(12) NACH DEM VERTRAG ÜBER DIE INTERNATIONALE ZUSAMMENARBEIT AUF DEM GEBIET DES PATENTWESENS (PCT) VERÖFFENTLICHTE INTERNATIONALE ANMELDUNG

(19) Weltorganisation für geistiges Eigentum
Internationales Büro

(43) Internationales Veröffentlichungsdatum
03. Oktober 2024 (03.10.2024)

WIPO | PCT

(10) Internationale Veröffentlichungsnummer
WO 2024/200764 A1

(51) Internationale Patentklassifikation:
H04L 9/32 (2006.01) *G06F 21/34* (2013.01)
H04W 12/06 (2021.01) *G06F 21/45* (2013.01)
G06F 21/31 (2013.01) *H04L 9/40* (2022.01)

(21) Internationales Aktenzeichen: PCT/EP2024/058675

(22) Internationales Anmeldedatum:
28. März 2024 (28.03.2024)

(25) Einreichungssprache: Deutsch

(26) Veröffentlichungssprache: Deutsch

(30) Angaben zur Priorität:
LU103094 29. März 2023 (29.03.2023) LU

(71) Anmelder: **HEYLOGIN GMBH** [DE/DE]; Sophienstrasse 40, 38118 Braunschweig (DE).

(72) Erfinder: **SCHÜRMANN, Dominik**; Sophienstrasse 40, 38118 Braunschweig (DE). **BREITMOSER, Vincent**; Sophienstrasse 40, 38118 Braunschweig (DE).

(74) Anwalt: **WHITE IP I PATENT & LEGAL GMBH**; Königstraße 7, 01097 Dresden (DE).

(81) Bestimmungsstaaten *(soweit nicht anders angegeben, für jede verfügbare nationale Schutzrechtsart)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(54) **Title:** INNOVATIVE SERVER-BASED METHOD FOR MANAGEMENT OF CONFIDENTIAL DATA

(54) **Bezeichnung:** INNOVATIVES SERVERBASIERTES VERFAHREN ZUM MANAGEMENT GEHEIMER DATEN

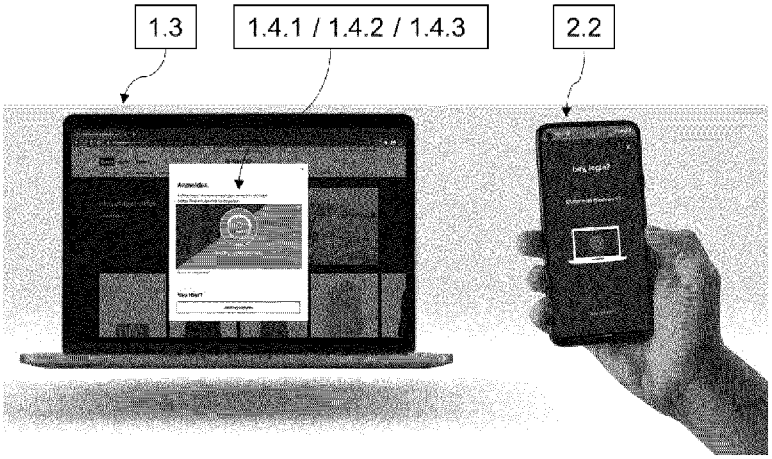

Fig. 1

(57) **Abstract:** The present invention relates to a computer-implemented method for the secure management of confidential data on a server, and the secure use of confidential data by means of different application clients. Through cryptographic encryption of the confidential data and the authorisation of all application clients by means of an authenticator client, the user retains full control over their confidential data available on the server at all times, while at the same time maintaining a high level of user-friendliness of the use of the confidential data. The method does not require a master password and comprises two-factor authentication.

(57) **Zusammenfassung:** Die vorliegende Erfindung betrifft ein computerimplementiertes Verfahren für ein gesichertes Management von geheimen Daten auf einem Server sowie die gesicherte Nutzung der geheimen Daten mittels verschiedener Applikationsclients. Der Nutzer behält durch kryptografische Verschlüsselung der geheimen Daten und die Autorisierung aller Applikationsclient mittels eines Authentifikatorclients jederzeit die volle Kontrolle über seine auf dem Server verfügbaren geheimen Daten bei gleichzeitig hoher Benutzerfreundlichkeit der Nutzung der geheimen Daten. Das Verfahren benötigt dabei kein Masterpasswort und umfasst Zwei-Faktor-Authentifizierung.

WO 2024/200764 A1

[Fortsetzung auf der nächsten Seite]

(84) Bestimmungsstaaten *(soweit nicht anders angegeben, für jede verfügbare regionale Schutzrechtsart)*: ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), eurasisches (AM, AZ, BY, KG, KZ, RU, TJ, TM), europäisches (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Veröffentlicht:

— *mit internationalem Recherchenbericht (Artikel 21 Absatz 3)*

— *in Schwarz-Weiss; die internationale Anmeldung enthielt in ihrer eingereichten Fassung Farbe oder Graustufen und kann von PATENTSCOPE heruntergeladen werden.*

INNOVATIVES SERVERBASIERTES VERFAHREN ZUM MANAGEMENT GEHEIMER DATEN

TECHNISCHES GEBIET

Die vorliegende Erfindung betrifft ein computerimplementiertes Verfahren für ein gesichertes Management geheimer Daten auf einem Server sowie die gesicherte Nutzung der geheimen Daten mittels verschiedener Applikationsclients. Der Nutzer behält durch kryptografische Verschlüsselung der geheimen Daten und die Autorisierung aller Applikationsclients mittels eines Authentifikatorclients jederzeit die volle Kontrolle über seine auf dem Server verfügbaren geheimen Daten bei gleichzeitig hoher Benutzerfreundlichkeit der Nutzung der geheimen Daten. Das Verfahren benötigt dabei kein Masterpasswort und umfasst Zwei-Faktor-Authentifizierung.
Die vorliegende Erfindung betrifft außerdem ein System, einen Server und drei Computerprogrammprodukte A, B und C zur Umsetzung des erfindungsgemäßen Verfahrens.

STAND DER TECHNIK

Die Nutzung von Nutzernamen und Passwörtern ist eine vielfach genutzte Methode der Authentifizierung eines Nutzers bei dessen Zugangsversuch zur Nutzung einer Webanwendung oder computerimplementierten Anwendung. Ein Passwort wird vom Nutzer zumindest bei dessen Initialisierung gesetzt und muss geheim gehalten werden. Wird das Passwort von einem potenziellen Angreifer unverschlüsselt angezeigt, abgefangen oder abgerufen, kann es erraten werden oder errechnet werden, sind Sicherheit und Integrität der Daten, Anwendungen und des Nutzerprofils gefährdet.

Gleichzeitig wächst die Zahl der Nutzeraccounts bei verschiedenen Anwendungen, die Nutzer mithilfe von Passwörtern zu sichern versuchen. Dadurch ergibt sich ein Zielkonflikt zwischen IT-Sicherheit und Nutzerfreundlichkeit. Werden alle Zugänge durch ein gleiches Passwort gesichert, ergibt sich ein „Single-Point-of-Failure“ und ein hoher Schweregrad des Problems, wenn dieses Passwort gebrochen wird. Werden Passwörter zu einfach und gut zu merken gewählt, können sie erraten werden. Sind sie zu kurz ergeben sich Chancen, mittels Brute-Force Methoden das Passwort zu erraten. Legt ein Nutzer seine Passwörter für erhöhte Nutzerfreundlichkeit zentral und unverschlüsselt ab, kann ein Angreifer dieses Verzeichnis kompromittieren.

Klassische Passwortmanager versuchen die Probleme der Nutzerfreundlichkeit durch ein Masterpasswort zu lösen. Alle Passwortdaten werden zentral gespeichert und der Zugang

mittels des Masterpassworts gewährt. Solche Systeme offenbaren dennoch klare Nachteile bei der IT-Sicherheit. Das Masterpasswort als Single-Point-of-Failure ist für Angreifer ein lukrativer Angriffsvektor. Bei Offline-Passwortmanagern kann es durch Brute-Force Methoden ermittelt werden, wenn das Masterpasswort nicht lang und stark genug ist. Studien offenbaren, dass Nutzer häufig nicht in der Lage sind, ein ausreichend langes und starkes Passwort zu generieren und sich dieses zu merken, ohne es wiederum im Klartext als Erinnerung zu offenbaren. Durch Key-Logging oder andere Malware kann außerdem das Masterpasswort ausgespäht oder der Nutzer durch Phishing oder ähnliche Methoden zur Eingabe seines Masterpassworts an der falschen Stelle angeleitet werden.

