verifoo: Fachliche Dokumentation

Stand: 30.05.2021

Kurzbeschreibung:

verifoo ist eine Produkt zur Speicherung und Validierung von Testergebnissen.

Testergebnisse werden zusammen mit Daten, die eine getestete Person eindeutig identifizieren, von einem Testzentrum an *verifoo* gesendet und verarbeitet. Die getestete Person erhält dann einen QR-Code. Ob ein Testergebnis gültig ist, kann durch Vorzeigen des QR-Codes ermittelt werden. Dazu werden die im QR-Code hinterlegten Daten zur Prüfung an die Schnittstelle von *verifoo* gesendet.

Anlage und Verarbeitung eines Testergebnisses

Gegeben sei ein an das System übermittelter Datensatz personenbezogener Daten P sowie ein dazugehöriger Testdatensatz T, definiert als:

1. $P = (Vorname, Nachname, PLZ, Geburtsdatum, PIN)$
2. $T = (Testzeitpunkt\ Z, Testergebnis, ID\ des\ Testzentrums)$

Aus dem Datensatz P wird mit einem für alle P geltenden *Salt*[1] S_G und einer mathematischen Einwegefunktion[2] H ein *Hash*-Wert $Hash_P$ erzeugt:

3. $Hash_P = H_{SHA3}(P \otimes S_G)$

Außerdem wird aus *Vorname* und *Nachname* des Datensatzes P ein weiterer *Hash*-Wert $Hash_N$ erzeugt. Hierbei wird ein je für P individueller Salt[3] S_P verwendet:

4. $Hash_N = H_{SHA3}(\{Vorname, Nachname\} \otimes S_P)$

Sowohl S_G als auch der je Datensatz individuell generierte *Salt* S_P sind 128 Bit lang.

Werden die übermittelten Datensatz P und T erfolgreich durch das System verarbeitet, werden folgende Referenzdatensätze P_D und T_D in der Datenbank angelegt:

5. $P_D = (Hash_P, Hash_N, S_P, PLZ)$
6. $T_D \equiv T$, da keine weiteren personenbezogenen Daten erfasst werden.

---

1 Ein *Salt* ist ein zusätzlicher Parameter, mit dem Daten im Klartext angereichert werden, bevor sie durch eine mathematische Einwegefunktion (*Hash*-Funktion) verarbeitet werden, um Rückschlüsse auf die Ursprungsdaten zusätzlich zu erschweren.

2 *Hash*-Funktion, hier *SHA-256 (SHA-2)*. Theoretisch ist es nicht möglich, eine Hash-Funktion umzukehren.

3 Der individuelle Salt wird erzeugt, wenn $Hash_N$ erzeugt wird.

Aus P kann nur dann ein Datensatz P_D mit $Hash_P$ aus der Datenbank abgeleitet werden, wenn eine Person eine *PIN* angegeben hat. Die übermittelte *PIN* ist optional und ein numerischer, vierstelliger Wert. Er kann von der Person frei gewählt werden und dient dazu, einen obfuskierten Datensatz, also $Hash_P$, zu identifizieren. Dies ist notwendig, wenn eine Person einen einzigen QR-Code zur Validierung mehrerer Testergebnisse verwenden möchte.

Gibt die Person keine *PIN* an, wird stattdessen eine 128 Bit lange *UUID (Universally Unique Identifier)* verwendet. Eine *UUID* ist ein praktisch eindeutiger Schlüssel, der hier zur Anwendung kommt, um keine Rückschlüsse auf einen Datensatz P einer Person zuzulassen, die keine *PIN* angegeben hat. Das wird dadurch erreicht, dass die verwendete UUID praktisch eineindeutig ist und nicht gespeichert wird.

Wird eine *PIN* angegeben, ist ein Rückschluss auf personenbezogene Daten ebenfalls nicht möglich, da die PIN nur der entsprechenden Person bekannt ist. Auch diese wird nicht gespeichert.

