So funktioniert eine Corona-Tracing-App, die Ihre Privatsphäre schützt

Dank Kryptografie lassen sich Begegnungen anonym und dezentral aufzeichnen und auswerten. Wir zeigen, wie das funktioniert – in acht Schritten.

Von Patrick Recher (Text) und Anna Traussnig (Illustrationen), 16.04.2020

Registrieren, wer zu welcher Zeit mit wem in Kontakt war. Mutmassen, wer sich mit dem Coronavirus angesteckt haben könnte. Melden, wer wahrscheinlich besser zu Hause bleiben sollte, um weitere Infektionen zu verhindern. Und all dies auf anonymer, weitgehend dezentraler Basis, unter Wahrung der Privatsphäre.

Das wollen sogenannte *Proximity-Tracing*-Apps ermöglichen. Sie werden zurzeit von Firmen und Universitäten entwickelt und sollen der Gesellschaft helfen, die Corona-Epidemie einzudämmen, ohne Nutzerdaten zentral auf Servern zu sammeln.

Ob das realistisch ist und worauf es bei der Umsetzung ankommt, lesen Sie in der Analyse «Das grosse sozial-digitale Experiment».

In diesem Beitrag zeigen wir auf, wie solche Apps technisch funktionieren. Schritt für Schritt – sodass Sie es in einer Viertelstunde verstehen.

Unser Anschauungsbeispiel ist das Projekt STAR, aus dem die Prototyp-App «Next Step» hervorging. Diese wurde vor kurzem von der Schweizer Firma Ubique zu Testzwecken veröffentlicht. Inzwischen haben sich die Entwickler dem Projekt DP3T von Forschern der beiden ETH angeschlossen. Der Code der «Next Step»-App wurde dabei übernommen und wird nun gemeinsam weiterentwickelt. Das System, das wir hier beschreiben, könnte also bald in der Schweiz zum Einsatz kommen. In der Dokumentation von DP3T wird allerdings bereits eine Variante beschrieben, welche die Privatsphäre noch besser schützt, jedoch um einiges komplizierter ist.

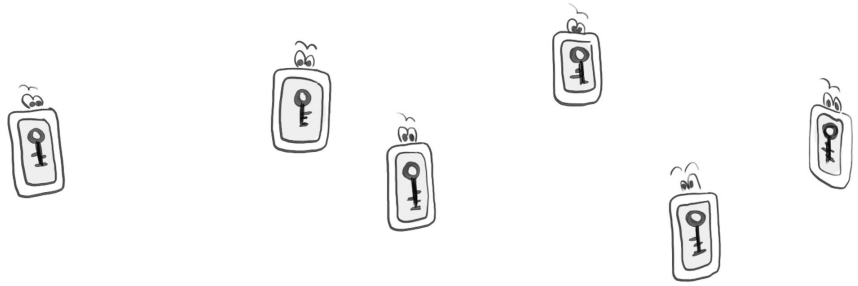
Die App zeichnet Kontakte auf und schlägt Alarm, falls ein aufgezeichneter Kontakt später eine positive Covid-19-Testmeldung eingibt.

Damit das klappt, werden folgende acht Schritte durchlaufen.

1. Schlüssel generieren

Jede App-Installation generiert beim ersten Start einen geheimen Schlüssel und speichert diesen lokal auf dem Smartphone ab. Dieser Schlüssel ist

rein zufällig und beinhaltet keine privaten Daten. So wird eine bestimmte App-Installation anonym identifiziert. Hier ein reales Beispiel: «44c6dfb4-cbdbea397122d47f9e0bfe397aafcad3a38db91a13d617ec4a3cfa19».