Auch von Seiten der Nutzerfreundlichkeit sind klassische Passwortmanager in der Regel suboptimal, da sie das regelmäßige Eingeben des, möglichst starken, Masterpasswortes voraussetzen.

Klassische Passwortmanager bieten in der Regel eine Möglichkeit der Zwei-Faktor-Authentifizierung (2FA) an, häufig vertrauen Nutzer aber nur auf ihr Passwort. 2FA bedeutet, dass die Authentifizierung zusätzlich über eine zweite Methode erfolgen muss, beispielsweise die Bestätigung eines Logins durch Eingabe eines SMS Code zusätzlich zum Passwort. Dabei können für den Nutzer vorteilhaft solche Faktoren, an die sich ein Nutzer erinnert (z.B. Passwort) mit solchen kombiniert werden, die ein Nutzer besitzt (Mobiltelefon, Biometrische Daten). 2FA ist für wichtige Zugänge grundsätzlich empfehlenswert und könnte zahlreiche Sicherheitslücken durch eine überlistete Authentifizierung im Ansatz verhindern.

Eine alternative Variante der Nutzung und Speicherung von Passwörtern sind cloudbasierte, zentrale Systemlösungen zur Passwortverwaltung, wie sie beispielsweise von Google oder Apple für die Browser von Clients angeboten werden. Dabei wird der Zugang zu den Passwörtern z.B. durch den bereits vorhandenen Account beim jeweiligen Anbieter abgesichert. Es ergeben sich nutzerfreundliche Funktionen wie die Synchronisation in alle Clientgeräte, auf denen ein Nutzer angemeldet ist. Weiterhin kann die serverbasierte Speicherung von Passwörtern relativ simpel Brute-Force Angriffe durch Kontrolle der Häufigkeit der Loginversuche verhindern.

Nichtsdestotrotz basieren solche Lösungen auf dem Zugang zu dem jeweiligen Nutzeraccount und damit über ein Masterpasswort und verlangen nicht zwingenderweise eine 2FA. Die Verschlüsselung der Passwortdaten auf dem Server wird je nach Verfahren nicht in jedem Fall durch die eigenen Clientgeräte kontrolliert. Wenn die Verschlüsselung der serverseitigen Passwortdaten bereits durch ein Clientgerät ausgeführt wurde, so ist nicht in garantiert, dass das Schlüsselmaterial zur Entschlüsselung lokal sicher, beispielweise auf einem Sicherheitschip, aufbewahrt wird. Ein zentrales Zugriffsmanagement, das

Passwortsharing oder die individuelle Verwaltung einzelner Geräte eines Nutzeraccounts und deren Zugriffsrechte sind in der Regel nicht gegeben.

Eine weitere Möglichkeit der Nutzung und Speicherung von Passwörtern sind „Single-Sign-On“ (SSO) Verfahren. SSO Verfahren bietet einem Nutzer die Möglichkeit des einfachen Zugriffs auf alle Dienste und Anwendungen seiner Berechtigung durch einmalige Authentifizierung. SSO Verfahren werden verbreitet als Software-as-a-Service (SaaS) Modell angeboten. Solche Passwortmanager, wie sie weiter oben beschrieben werden, können als ein Beispiel eines lokalen SSO Systems gesehen werden, bei dem ein lokaler Client, bei dem sich ein Nutzer authentifiziert, alle Zugangsdaten automatisiert ausfüllt. Darüber hinaus sind SSO Verfahren verbreitet, welche ein einzelnes Portal nutzen, welches wiederum den Login für andere Dienste und Anwendungen übernimmt, oder aber Ticketing Systeme, welche innerhalb eines vertrauenswürdigen Kreises von Diensten oder Anwendungen Nachweise der Authentifizierung eines Nutzers übertragen und somit bei allen Diensten und Anwendungen eine Anmeldung durchgeführt wird.

Ein Nachteil von SSO Verfahren ist die Unerschöpflichkeit der Dienste und Anwendungen, die vom Verfahren integriert werden müssen. So muss beispielweise bei Ticketing Systemen der Circle of Trust auf möglichst alle Dienste und Anwendungen erweitert werden, um ein großes Maß an Nutzerfreundlichkeit zu erreichen. Ein Nutzer kann sich nicht sicher sein, dass beispielsweise jede Webanwendung vom SSO Verfahren auch unterstützt wird. Insbesondere SSO Verfahren mit hoher Nutzerfreundlichkeit durch die Integration vieler Dienste und Anwendungen sind in der Regel teuer. Auch bei SSO Verfahren ist das Teilen von Zugangsdaten in der Regel nicht vorgesehen. Darüber hinaus sind SSO Verfahren je nach erster Authentifizierung nicht zwingend frei von einem Masterpasswort.

Passwörter sind nicht die einzigen Daten im digitalen Umfeld, die geheim zu halten sind. Kreditkartendaten werden beispielsweise vielfach im Onlineshopping eingesetzt. Sensible personenbezogene Daten wie Adressen und Kontaktdaten sollten ebenso geschützt werden. Grundsätzlich ist die Liste an potenziell zu schützenden, geheimen Daten von einem Nutzer erweiterbar. Denkbar sind hierbei im Wesentlichen beliebige Datentypen, insbesondere auch geheime Notizen oder Bilddaten.

Unabhängig vom Verfahren der Authentifizierung und der Verwaltung von Zugängen müssen geheime Daten, wie z.B. Passwörter, in der Regel verschlüsselt übertragen und gespeichert werden. Hierfür eignen sich kryptografische Verfahren. Solche Verfahren zielen dabei nicht nur auf die Verschlüsselung von Daten ab, sondern dienen darüber hinaus auch der Sicherstellung der Integrität und Herkunft von Daten.

Aus dem Stand der Technik ist beispielsweise ein Verfahren zur Sicherstellung der Integrität von Daten bekannt, bei dem aus Daten ein sogenannter Streuwert (auch Hash oder Prüfsumme genannt) berechnet wird. Dabei kommt eine sogenannte Streuwertfunktion (Hashfunktion) zum Einsatz. Eine solche Hashfunktion wird beispielsweise durch den Secure Hash Algorithmus (SHA) bereitgestellt. Mit einer Hashfunktion kann ein in seiner Größe nicht notwendigerweise beschränkter Datenblock auf einen Datenblock fester Größe, den Hash, abgebildet werden. Eine typische Länge für einen Hash ist beispielsweise 256 bit. Eine wünschenswerte Eigenschaft einer guten kryptografischen Hashfunktion ist die Injektivität und eine dadurch erzielte Kollisionsresistenz, was bedeutet, dass die Hashfunktion unterschiedliche Eingangsdaten immer auf unterschiedliche Hashes abbildet. Wenn der Sender der Daten ihren Hashwert berechnet und diesen Hashwert dem Empfänger zur Verfügung stellt, kann der Empfänger die Echtheit und Unverfälschtheit der Daten verifizieren, vorausgesetzt der Hashwert selbst ist echt und verlässlich.

Das Hash-Verfahren bringt jedoch den Nachteil mit sich, dass das Verfahren völlig außer Kraft gesetzt wird, wenn der Hashwert selbst während einer Man-In-The-Middle-Attacke manipuliert wird. Man-In-The-Middle Attacken lassen sich weitestgehend vermeiden, indem bei der Kommunikation zwischen A und B beide Seiten Zertifikate voneinander austauschen und kommunizieren, welche Zertifikate sie erwarten. Dadurch kann ausgeschlossen werden, dass ein Angreifer E die Kommunikation manipuliert. Dieser Vorgang wird als Certificate Pinning bezeichnet.