Existiert zu P Datensatz P_D im System, wird ihm der neu übermittelte Datensatz T_D angehangen. Andernfalls wird zuvor ein neuer Datensatz P_D erzeugt.

Einem Datensatz P_D können 1 bis beliebig viele Testdatensätze T_D zugeordnet sein.

Anschließend erzeugt das System einen QR-Code Q und gibt diesen zurück. Dieser ist wie folgt definiert:

7. $Q = (Vorname, Nachname, Hash_P)$

Q wird durch das System generiert, aber nicht gespeichert. $Hash_P$ ist durch Einbettung in Q öffentlich bekannt. S_G und S_P werden nicht übermittelt und werden geheim gehalten.

Weitere Erläuterungen:

- Ein Testergebnis ist definiert als ein Zustand $E \in \{positiv, negativ, unbekannt\}$.
- Es ist vorgesehen, die Postleitzahl der Person aus statistischen Gründen zu speichern. Sie allein lässt keine Rückschlüsse auf einen personenbezogenen Datensatz P zu
- Die *ID des Testzentrums* ist optional und wird ggf. von einem Testzentrum übermittelt. Auch sie lässt keine Rückschlüsse auf personenbezogene Daten zu und wird ausschließlich aus statistischen Gründen erfasst.

Validierung eines QR-Codes

Gegeben sei ein übermittelter Datensatz $Q = (Vorname, Nachname, Hash_P)$ sowie ein Gültigkeitszeitraum G <= 72 Stunden. Des Weiteren sei J als aktueller Zeitpunkt definiert.

Über den in *Q* enthaltenen $Hash_P$ wird versucht, einen bereits gespeicherten Datensatz P_D zu ermitteln. Falls ein solcher Datensatz gefunden wurde, wird ein *Hash*-Wert $Hash_{N'}$ aus den in Q enthaltenen Vornamen und Nachnamen berechnet.

Dieser wird auf Gleichheit mit dem bereits gespeicherten $Hash_N$ geprüft. Falls kein passender Datensatz in der Datenbank gefunden wurde oder wenn $Hash_{N'} \neq Hash_N$, wird die Validierung nicht fortgesetzt und ein negatives Validierungsergebnis R_N zurückgegeben.

Dieser Prüfungsschritt ist notwendig, um festzustellen, ob der Vor- und Nachname mit dem in *Q* enthaltenen Hash-Wert übereinstimmt. Dadurch kann ausgeschlossen werden, dass ein Dritter einen QR-Code mit eigenem Namen und einem $Hash_P$ einer anderen Person erzeugt.

Falls die Prüfung erfolgreich war, wird der neueste Testdatensatz T_D zu P_D und daraus der hinterlegten Testzeitpunkt Z_T sowie das Testergebnis E_T ermittelt.

Die Validierung ist erfolgreich, wenn gilt: $(Z_T + G > J) \wedge (E_T = negativ)$. In diesem Fall wird ein positives Validierungsergebnis zurückgegeben.

Ein negatives Validierungsergebnis ist definiert als $R_N = (falsch)$ und ein positives Validierungsergebnis als $R_P = (wahr, Z_T)$.

Löschen von Datensätzen

Alle Testergebnisse, werden nach einer definierten Frist F = 14 Tage automatisiert vom System gelöscht. Ein Testergebnis wird demnach gelöscht, wenn gilt: $Z(T_D) + F < J$. Datensätze mit obfuskierten personengezogenen Daten werden ebenfalls gelöscht, wenn ihnen keine Testergebnisse mehr zugeordnet sind.

Alle Datensätze werden alle drei Stunden auf o.g. Kriterien geprüft.

Zugriffsberechtigungen

Für die Nutzung der oben beschriebenen Funktionalitäten ist eine Authentifizierung gegenüber dem System notwendig. Jeder Funktionalität ist außerdem einer Berechtigungsgruppe zugeordnet.

Dies bedeutet, dass ein dem System gegenüber authentifizierter Benutzer der entsprechenden Berechtigungsgruppe angehören muss, um die entsprechende Funktion nutzen zu können.