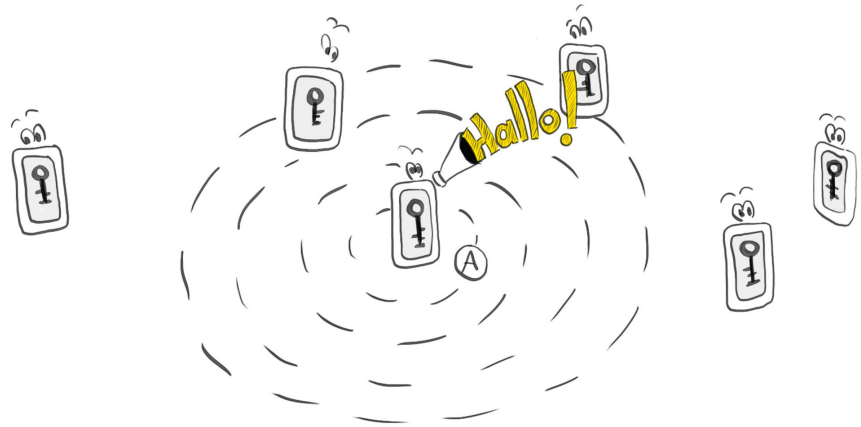


Ich wills genauer wissen: Schlüssel

Ein digitaler Schlüssel ist nichts weiter als eine lange, zufällige Zeichenkette. Der hier gezeigte Schlüssel ist 256 Bit lang, das ergibt 2^{256} = 1.156*-10^{77} mögliche Schlüssel, was eine unglaublich grosse Zahl mit 78 Stellen ist. Wäre jedes Atom der Erde ($1*10^{50}$) ein Computer, der pro Sekunde 10 Milliarden Schlüssel generieren könnte, würde es immer noch 36 Milliarden Jahre dauern, bis alle Schlüssel errechnet wären. Die Chance, dass zwei Geräte zufällig den gleichen Schlüssel haben, ist damit gleich null.

2. Sich bemerkbar machen

Nachdem eine App (A) gestartet wurde, sendet sie alle paar Sekunden per Bluetooth ein Datenpaket in die nahe Umgebung aus: Ein kleines «Hallo, hier ist die STAR-App».



Ich wills genauer wissen: Bluetooth

Bluetooth ist ein Funkstandard zur lokalen, direkten Datenübertragung zwischen zwei Geräten und wurde in den 1990er-Jahren entwickelt. Hier kommt die moderne Erweiterung Bluetooth-Low-Energy zum Einsatz, die Strom spart und sich deshalb dazu eignet, ständig Pakete zu senden und zu empfangen.

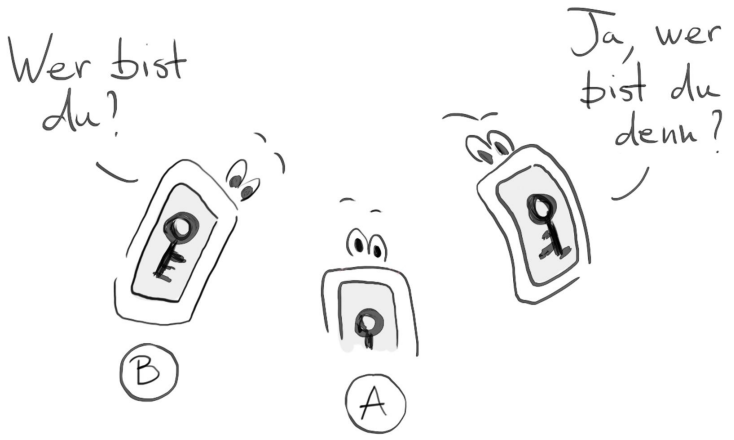
Zwei kommunizierende Geräte können abschätzen, wie weit sie voneinander entfernt sind. Dazu schickt der Sender die ausgehende Signalstärke mit. Das andere Gerät kann anhand des Pegels des empfangenen Signals (RSSI: Received Signal Strength Indicator) bestimmen, wie stark das Signal während der Übertragung abgeschwächt wurde, und deshalb die Distanz bestimmen.

Das Problem ist allerdings, dass Bluetooth (2,4 Ghz ~10 cm Wellenlänge) in einem Bereich sendet, der von Hindernissen (Gewebe, Gebäude etc.) stark absorbiert wird, und dass Reflexionen des Signals die genaue Distanzmessung erschweren. Entsprechend werden bei STAR alle Pakete ungeachtet der Distanzmessung aufgezeichnet.

3. Nach der Nummer fragen

Empfängt eine andere App (B) ein solches «Hallo», schickt sie eine Antwort an den Absender A zurück – und bittet diesen um zwei Angaben, um sich den Kontakt merken zu können:

1. um einen sogenannten Zeitstempel: die aktuelle Zeitangabe nach dem Muster «2020-04-07T22:13:00», und um
2. eine sogenannte Prüfsumme (was das ist, erfahren Sie gleich).