Ein Zertifikat ist ein Datensatz, der Eigenschaften einer Person oder eines Geräts bestätigt und dessen Herkunft und Integrität kryptografisch verifiziert werden kann. Weit verbreitet sind Zertifikate auf Basis von asymmetrischen kryptografischen Verfahren, auch public-key- oder public-private-key Verfahren genannt. Sie werden unter anderem zur digitalen Signatur eingesetzt. Bei asymmetrischen kryptografischen Verfahren besitzt ein Inhaber ein Schlüsselpaar aus einem öffentlichen und einem geheimen (oder privaten) Schlüssel. Durch eine eindeutige kryptografische Funktion berechnet der Inhaber ein Zertifikat bzw. eine Signatur zu einer bestimmten Nachricht bzw. deren eindeutigen Hash. Ein beliebiger Empfänger kann durch eine gegenseitige kryptografische Funktion aus dem vom Inhaber veröffentlichten öffentlichen Schlüssel und dem Zertifikat bzw. der Signatur die Nachricht oder deren Hash wiederherstellen und durch Vergleich mit der tatsächlichen (unverschlüsselten) Nachricht zweifelsfrei die Urheberschaft des Inhabers sowie die Integrität der Nachricht sicherstellen. Durch Versenden eines Zertifikates zu einer Kontaktaufnahme kann sich beispielsweise ein Gerät gegen eine Webanwendung bzw. einen Server authentifizieren, indem der Besitz des privaten Schlüssels verifiziert wird,

welcher wiederum z.B. durch eine vertrauenswürdige dritte Instanz an das Gerät ausgestellt wurde.

Asymmetrische kryptografische Verfahren können auch zur Verschlüsselung von Daten eingesetzt werden, sodass die originalen Daten im Klartext durch Dritte nicht eingesehen werden können. Dabei werden die Daten mittels des öffentlichen Schlüssels eines asymmetrischen Schlüsselpaars unter Nutzung kryptografischer Algorithmen durch einen Akteur im System verschlüsselt. Der geheime Schlüssel ist nur durch den Akteur bekannt, der die Daten entschlüsseln darf und sollte nicht geteilt, sondern sicher gespeichert werden. Der resultierende, verschlüsselte Datensatz kann nur vom zugehörigen geheimen Schlüssel entschlüsselt werden, also nur vom entsprechenden Akteur.

Bei der alternativen symmetrischen Verschlüsselung werden Daten dagegen mit einem einzelnen kryptografischen Schlüssel verschlüsselt. Sie kann nur unter Nutzung des gleichen Schlüssels entschlüsselt werden. Vorteilhaft an der symmetrischen Verschlüsselung ist die Tatsache, dass das Verfahren relativ simpel, effizient und schnell ist. Nachteilig an der symmetrischen Verschlüsselung ist, dass diese voraussetzt, dass alle vertrauenswürdigen Parteien, die berechtigt zum Einsehen der Daten sind, den Schlüssel kennen müssen, da sowohl das Verschlüsseln als auch das Entschlüsseln von Daten die Kenntnis des gleichen Schlüssels voraussetzt. Gleichzeitig muss gesichert sein, dass auch ausschließlich diese Parteien Kenntnis über diesen Schlüssel haben. Es ergibt sich dadurch das Problem des sogenannten Schlüsseltauschs über einen potenziell unsicheren digitalen Kommunikationskanal zur Initialisierung eines verschlüsselten Datenaustauschs zwischen mehreren Parteien.

Die Algorithmen bzw. Funktionen zur Erstellung von kryptografischen Schlüsseln aus beispielweise einer Zufallszahlenreihe müssen spezielle Anforderungen aufweisen. So muss nicht nur der geheime Schlüssel in der Lage sein, die mittels des öffentlichen Schlüssels verschlüsselte Nachricht zu entschlüsseln. Es darf außerdem nicht oder quasi nicht möglich sein, aus einem öffentlichen Schlüssel eines asymmetrischen Schlüsselpaars auf den geheimen Schlüssel zurückzuschließen. Um diese Voraussetzungen zu erfüllen, existieren verschiedene kryptografische Systeme mit verschiedenen Einwegfunktionen wie der zuvor beschriebene SHA. Mathematisch ist es nicht bewiesen, dass diese Einwegfunktionen bei asymmetrischen Verschlüsselungsverfahren nicht durch einen Angreifer mit entsprechend hoher Rechenleistung umkehrbar sind, man einen privaten Schlüssel also aus dem öffentlichen Schlüssel ableiten kann. Theoretisch bietet daher eine symmetrische Verschlüsselung wie beispielsweise das One-Time-Pad eine noch höhere Sicherheit als eine asymmetrische Verschlüsselung. Ein sich ständig weiterentwickelnder Stand der Technik sowie die Nutzung von viel beachteten Open-Source-Standards für die zugrundeliegenden

kryptografischen Systeme und Funktionen, wie beispielsweise Curve25519 für asymmetrische Verschlüsselung, sind ein Ansatz, ein hohes Maß an Sicherheit zu gewährleisten.

Transport Layer Security (TLS) ist ein kryptografisches Protokoll zur Absicherung von Kommunikation in Netzwerken. TLS ist ein Beispiel für einen vielbeachteten Standard, welcher in verschiedenen Versionen auf einem neuen Stand der Technik gehalten wird. TLS umfasst unter anderem Methoden für die Erstellung und Nutzung symmetrischer und asymmetrischer kryptografischer Schlüssel, zum Erstellen von digitalen Signaturen, zum Schlüsseltausch über unsichere Kanäle und zur Authentifikation.

Nachfolgend werden einige konkrete Ansätze aus dem Stand der Technik beschrieben, Verfahren für ein sicheres und nutzerfreundliches Management geheimer Daten, in der Regel bezogen auf Passwortdaten, bereitzustellen.

Aus US10893045B2 ist ein Verfahren bekannt, bei dem nur bestimmten Endgeräten der Zugriff auf Daten, unter anderem auf einem Server, erlaubt wird, indem mittels einer eindeutigen ID des Endgeräts auf dem Server eine Sitzung erstellt wird. Außerdem kann ein Authentifikator eingesetzt werden, um die Zugriffe von Endgeräten zu erlauben oder abzulehnen, indem der Nutzer mit einem zweiten Faktor authentifiziert wird. Allerdings sind auf dem Server nicht explizit Passwortdaten, sondern beliebige Daten gespeichert und der Zugang zu den Daten geschieht mittels eines Masterpasswortes. Die Verschlüsselungsmethode der serverseitigen Daten ist in dem Verfahren durch ein beliebiges Endgerät kontrolliert. Somit eignet sich das Verfahren als eine gesicherte Datenablage, aber keinen Passwortmanager ohne Masterpasswort.

Aus US9590959B2 ist weiterhin ein Verfahren bekannt, bei dem durch einen kryptografischen Service eine Zugriffsanfrage eines Nutzers auf einen Datenbankservice authentifiziert wird. Dabei werden unterschiedliche kryptografische Verfahren bereitgestellt, um die Authentizität der Anfrage bzw. des Nutzers zu beweisen. Implizit kann dieses Verfahren als Authentifikation für einen serverbasierten Passwortmanager mit zusätzlichen Funktionen verstanden werden. Es wird allerdings keine 2FA oder eine konkretere Verschlüsselungstechnik von potenziellen, in der Datenbank gespeicherten Passwortdaten offenbart.

Aus US8589442B2 ist ein Single-Sign-On System bekannt, welches nach einem dezentralen Prinzip funktioniert. Logindaten eines Nutzers auf einem Gerät können dabei auch auf anderen Geräten genutzt werden, indem eindeutige Geräte-IDs aller berechtigten Geräte mit einem Mapping-Algorithmus authentifiziert werden. In diesem Verfahren wird allerdings keine

2FA bei einzelnen Anmeldungen offenbart. US10341334B2 offenbart dagegen eine Single-Sign-On Lösung in einer Portalvariante, bei der ein Nutzer sich mittels eines Masterpassworts einmalig auf mehreren assoziierten Webseiten einloggt. Die zugehörigen Logins und Passwörter werden auf einem Webserver, mithilfe des Masterpassworts verschlüsselt, gespeichert. Eine 2FA wird auch hier nicht offenbart.

