C. Lang. (2020). Hackathon - #WirVSVirus, 4Germany UG, Adresse: https://wirvsvirus.org/hackaton/ (besucht am 20. 05. 2020).

[2] PEPP-PT, „Data Protection and Information Security Architecture, Illustrated on German Implementation“, Techn. Ber., 20. Apr. 2020. Adresse: https://github.com/pepp-pt/pepp-pt-documentation/blob/master/10-data-protection/PEPP-PT-data-protection-information-security-architecture-Germany.pdf (besucht am 20. 05. 2020).

[3] C. Troncoso, M. Payer, J.-P. Hubaux, M. Salathé, J. Larus, E. Bugnion, W. Lueks, T. Stadler, A. Pyrgelis, D. Antonioli, L. Barman, S. Chatel, K. Paterson, S. Čapkun, D. Basin, J. Beutel, D. Jackson, B. Preneel, N. Smart, D. Singelee, A. Abidin, S. Guerses, M. Veale, C. Cremers, R. Binns und C. Cattuto, „Decentralized Privacy-Preserving Proximity Tracing“, Techn. Ber., 12. Apr. 2020. Adresse: https://github.com/DP-3T/documents/blob/master/DP3T%20White%20Paper.pdf (besucht am 20. 05. 2020).

[4] Deutsche Telekom AG und SAP SE. (20. Mai 2020). Corona-Warn-App, README, Adresse: https://github.com/corona-warn-app/cwa-documentation/blob/master/README.md (besucht am 21. 05. 2020).
[5] ——, (12. Juni 2020). Corona-Warn-App Solution Architecture, Adresse: https://github.com/corona-warn-app/cwa-documentation/blob/master/solution_architecture.md (besucht am 14. 06. 2020).
[6] linus. (6. Apr. 2020). 10 Prüfsteine für die Beurteilung von „Contact Tracing"-Apps, Chaos Computer Club e. V., Adresse: https://www.ccc.de/de/updates/2020/contact-tracing-requirements (besucht am 28. 05. 2020).
[7] D64 – Zentrum für digitalen Forschritt e.V., LOAD e.V. – Verein für liberale Netzpolitik, Forum InformatikerInnen für Frieden und gesellschaftliche Verantwortung e.V., Gesellschaft für Informatik (GI) e.V., Chaos Computer Club e. V. (CCC) und Stiftung Datenschutz. (24. Apr. 2020). Offener Brief: Geplante Corona-App ist höchst problematisch, Adresse: https://www.ccc.de/system/uploads/300/original/Offener_Brief_Corona_App_BMG.pdf (besucht am 07. 05. 2020).
[8] B. Schneier. (1. Mai 2020). Schneier on Security, Me on COVID-19 Contact Tracing Apps, Adresse: https://www.schneier.com/blog/archives/2020/05/me_on_covad-19_.html (besucht am 16. 06. 2020).
[9] K. B. Becker, „Erste Zwischenbilanz, Corona-App bereits 6,5 Millionen Mal heruntergeladen", *Frankfurter Allgemeine*, 17. Juni 2020. Adresse: https://www.faz.net/aktuell/politik/inland/corona-app-wurde-bereits-6-5-millionen-mal-heruntergeladen-16818946.html (besucht am 19. 06. 2020).
[10] D. Rzepka, „Erfolgreicher Start, Corona-App acht Millionen Mal runtergeladen", *ZDF*, 18. Juni 2020. Adresse: https://www.zdf.de/nachrichten/politik/corona-app-millionen-downloads-100.html (besucht am 21. 06. 2020).
[11] Apple Inc. (16. Juni 2020). Corona-Warn-App, Gemeinsam Corona bekämpfen. Version 1.0.2, Adresse: https://apps.apple.com/de/app/corona-warn-app/id1512595757 (besucht am 21. 06. 2020).