4. Prüfsumme ausrechnen

Die Absender-App A empfängt die Bitte und macht sich sofort an die Arbeit: Sie nimmt den aktuellen Zeitstempel und den persönlichen Schlüssel und gibt beides in die kryptografische Funktion HMAC ein. Heraus kommt eine Prüfsumme.

Das ist wiederum eine lange, (auch für Computer) zufällig aussehende Zeichenkette. Als Beispiel hier die echte Prüfsumme für den obigen Schlüssel und den Zeitstempel:

HMAC (Schlüssel + Zeitstempel) = Prüfsumme

HMAC (44c6df... + 2020-04-07T22:13:00) = 923a3b223fd620e788583abac375-8410e724481a1756af514e588f3f7427317d

Der Clou der HMAC-Funktion ist, dass für den gleichen Schlüssel und Zeitstempel immer die gleiche Prüfsumme herauskommt – und für einen leicht anderen Schlüssel oder Zeitstempel eine ganz andere Prüfsumme. Die Funktion ist auch nicht umkehrbar: Man kann sie nicht zurückrechnen.

Man kann sich die Prüfsumme wie ein Schloss vorstellen, das nur mit dem richtigen Zeitstempel *und* dem richtigen Schlüssel geöffnet werden kann.

Ich wills genauer wissen: HMAC

Das verwendete Prinzip HMAC wurde 1996 erfunden. HMAC bedeutet Keyed-Hash Message Authentication Code, in etwa: Nachrichtenauthentifizierungscode, der auf einem Schlüssel und einer kryptografischen Hash-Funktion beruht.

Die Funktion nimmt als Eingabe einerseits einen Schlüssel und andererseits eine Nachricht (in unserem Fall den Zeitstempel) und liefert als Ergebnis eine Prüfsumme, die zufällig aussieht und unabhängig von Schlüssel und Nachricht immer gleich lang ist. Man spricht deshalb von einer *compression function,* die Informationen werden komprimiert.

Die interne Verschaltung garantiert, dass für die gleiche Nachricht und den gleichen Schlüssel immer dieselbe Prüfsumme generiert wird. Für eine leicht andere Nachricht oder einen anderen Schlüssel eine komplett andere Prüfsumme.

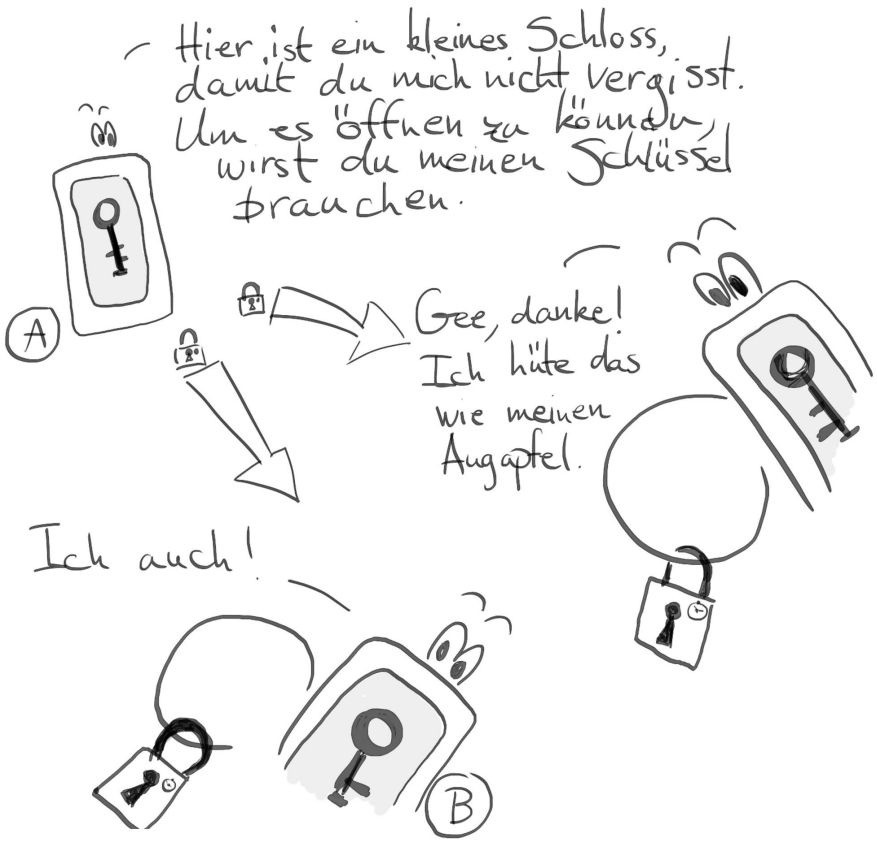
Es gibt keinen praktischen Weg, das zurückzurechnen. Sprich: Selbst wenn jemand Schlüssel und Prüfsumme oder Nachricht und Prüfsumme kennt: Der fehlende Teil kann nicht herausgefunden werden. Dies wird erreicht, indem bei der Verarbeitung der Nachricht zusammen mit dem Schlüssel Information verloren geht. Sie können sich das grob so vorstellen: Sie dividieren die Nachricht (als Zahl) durch den Schlüssel – und streichen den Rest. Dann teilen Sie dieses Zwischenresultat wieder durch den Schlüssel und streichen wiederum den Rest. Nun machen sie das 1000-mal, am Ende haben Sie die Prüfsumme. Wenn Sie nun wieder von vorne beginnen, werden Sie immer zum gleichen Ergebnis kommen, aber nie vom Ergebnis zurück zu den Anfangszahlen.