Aus US8904180B2 ist ein Verfahren und System bekannt, bei welchem verschlüsselte Daten auf einem Gerät separat von dem Schlüsselmaterial zur Entschlüsselung der Daten auf einem Server aufbewahrt werden. Der Zugang zum Schlüsselmaterial auf dem Server erfolgt durch Abfrage eines Client-Geräts mittels Sendens eines signierten Einmal-Schlüssels, wobei die Signatur vom Server oder einem dritten Dienst authentifiziert werden kann. Der Server stellt dann das Schlüsselmaterial, mit dem Einmal-Schlüssel verschlüsselt, an das Client-Gerät zur Verfügung. Durch die Nutzung zweiteiliger symmetrischer Schlüssel zur Ver- und Entschlüsselung der geheimen Daten, welche auf einem Clientgerät und zusätzlich auf der Serverseite gespeichert werden, wird eine Möglichkeit zur 2FA offenbart, welche implizit auch mit einem Smartphone als zweiten Faktor denkbar wäre. Allerdings offenbart das Patent keine Verfahren zur kontrollierten Authentifizierung anderer, vertrauenswürdiger Clients. Weiterhin ist die Verschlüsselung der Zugangsdaten auf dem Server nicht durch den Client oder ein zusätzliches Gerät gesteuert. Stattdessen werden die Zugangsdaten entweder mit einem zusätzlichen Masterkey auf Server verschlüsselt, oder aber als „Key-Bag“ auf dem Client, sodass der Server nur noch die Authentifizierung beisteuert. So entfallen entweder der Vorteil, die Sicherheit der Zugangsdaten gegen Angriffe auf den Server in der eigenen Hand zu haben, oder die Vorteile des Speicherns auf dem Server.

Aus US20080031447A1 ist ein Verfahren zur sicheren Speicherung von Passwörtern und Benutzernamen von Webseiten auf einem Server bekannt. Die Erfindung offenbart eine verschlüsselte Speicherung, wobei die Entschlüsselung der Daten nur durch Schlüsselmaterial eines Clients möglich ist, sodass die Daten auf dem Server nicht angreifbar sind. Allerdings offenbart das Verfahren keine 2FA und erwartet ein Master-Kennwort des Nutzers.

US20220209955A1 offenbart ein Verfahren zur Ermöglichung eines sicheren Loginprozesses auch unter Offlinebedingungen, wobei ein Server verschiedene Services, wie das regelmäßige Wechseln von Passwörtern und die Authentifizierung von Nutzern zur Verfügung stellt. Die Erfindung offenbart zwei Varianten der Speicherung von Passwortdaten: Online und offline. In der Onlinevariante wird zwar offenbart, dass sich ein Client mittels eines zusätzlichen Geräts als zweiten Faktors authentifizieren muss, aber es wird nicht eindeutig offenbart, dass die Passwortdaten auf dem Server durch von einem

Client kontrollierten Schlüsselmaterial dauerhaft verschlüsselt sind und somit nicht angegriffen werden können. Der zweite Faktor bzw. Authentifikator ist außerdem nicht in der Lage, im Sinne eines Passwortmanagers das Koppeln anderer Clients zu kontrollieren. In der Offlinevariante sind die Schlüssel dagegen auf dem Client verschlüsselt gelagert, nicht auf einem Server.

In US9727715B2 ist ein Verfahren offenbart, welches ebenfalls einen Server und eine Applikation zur Passwortverwaltung umfasst. Passwortdaten werden verschlüsselt mit einem zu einem Client zugehörigen asymmetrischen Schlüssel verschlüsselt, auf dem Server gelagert und können bei Besuch der zugehörigen Webseite abgerufen werden. Der private Schlüssel auf dem Client wird wiederum mittels eines weiteren Schlüssel verschlüsselt, welcher aus einem Authentifizierungsmechanismus generiert wird, sodass eine 2FA erreicht werden kann. Dieser zweite Faktor muss allerdings kein zusätzliches Gerät sein und ist nicht zur Verwaltung der verfügbaren Clients im Sinne eines Passwortmanagers eingerichtet. In dieser Erfindung ist der Client Besitzer des ersten Schlüssels, mehrere Clients können nicht durch einen vertrauenswürdigen zweiten Faktor, zum Beispiel eine mobile Applikation, verwaltet werden.

Aus US10491588B2 ist ein Verfahren bekannt, in der ein Server für das Passwortmanagement als Authentifikator agiert, um die Verbindung zwischen einem Gerät und einer sicheren Passwortdatenbank herzustellen. Das Verfahren umfasst auch das automatische Ausfüllen von Passwortformularen auf dem Gerät durch eine Webanwendung. Durch den Server als Schnittstelle muss allerdings immer sowohl eine Verbindung vom Server zum Gerät als auch der sicheren Passwortdatenbank hergestellt werden, wodurch die Nachteile von lokaler und cloudbasierter Speicherung kombiniert werden. Außerdem wird ein Masterpasswort benötigt.

Aus EP2332114B1 und US9948648B1 sind Verfahren zum automatischen Ausfüllen und Generieren von Passwortdaten in Webanwendungen bekannt. Die Verfahren spezifizieren allerdings keine sichere Kontrolle und Speicherung der Passwortdaten.

Aus EP3420677B1 ist weiterhin ein Verfahren bekannt, bei dem eine passwortfreie Koppelung von zwei Client Devices durch Kommunikation mittels eines Servers durchgeführt wird. Dabei wird auf verschlüsselte Weise unter Nutzung asymmetrischer Schlüsselpaare zwischen beiden Clients geheimes Material zur sicheren Authentifizierung ausgetauscht und auf einem Server hinterlegt, dass die Geräte gekoppelt sind. Der Austausch der Informationen über asymmetrische Verschlüsselung wird durch das Scannen von QR-Codes

von einem Client von dem anderen Client vereinfacht. Es wird ein Teilen von Logindaten wie Passwörtern zwischen den Clients offenbart, allerdings kein serverbasierter Passwortmanager, oder eine primäre Kontrolle der Passwortdaten durch einen Client.

Aus dem Security-Whitepaper v1.4 der heylogin GmbH ist ein Verfahren bekannt, welches eine passwortfreie Verwaltung geheimer Daten in nutzerfreundlicher Weise bereitstellt. Hierbei können geheime Daten sicher auf einem Server gespeichert und durch flexible Clientgeräte verschlüsselt werden. Das Verfahren offenbart allerdings keine Möglichkeit, nutzerbasiert die Berechtigungen zu den geheimen Daten zu verwalten, oder Zugänge auszutauschen, da jede Verschlüsselung von geheimen Daten fest einem Gerät zugewiesen ist.

AUFGABE

Es ist daher die Aufgabe der vorliegenden Erfindung, ein Verfahren zum sicheren Speichern, Nutzen und zum Management von geheimen Daten wie insbesondere Passwörtern bereitzustellen. Dabei soll der Nutzer keinen Zeit- oder Sicherheitsverlust durch das Merken und Eingeben geheimer Daten, insbesondere als Single-Point-of-Failure, in Kauf nehmen müssen. Stattdessen soll eine Benutzerfreundlichkeit ähnlich einer SSO-Lösung bereitgestellt werden. Dabei soll der Nutzer jedoch die geheimen Daten für jede beliebige Webanwendung und an jedem seiner genutzten Geräte einfach, effizient und kostengünstig verwalten können. Die Sicherheit und Integrität aller geheimen Daten auf im Internet exponierten Servern soll dabei sichergestellt und über die Geräte des Nutzers kontrolliert werden. Außerdem sollen die Berechtigungen zum Zugriff auf die geheimen Daten flexibel Nutzern zu- und abzuordnen sowie zwischen Nutzern teilbar sein.

LÖSUNG

Die Aufgabe wird durch ein Verfahren mit den Merkmalen des Anspruchs 1 gelöst.

Weitere vorteilhafte Ausgestaltungen und Weiterbildungen ergeben sich aus den Unteransprüchen sowie aus der Beschreibung unter Bezugnahme auf die Figuren.

ALLGEMEINE VORTEILE

Vorteilhaft kontrolliert ein Authentifikatorclient die Nutzung der geheimen Daten, die Geräte zur Nutzung dieser geheimen Daten und die zugrundeliegende Verschlüsselung aller geheimen Daten. Die geheimen Daten sind auf einem Server gespeichert, sodass mehrere Geräte auf diese zugreifen können, was einen Vorteil bezüglich der Benutzerfreundlichkeit

der Erfindung mit sich führt. Auf dem Server werden jegliche geheimen Daten ausschließlich verschlüsselt gespeichert, mit Schlüsseln, welche nur für auf den Geräten des Nutzers zur Verfügung stehen. Die Geräte senden niemals geheime Daten in unverschlüsselter Form zum Server, sodass der Server nur als Datenspeicher und Kommunikationsproxy verwendet wird. Der Server erlaubt jedoch einen zentralen Zugriff auf die geheimen Daten und deren Nutzung mit verschiedenen Geräten, solange der Nutzer dies mithilfe des Authentifikatorclients autorisiert. Jede Kommunikation zwischen den Geräten des Nutzers geschieht dabei über verschlüsselte Kanäle. Durch die Nutzung insbesondere eines Mobiltelefons mit Zugriffsschutz als Authentifikatorclient ist das Verfahren inhärent durch 2FA geschützt, wobei über den Authentifikatorclient gleichzeitig die geheimen Daten und Berechtigungen anderer Geräte einfach verwaltet werden können.

AUSFÜHRLICHE BESCHREIBUNG

Nachfolgend wird die Erfindung ausführlich beschrieben. Dabei wird die vorliegende Erfindung näher erläutert, ohne die Erfindung auf diese Beschreibungen zu beschränken. Sofern in dieser Anmeldung der Begriff "kann" verwendet wird, handelt es sich sowohl um die technische Möglichkeit als auch um die tatsächliche technische Umsetzung. Der Singular schließt den Plural ein, es sei denn, aus dem Kontext geht eindeutig etwas anderes hervor.

Insbesondere impliziert nachfolgend der Begriff Schlüsselpaar, dass es sich um ein asymmetrisches Schlüsselpaar, umfassend einen privaten und einen öffentlichen Schlüssel, handelt. Der Begriff Client umfasst nachfolgend sowohl einen Authentifikatorclient als auch einen Applikationsclient mit dessen Applikationsclienterweiterung.

Das erfindungsgemäße Verfahren basiert auf drei grundlegenden Schritten S01) bis S03). In Schritt S01) wird ein Push-Authentifikator auf einem initialen Authentifikatorclient initialisiert. Der Authentifikatorclient kann als Kernstück des erfindungsgemäßen Verfahrens auf der Nutzerseite gesehen werden, da mit diesem Authentifikatorclient in dem nachfolgend beschriebenen Verfahren jegliches Speichern und Nutzen, sowie jegliches Hinzufügen weiterer Clients gesteuert wird. Der Push-Authentifikator ist ein System von kryptografischen Schlüsseln des Authentifikatorclients, die es in ihrer Kombination erlauben, das nachfolgend beschriebene erfindungsgemäße Verfahren softwareseitig umzusetzen, während der Authentifikatorclient das hardwaretechnische Gerät an sich beschreibt, das für das erfindungsgemäße Verfahren genutzt wird. Der initiale Authentifikatorclient wird als initial beschrieben, weil er, z.B. bei Verlust des Geräts, ersetzt werden kann.

Zum Initialisieren des Push-Authentifikators auf dem Authentifikatorclient gehört in einem ersten Teilschritt von S01) das Generieren von zumindest zwei asymmetrischen

Schlüsselpaaren, einem Loginschlüsselpaar bestehend aus einem privaten Loginschlüssel und einem öffentlichen Loginschlüssel, sowie einem Codierungsschlüsselpaar bestehend aus einem privaten Codierungsschlüssel und einem öffentlichen Codierungsschlüssel. Für das Generieren und Speichern von asymmetrischen kryptografischen Schlüsseln wird bevorzugt ein zum Erstellen und Speichern kryptografischer Schlüssel designiertes Schlüsselelement verwendet. Bevorzugt wird dafür als Authentifikatorclient ein modernes Smartphone eingesetzt, bei denen solche Schlüsselelemente unter anderem für deren biometrische Funktionen zur Standardausstattung gehören. Solche Schlüsselelemente sind vorteilhaft dazu eingerichtet, besonders resilient gegen Cyberattacken und physische Attacken zu sein, sodass sie die privaten Schlüssel besonders gut schützen. Besonders bevorzugt wird zur Erstellung der asymmetrischen Schlüssel Curve25519 eingesetzt, welches im TLS-Protokoll 1.3 vorteilhaft empfohlen wird.

Die beiden öffentlichen Schlüssel werden in einem zweiten Teilschritt von S01) auf einem Server gespeichert. Durch Besitz des privaten Loginschlüssels kann sich ein Client gegen den Server authentifizieren. Der Loginschlüssel kann in diesem Verfahren als Login zu einem Account für das erfindungsgemäße Verfahren gesehen werden, allerdings ist dieser nicht abhängig vom Wissen um ein Masterpasswort, sondern von dem Besitz eines Geräts, des Authentifikatorclients. Für einen solchen Authentifikationsprozess wird bevorzugt das Erstellen einer Signatur verwendet für eine Nachricht, die mit der Kontaktaufnahme des Clients mit dem Server gesendet wird, mittels des privaten Loginschlüssels. Der Server kann durch Prüfen der Signatur mittels des öffentlichen Loginschlüssels verifizieren, dass die Nachricht tatsächlich von einem Client kommt, der in Besitz des privaten Loginschlüssels ist. Besonders bevorzugt kommt für den Authentifikationsprozess ein im TLS-Protokoll beschriebenes kryptografisches Verfahren zum Einsatz. Den öffentlichen Codierungsschlüssel kann der Server nutzen, um Daten so zu verschlüsseln, dass nur noch der Besitzer des privaten Codierungsschlüssels Zugriff erhält.

In Schritt S02) wird der nun vorhandene Push-Authentifikator genutzt, um weitere Clients mit dem Server zu koppeln, die die für diesen Account gespeicherten geheimen Daten abrufen können. Diese Clients werden nachfolgend Applikationsclients genannt.

Dieser Prozess des Koppelns eines solchen Applikationsclients wird in einem ersten Teilschritt von S02) durch den Authentifikatorclient autorisiert, indem dieser zumindest seinen privaten Loginschlüssel in verschlüsselter Form mit dem Applikationsclient austauscht. Diese verschlüsselte Übertragung wird bevorzugt mittels asymmetrischer Verschlüsselung durchgeführt. Eine bevorzugte Ausführungsvariante der verschlüsselten Übertragung wird im weiteren Verlauf der Beschreibung beschrieben.

Der Applikationsclient nutzt in einem zweiten Teilschritt von S02) den privaten Loginschlüssel, um sich in bereits beschriebener Form gegen den Server zu authentifizieren.

In einem dritten Teilschritt von S02) wird auf dem Server für den Applikationsclient eine Sitzung erstellt, indem der authentifizierte Applikationsclient den öffentlichen Sitzungsschlüssel eines durch den Applikationsclient generierten Sitzungsschlüsselpaars auf dem Server hinterlegt. Mit diesem öffentlichen Sitzungsschlüssel können Daten verschlüsselt werden, auf die nur dieser Applikationsclient mittels seines privaten Sitzungsschlüssels Zugriff hat. Das Erstellen einer Sitzung setzt voraus, dass der Applikationsclient ein Computerprogrammprodukt umfasst, welches für das Generieren eines asymmetrischen kryptografischen Sitzungsschlüsselpaars eingerichtet ist. Ein Computerprogrammprodukt, welches unter anderem diese Funktionalität bietet, ist ein Teil dieser Erfindung und wird im weiteren Verlauf der Beschreibung beschrieben. In einer bevorzugten Variante wird eine Softwarefunktionalität zur Erstellung von kryptografischen Schlüsseln auf dem Applikationsclient durch eine im weiteren Verlauf beschriebene Applikationsclienterweiterung bereitgestellt. In einer besonders bevorzugten Variante nutzt der Applikationsclient ebenfalls ein Schlüsselelement zum sicheren Speichern der privaten Schlüssel, hier insbesondere der private Codierungsschlüssel und der private Sitzungsschlüssel, über die er verfügt.