Für das regelmässige Erzeugen von Prüfsummen basierend auf einem fortlaufenden Zeitstempel existiert eine spezielle Version von HMAC:

HMAC-Based One-Time Password Algorithm – HOTP, diese wird von STAR verwendet.

Die App A schickt nun diese zwei Angaben, also den Zeitstempel und die Prüfsumme, an App B.

Den Schlüssel, den A zur Berechnung der Prüfsumme verwendet hat, behält sie hingegen weiterhin für sich.

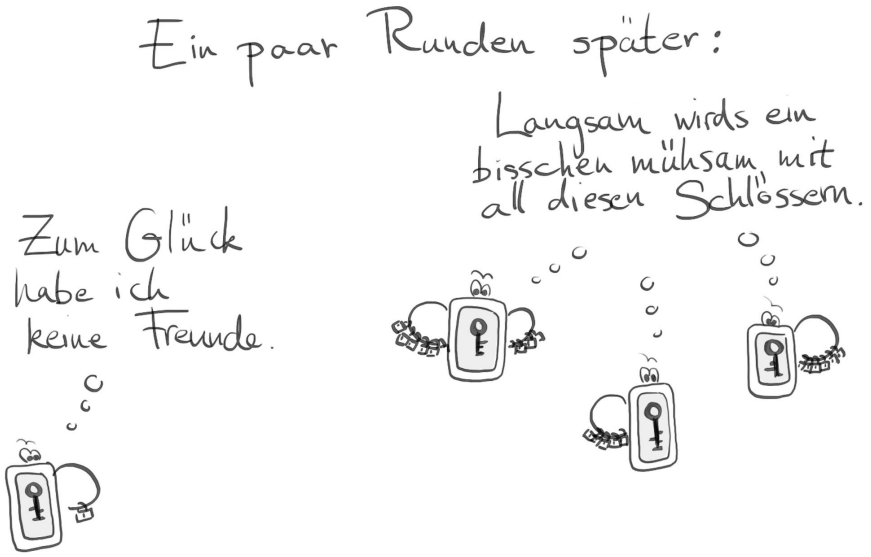


5. Angaben abspeichern

Die App B empfängt Zeitstempel und Prüfsumme und legt sie im lokalen Speicher ab.

Gleichzeitig findet der Austausch umgekehrt statt: Also B generiert aus *seinem* Schlüssel und dem aktuellen Zeitstempel eine Prüfsumme und schickt Zeitstempel und Prüfsumme an A.

So läuft das für jede Begegnung zweier Apps ab. Mit der Zeit sammeln sich auf einem Smartphone mehrere hundert oder tausend Zeitstempel-Prüfsummen-Einträge an. Einträge, die älter sind als 14 Tage, werden von den Apps automatisch wieder gelöscht.