Der Schritt S02) kann wiederholt werden, um mehrere Applikationsclients mithilfe des Authentifikatorclients zu koppeln. In einer bevorzugten Ausführung des Verfahrens werden alle gekoppelten Clients bzw. deren Sitzungen auf dem Authentifikatorclient in Listenform angezeigt. In einer besonders bevorzugten Ausführung ist das Verfahren für eine Option des Authentifikatorclients eingerichtet, die Sitzungen von Applikationsclients zu kontrollieren, die öffentlichen Sitzungsschlüssel vom Server zu Löschen und damit die Sitzung serverseitig zu entfernen. In dieser Variante wird jegliche Kontaktaufnahme des Applikationsclients ohne Sitzung mit dem Server vom Server abgelehnt. In einer besonders bevorzugten Variante ist der Applikationsclient dazu eingerichtet, in diesem Fall der Ablehnung durch den Server, jegliche Schlüssel, die er besitzt, zu löschen. Dadurch wird vorteilhaft das langfristige und dadurch unnötig riskante Speichern von privaten Schlüsseln auf nicht autorisierten Geräten verhindert. In einer weiterhin bevorzugten Variante ruft der Applikationsclient den Status seiner Sitzung in regelmäßigem zeitlichem Abstand vom Server ab, sodass vorteilhaft spätestens zu diesem Zeitpunkt das private Schlüsselmaterial vom Authentifikatorclient gelöscht wird, wenn keine Sitzung mehr existiert. In einer besonders bevorzugten Ausführungsvariante wird der Status der Sitzung vom Applikationsclient unmittelbar beim Start der Applikationsclienterweiterung erstmalig abgerufen. In einer weiteren besonders bevorzugten Variante kann das zeitliche Intervall eingestellt werden und wird außerdem abhängig gemacht von den Sicherheitsvoraussetzungen auf dem Applikationsclient. Ist

beispielsweise kein Schlüsselelement im Applikationsclient verbaut und der Applikationsclient ist mit einem potenziell gefährlichen öffentlichen Netzwerk verbunden, so kann das zeitliche Intervall zum Abrufen des Sitzungsstatus auf eine besonders geringe Zeit gestellt werden, beispielsweise 1 Sekunde. In einer außerdem bevorzugten Variante kann ein Client nicht sehen, dass seine eigene Sitzung nicht mehr existiert, ohne Kontakt zum Server aufzunehmen, wodurch das Schlüsselmaterial gelöscht wird. Dadurch wird vorteilhaft verhindert, dass ein Angreifer überhaupt incentiviert wird, das Löschen der Sitzung hinauszögern zu wollen. In einer weiterhin bevorzugten Ausführungsvariante erhalten auch Authentifikatorclients jeweils eine Sitzung auf dem Server, sodass vorteilhaft mehrere Authentifikatorclients eines Nutzers initialisiert, aber auch von einem anderen Authentifikatorclient aus kontrolliert werden können.

In Schritt S03) werden der initialisierte Authentifikatorclient sowie der gekoppelte Applikationsclient zum Speichern bzw. Abrufen von geheimen Daten durch den Applikationsclient auf bzw. von einem auf dem Server befindlichen Tresor genutzt. Ein Tresor ist ein Datenpaket auf dem Speichermedium des Servers, welcher einem Account zugeordnet ist und, wie nachfolgend beschrieben, kryptografisch verschlüsselt ist. In einer bevorzugten Ausführung enthält der Tresor eine geordnete Liste an Übergaben, welche die zeitliche Reihenfolge der gespeicherten geheimen Daten abbilden, sodass alle geheimen Daten eindeutig sortiert und zugeordnet sind. Grundsätzlich können mehrere Tresore für einen Account existieren, die verschieden verschlüsselt werden können.

Geheime Daten im Tresor umfassen bevorzugt Logindaten für Webanwendungen, das heißt Benutzernamen, Mailadressen und Passwörter. Darüber hinaus können auch alternative geheime Daten erfindungsgemäß in Tresoren gespeichert werden, beispielsweise Zahlungs- und Kreditkarteninformationen oder Adressen. In einer weiteren Ausführung des Tresors können auch vom Nutzer definierbare, andere Datentypen wie freie Texte für geheime Notizen oder Bilddaten gespeichert werden.

In einem ersten Teilschritt zum Zweck des Speicherns und Abrufens von geheimen Daten in Schritt S03) wird der Tresor, welcher geheime Daten beinhaltet, mittels eines symmetrischen Schlüssels verschlüsselt. In dem Fall, dass mehrere Push-Authentifikatoren Zugriff auf den Tresor erhalten sollen, müssen durch diese Verschlüsselung vorteilhaft nicht mehrere Kopien des Tresors für die unterschiedlichen Push-Authentifikatoren verschlüsselt werden. Es existiert daher sinnvollerweise immer genau ein symmetrischer Schlüssel zu einem Tresor.

Da bei einer symmetrischen Verschlüsselung das Entschlüsseln ebenfalls unter Nutzung des gleichen symmetrischen Schlüssels geschieht, muss dieser Schlüssel ebenfalls verschlüsselt werden, da er auf dem Server gespeichert werden soll, um autorisierten Zugriff auf den

Tresor zu erlauben. Dieses Verschlüsseln geschieht in einem zweiten Teilschritt von S03) mittels des öffentlichen Codierungsschlüssels des Push-Authentifikators, sodass vorteilhaft nur ein Client Zugriff auf den symmetrischen Schlüssel und damit den Tresor erhält, der in Besitz des privaten Codierungsschlüssels ist.

In einem dritten Teilschritt von S03) wird das Speichern und Abrufen von geheimen Daten aus dem Tresor durch den Applikationsclient autorisiert, und zwar durch den Authentifikatorclient, welcher in verschlüsselter Form den privaten Codierungsschlüssel für den Applikationsclient bereitstellt, damit dieser den privaten Codierungsschlüssel abrufen und den Tresor entschlüsseln kann. Bevorzugt wird dieses verschlüsselte Bereitstellen des privaten Codierungsschlüssels durch eine aktive Eingabe am Authentifikatorclient durch einen Nutzer autorisiert. In einer möglichen Ausführung wird das verschlüsselte Bereitstellen bzw. Abrufen dadurch realisiert, dass der private Codierungsschlüssel vom Authentifikatorclient mittels des öffentlichen Sitzungsschlüssel verschlüsselt und über den Server an den Applikationsclient gesendet wird, wobei nur der Applikationsclient diese Sendung entschlüsseln kann. In einer weiteren Variante wird der private Codierungsschlüssel gemeinsam mit dem privaten Loginschlüssel in Schritt S02) an den Applikationsclient übertragen. In einer bevorzugten Ausgestaltung der beider Übertragungsvarianten kann der private Codierungsschlüssel nicht durch den Applikationsclient in beständigem Speicher gespeichert werden, sodass der Authentifikatorclient jedes einzelne Speichern oder Abrufen bestätigen muss und der private Codierungsschlüssel bei einem physischen Angriff auf den Applikationsclient, während dieser abgeschaltet ist, vorteilhaft nicht gestohlen werden kann. Weitere bevorzugte Ausführungen zum Autorisieren des Speicherns und Abrufens durch Bereitstellen des Codierungsschlüssels durch den Authentifikatorclient ergeben sich aus weiteren beschriebenen bevorzugten Merkmalen der Erfindung.

Vorteilhaft kontrolliert der Authentifikatorclient durch diese Konfiguration insbesondere in Schritt S02) die autorisierten Clients und in Schritt S03) jeden Zugriff auf die geheimen Daten. Weiterhin vorteilhaft wird hier für einen Abrufversuch von geheimen Daten vom Applikationsclient der Authentifikatorclient als zweiter Faktor eingesetzt. In einer bevorzugten Ausführungsvariante wird hierfür auf dem Authentifikatorclient das aktivierte Nutzen einer Schutzfunktion des Geräts abverlangt, zum Beispiel durch eine biometrische Authentifizierung. Dadurch wird vorteilhaft der 2FA-Schutz vollständig durch den Besitz eines Gerätes durch den Nutzer (Authentifikatorclient) und eine Abfrage eines Geheimnisses des Nutzers (z.B. biometrische Authentifizierung). Diese Authentifizierung des Nutzers am Authentifikatorclient ist, insbesondere bei Nutzung eines modernen Smartphones, heutzutage in jedem Fall Standard und unabhängig vom Verfahren zum Management

geheimer Daten empfehlenswert. Insofern ist diese Maßnahme vorteilhaft nicht als zusätzlicher Aufwand zu sehen, wie andere Methoden zur 2FA in der Regel gesehen werden.

Wie bereits beschrieben wird in einer bevorzugten Ausführung der Erfindung ein Smartphone als Authentifikatorclient eingesetzt, welches über ein Schlüsselelement verfügt. Weiterhin bevorzugt verfügt der Authentifikatorclient über Möglichkeiten der Benutzereingabe zum Autorisieren der Schritte S02) und S03).
In einer speziellen Ausführung wird ein Authentifikatorclient gleichzeitig als Applikationsclient eingesetzt. In diesem Fall werden die Computerprogrammprodukte, die bei deren Ausführung jeweils die Merkmale des Verfahrens auf dem Authentifikatorclient bzw. dem Applikationsclient implementieren, getrennt voneinander installiert und ausgeführt. So werden die Merkmale des Verfahrens, die zu dem Authentifikatorclient gehören, beispielsweise in einer mobilen App ausgeführt und die Merkmale des Verfahrens, die zum dem Applikationsclient gehören in einer Webanwendung, die über den Browser aufgerufen wird. Besonders bevorzugt wird in diesem Fall die Ausführungsvariante, bei der die Nutzung des Push-Authentifikators innerhalb der mobilen App eine Schutzfunktion des Smartphones beim Öffnen der mobilen App voraussetzt, sodass vorteilhaft eine 2FA bei Nutzung des Gerätes gesichert wird.

Das erfindungsgemäße Verfahren umfasst weiterhin den Einsatz zumindest eines Profils als Indirektionsebene des Authentifikatorclients, zusätzlich zu dem initialisierten Push Authentifikator. Ein Profil ist im Wesentlichen technisch äquivalent zu einem Push-Authentifikator zu betrachten, mit dem Unterschied, dass ein Profil keinen eigenen Loginschlüssel besitzt, um sich gegen den Server zu authentifizieren. Stattdessen verschlüsselt ein Profil in einer standardmäßigen Ausführungsform anstelle zumindest eines Push-Authentifikators den symmetrischen Schlüssel zumindest eines Tresors mit einem öffentlichen Profilschlüssel eines auf dem Authentifikatorclient erstellten asymmetrischen Profilschlüsselpaars. Das Profil und damit insbesondere der private Profilschlüssel, wird dann in dieser standardmäßigen Ausführungsform serverseitig durch den öffentlichen Codierungsschlüssel des zugehörigen Push-Authentifikators verschlüsselt. Ein Profil umfasst zumindest das Profilschlüsselpaar, im weiteren Verlauf der Beschreibung werden allerdings noch weitere zum Profil gehörige Schlüssel beschrieben, welche entsprechend ebenfalls durch den öffentlichen Codierungsschlüssel des zugehörigen Push-Authentifikators verschlüsselt werden.

Durch diese Konfiguration an standardmäßigen und bevorzugten Verschlüsselungen ist der Begriff des Profils als Indirektionsebene erklärt, der bedeutet, dass das Profil im Verschlüsselungsprozess eines Tresors als zusätzliche Ebene zwischen einen Push-

Authentifikator und einen Tresor geschaltet ist. Das Profil wird insofern durch diesen Push-Authentifikator und damit durch den Authentifikatorclient kontrolliert. Gleichzeitig kontrolliert das Profil den Zugriff auf mit seinem Profilschlüsselpaar verschlüsselte Tresore. Der technische Effekt ist eine zusätzliche Verschlüsselungsebene, mit der sich eine große Bandbreite an Verschlüsselungsvarianten zwischen Push-Authentifikatoren und Tresoren umsetzen lassen.

Durch die große Bandbreite an Verschlüsselungsvarianten ergeben sich eine Reihe von Vorteilen bezüglich der Flexibilität des gesamten erfindungsgemäßen Verfahrens. Einerseits ergibt sich ein Vorteil bezüglich der Aufgeräumtheit und Übersichtlichkeit des Verschlüsselungsverfahrens, welche auch als „Schlüsselhygiene“ bezeichnet werden kann. Profile erlauben es, für getrennte Kontexte getrennte Schlüssel in Profilen anzulegen, beispielsweise in Profilen für private Nutzung und für Unternehmensnutzung. Profile sind dabei jeweils völlig unabhängig voneinander zu betrachten, da keine Abhängigkeiten zwischen zufälligen Profilstartwerten bzw. Profilschlüsseln von Profilen bzw. zwischen Profilen und Push-Authentifikatoren bestehen. Weiter gedacht und konkreter erlaubt diese Schlüsselhygiene vorteilhaft die Schaffung inhaltlicher sinnvoller Rollen und Zuordnungen innerhalb des erfindungsgemäßen Systems.

Beispielsweise wird in einer bevorzugten Ausführungsform einem Nutzer pro Organisation oder Kontext ein Profil zugeordnet, in dem alle Zugänge für diesen Nutzer gesammelt werden können. Profile erlauben vorteilhaft, Berechtigungen nutzerbezogen oder anderweitig flexibel umzusetzen. Weiterhin erlauben Profile beispielsweise das Umsetzen eines Superadmin-Profils, welches alle Tresore einer Organisation verschlüsseln kann und somit eine Grundvoraussetzung für die etwaige Vergabe von Administratorrechten zur Verwaltung der geheimen Daten, aber auch zum Verhindern des Verlustes von geheimen Daten durch Schlüsselverlust ist.

Grundsätzlich vorteilhaft ist der Einsatz von Profilen weiterhin in Bezug auf die Kryptoagilität der Erfindung beziehungsweise der Organisation, welche das erfindungsgemäße Verfahren nutzt. So können beispielsweise Kryptoverfahren schrittweise umgestellt bzw. aktualisiert werden. Somit kann bei geringem Aufwand eine Grundsicherheit gegen bestimmte Bedrohungen hergestellt werden, ohne das beispielsweise jeder einzelne Tresor umgeschlüsselt wird. Dies ist ein Flexibilitätsvorteil gegenüber dem Einsatz von ausschließlich direkten Verschlüsselungen von Tresoren durch Push-Authentifikatoren.

In einer bevorzugten Ausführungsform wird jegliches zu einem Profil zugehörige Schlüsselmaterial aus einem zufälligen Profilstartwert erstellt. In dieser bevorzugten Ausführungsform bedeutet das Verschlüsseln eines Profil, z.B. durch den

Codierungsschlüssel des zugehörigen Push-Authentifikators, dass der zufällige Profilstartwert verschlüsselt wird.

In einer Ausführungsform werden jedem Nutzer pro Organisation oder Kontext nicht nur genau ein Profil zum Verschlüsseln von Tresoren zugeordnet, sondern zumindest ein weiteres Profil, welches dem vollverschlüsselten Empfangen von Nachrichten und Informationen und/oder dem Speichern von Präferenzen und Einstellungen bezüglich der Nutzung des erfindungsgemäßen Verfahrens, beispielsweise die Zeit bis zum automatischen Schließen einer Sitzung, dient. Vorteilhaft können somit in der gleichen Verschlüsselungsarchitektur weitere Funktionen neben dem Speichern und Abrufen geheimer Daten umgesetzt werden.

In einer Ausführungsvariante eines Profils werden mehrere gleiche Kopien des Profils serverseitig durch die öffentlichen Codierungsschlüssel mehrerer Push-Authentifikatoren verschlüsselt. Durch diese Ausführungsvariante erhalten mehrere Push-Authentifikatoren, insbesondere von mehreren Nutzern, Zugriff auf ein Profil. Vorteilhaft ist hierbei eine erhöhte Flexibilität bezüglich des Verwaltens und Veränderns von Authentifikatoren. Ein Profil kann durch beliebige Authentifkatoren verschlüsselt werden. Beispielsweise können Hardware Security Devices als zusätzliche bzw. Backup-Authentifikatoren hinzugefügt oder entfernt werden. Das gesamte Set der Authentifikatoren eines Nutzers oder mehrerer Nutzer kann so flexibel bezüglich seines Profils rotiert werden. Dies ist weiterhin vorteilhaft für eine sicherere Backup und Recovery Strategie, da es den Totalverlust von Schlüsselmaterial unwahrscheinlicher macht.

In einer weiteren Ausführungsvariante eines Profils werden mit dem gleichen öffentlichen Codierungsschlüssel eines Push-Authentifikators mehrere Profile verschlüsselt, sodass ein Push-Authentifikator Zugriff auf mehrere Profile erhält. Vorteilhaft kann durch diese Ausführung ein Push-Authentifikator als Master-Gerät für spezifische Anwendungen umgesetzt werden, in denen ein umfänglicher Zugriff nicht von Nutzern oder Profilen, sondern an bestimmten Geräten abhängen soll.