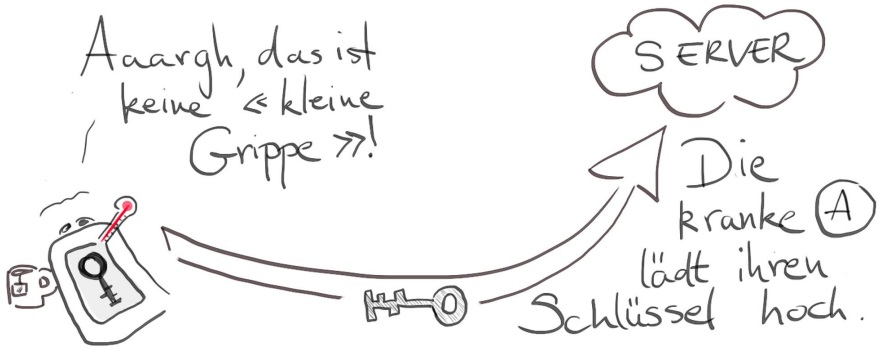


6. Infektion melden

Nun nehmen wir an, dass die Nutzerin von App A positiv auf Covid-19 getestet wird. Als verantwortungsvolle Mitbürgerin öffnet sie ihre App und wählt dort die Option «Ich wurde infiziert».

Nun geschieht Folgendes:

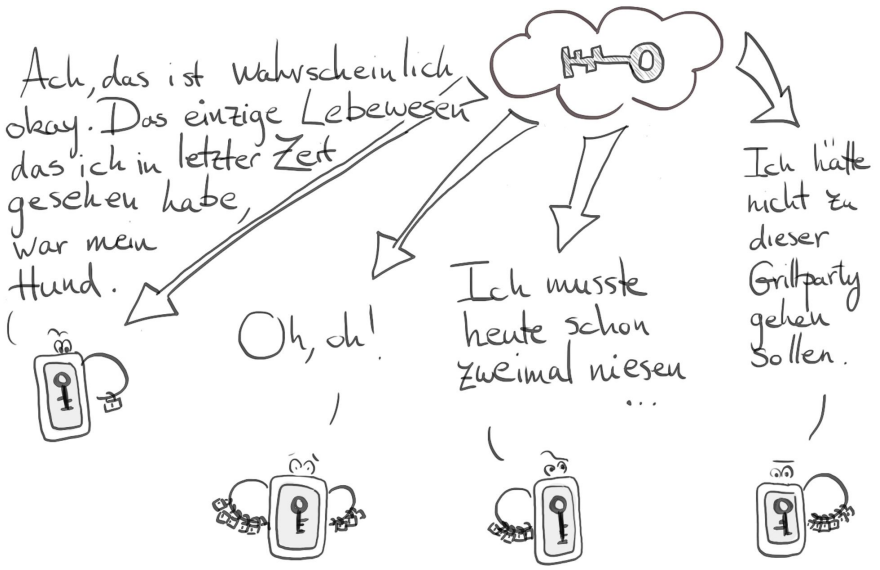
Die App A schickt den geheimen Schlüssel, mit welchem sie zuvor alle Prüfsummen generiert hat, an einen zentralen Server. Das ist bezüglich Datenschutz unproblematisch, denn der Schlüssel ist nur eine zufällige Zeichenkette und lässt keinen Rückschluss auf die infizierte Person zu.



Damit nur Personen, die tatsächlich positiv getestet wurden, ihren Schlüssel hochladen, könnte der Server zur Eingabe eines Passworts auffordern. Infizierte Personen würden dieses zum Beispiel in Form eines anonymen Passwort-Coupons vom medizinischen Personal ausgehändigt bekommen.

7. Schlüssel herunterladen

Alle Apps stehen von Beginn an in Dialog mit dem Server. Sie laden periodisch alle Schlüssel herunter, die von infizierten Personen veröffentlicht wurden. So erreicht der Schlüssel von A nach kurzer Zeit auch B.

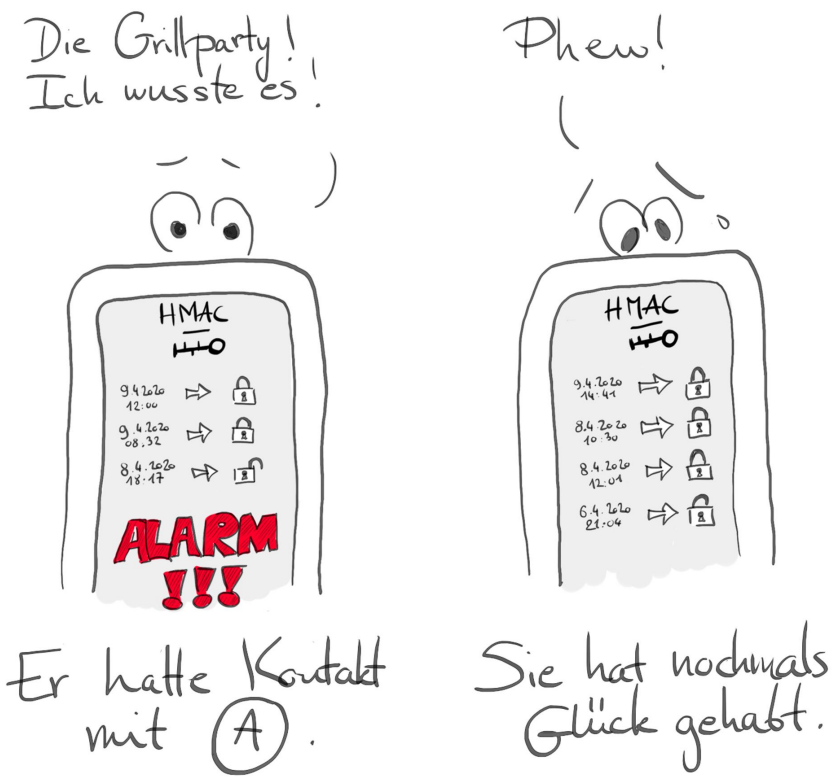


8. Schlüssel und Schlösser abgleichen

Nun muss die App B herausfinden, ob es auf «ihrem» Smartphone einen Zeitstempel-Prüfsummen-Eintrag gibt, auf den dieser Schlüssel passt. In anderen Worten: ob B ein Schloss gesammelt hat, welches sich mit dem heruntergeladenen Schlüssel öffnen lässt.

Die App nimmt dazu den Schlüssel und geht der Reihe nach jeden Eintrag im Speicher durch. Sie gibt den jeweiligen abgespeicherten Zeitstempel zusammen mit dem heruntergeladenen Schlüssel in die HMAC-Funktion ein und generiert so eine Prüfsumme. Ist diese identisch mit der zum entsprechenden Zeitstempel abgespeicherten Prüfsumme, weiss B, dass es mit dem Besitzer des heruntergeladenen Schlüssels einen Kontakt gab – genau zu dem Zeitpunkt, an dem der betreffende Zeitstempel generiert wurde.

Es ist auch möglich, dass B mehrere Einträge findet, auf die der Schlüssel passt. Das bedeutet, dass es mehrere Kontakte zu A gab. Dies würde sehr wahrscheinlich auf ein höheres Infektionsrisiko hindeuten. Anhand der Zeitstempel kann angezeigt werden, wann und wie lange die Kontakte stattfanden.



Der User der App B weiss nun, wann und wie lange ein verdächtiger Kontakt stattgefunden hat. Wer die betreffende Person ist und ob sie sich beim Kontakt angesteckt haben könnte, muss sich die Anwenderin selbst überlegen.

Passt ein heruntergeladener Schlüssel auf keinen Eintrag, passiert nichts. Dann hatte der User der App B keinen Kontakt mit der Userin der App A.

Fazit

Begegnungen speichern, Infekte melden, Alarm schlagen – ohne Nennung von Namen oder zentraler Speicherung von persönlichen Daten: Apps wie «Next Step» zeigen einen Lösungsweg. Alles, was es dazu braucht, ist ein gemeinsames Protokoll, etwas Kryptografie und einen Server, der ein Minimum an anonymer Information weiterleitet.

Die Frage, ob digitales *Contact Tracing* mit den Grundrechten vereint werden kann, hat somit eine eindeutige Antwort: ja. Rüstet man Smartphones mit Epidemiesoftware aus, so landet man also nicht zwingend in einem Überwachungsstaat.

Entscheidend ist, dass die Apps von Grund auf mit dem Ziel entwickelt werden, die Privatsphäre zu schützen, und dass der Quellcode offen liegt, sodass Softwareentwicklerinnen die Funktionsweise überprüfen können.

In einer ersten Version schrieben wir, dass die App wie beschrieben in der Schweiz eingesetzt werden könnte. Wir haben dies dahin gehend präzisiert, dass sie noch weiterentwickelt wird.