In einer weiteren Ausführungsvariante wird ein Profil durch die öffentlichen Codierungsschlüssel mehrerer Push-Authentifikatoren gleichzeitig verschlüsselt, sodass diese mehreren Push-Authentifikatoren gleichzeitig benötigt werden, um Zugriff auf das Profil zu erhalten. Diese Variante wird bevorzugt nur für besonders sensible geheime Daten eingesetzt, für welche ein hohes Sicherheitsmaß gilt oder für deren Verwaltung die Zustimmung mehrerer Nutzer nötig ist. Vorteilhaft kann so ein digitales Vier-Augen-Prinzip für den Zugriff auf dieses Profil hergestellt werden.

In einer durch die Einführung von Profilen folgenden, weiterhin bevorzugten Ausführungsvariante wird der symmetrische Schlüssel eines Tresors durch einen oder mehrere verschiedene öffentliche Profilschlüssel oder öffentliche Codierungsschlüssel von mehreren Profilen oder Push-Authentifikatoren verschlüsselt. Dabei kann der symmetrische Schlüssel durch die öffentlichen Schlüssel mehrerer Profile, mehrerer Push-Authentifikatoren oder sowohl Profile als auch Push-Authentifikatoren verschlüsselt werden.

Ein symmetrischer Schlüssel kann in einer Ausführungsform durch die öffentlichen Schlüssel mehrerer Profile oder Push-Authentifikatoren gleichzeitig verschlüsselt werden, sodass diese mehreren Profile oder Push-Authentifikatoren benötigt werden, um Zugriff auf den symmetrischen Schlüssel zu erhalten. Diese Variante wird bevorzugt nur für besonders sensible geheime Daten eingesetzt, für welche ein hohes Sicherheitsmaß gilt oder für deren Verwaltung die Zustimmung mehrerer Nutzer nötig ist. Vorteilhaft kann so ein digitales Vier-Augen-Prinzip für den Zugriff auf geheime Daten hergestellt werden.

In einer alternativen, bevorzugten Ausführungsform können mehrere Kopien des gleichen symmetrischen Schlüssels durch die öffentlichen Schlüssel mehrerer Profile oder Push-Authentifikatoren verschlüsselt werden, sodass diese mehreren Profile oder Push-Authentifikatoren unabhängig voneinander Zugriff auf den symmetrischen Schlüssel erhalten. Vorteilhaft kann so der Zugriff der Profile bzw. Push-Authentifikatoren auf einen Tresor unabhängig voneinander hergestellt und verwaltet werden.

Wie bereits beschrieben, können mehrere Tresore für einen Account erstellt werden, sodass im Umkehrschluss auch ein öffentlicher Profilschlüssel eines Profils oder ein öffentlicher Codierungsschlüssel eines Push-Authentifikators mehrere verschiedene, zu verschiedenen Tresoren gehörige, symmetrische Schlüssel verschlüsseln kann.

Durch diese möglichen Ausführungsvarianten kann beispielsweise, aber nicht bevorzugt, ein einzelner Push-Authentifikator sowie zwei Profile Zugriff auf einen symmetrischen Schlüssel erhalten, wobei wiederum jeweils zwei verschiedene, weitere Push-Authentifikatoren Zugriff auf jedes Profil haben. Damit hätten insgesamt 5 verschiedene Push-Authentifikatoren Zugriff auf den zugehörigen Tresor. In einer weiteren möglichen, aber nicht besonders bevorzugten Ausführungsvariante kann beispielsweise ein Push-Authentifikator direkten Zugriff auf den symmetrischen Schlüssel eines Tresors, und auf ein Profil haben, wobei das Profil weiterhin Zugriff auf drei weitere Tresore hat. Damit hätte ein Push-Authentifikator Zugriff auf insgesamt 4 Tresore.

Profile sind als Indirektionsebene für Push-Authentifikatoren vorteilhaft, da sie Flexibilität für die Zugriffsberechtigungen auf geheimen Daten mit sich bringen. Der Zugriff auf einen

Tresor ist nichtmehr zwangsweise an einen Nutzer und dessen (initialen) Authentifikatorclient gebunden. Stattdessen kann ein Nutzer in einer bevorzugten Ausführungsvariante den Zugriff auf einen Tresor über ein Profil mit anderen Push-Authentifikatoren auf anderen Authentifikatorclients anderer Nutzer teilen. Zum Teilen des Zugriffs auf einen Tresor über ein Profil verschlüsselt ein teilender Authentifikatorclient in einer bevorzugten Ausführung dieses Profil mit dem öffentlichen Codierungsschlüssel eines beim Server registrierten Push-Authentifikators eines anderen, empfangenden Authentifikatorclients, indem er den öffentlichen Codierungsschlüssel über den Server abruft. Danach speichert der teilende Authentifikatorclient das verschlüsselte Profil auf dem Server, sodass der empfangende Authentifikatorclient das nur für ihn lesbare Profil abrufen kann. Profile bieten damit eine Lösung für sicheres, vollständig verschlüsseltes Account-Sharing, wobei der Zugriff auf die geheimen Daten, nicht aber der Zugriff auf den Account über den Push-Authentifikator selbst geteilt wird. Diese Funktion ist vorteilhaft insbesondere in kleinen Teams mit spezialisierten Aufgaben im Arbeitsumfeld.

In einer bevorzugten Ausführungsform werden zwei Profile sequenziell eingesetzt, wobei zumindest ein erstes Profil direkt durch den öffentlichen Profilschlüssel des Profilschlüsselpaars zumindest eines zweiten Profils verschlüsselt wird. Durch die Ausführungsform zweier sequenzieller Profile können vorteilhaft bestimmte Berechtigungsverkettungen noch einfacher und aus Sicht der Schlüsselhygiene noch sauberer umgesetzt werden. Beispielsweise kann ein Nutzer temporär den Zugriff auf geheime Daten oder Zugangsdaten von einem anderen Nutzer übernehmen, ohne dass die dem anderen Nutzer zugeordneten Profile oder Authentifikatoren in sich geändert werden. Stattdessen muss nur eine Verschlüsselung eines Profils des einen Nutzers mit einem Profil des anderen Nutzers erstellt werden und an das Profil des einen Nutzer gesendet werden. So kann beispielsweise ein Erziehungsberechtigter temporär den Zugriff auf die Zugangsdaten eines Kindes übernehmen. In einem weiteren Anwendungsbeispiel können Zugangsdaten eines eine Organisation verlassenden Mitarbeiters sicher und einfach an einen neuen Mitarbeiter übergeben werden.

Die direkte Verschlüsselung zumindest eines ersten Profils durch den öffentlichen Profilschlüssel des Profilschlüsselpaars zumindest eines zweiten Profils bedeutet, dass keine weiteren Zwischenschritte eingeführt werden, wie beispielsweise eigene Tresore für die Aufbewahrung bestimmter Profile. Diese Ausführungsform ist weniger rechen- und speicherintensiv. Vorteilhaft wird die Verschlüsselung der Profile somit effizienter und aus Sicht der Schlüsselhygiene simpler und übersichtlicher.

In einer besonders bevorzugten Ausführungsform des sequenziellen Einsetzens zweier Profile wird der zufällige Profilstartwert des ersten Profils, aus dem dessen Schlüsselmaterial

generiert werden kann, direkt durch das Profilschlüsselpaar des zweiten Profils verschlüsselt. Der Profilstartwert umfasst alle notwendigen Informationen des Profils, sodass vorteilhaft nicht mehrere einzelne Schlüssel verschlüsselt werden müssen.

In einer besonders bevorzugten Ausführungsform werden zwei Profile dergestalt sequenziell eingesetzt, dass vorteilhaft eine umfängliche Berechtigungsstruktur für zumindest einen Administrator der geheimen Daten einer Organisation kryptografisch umgesetzt werden. Dabei verschlüsselt das erste Profil sämtliche zu einer Organisation gehörigen Tresore und das zumindest zweite Profil wird von den Push-Authentifikatoren eines Administrators der Organisation verwaltet. Das erste Profil nimmt dabei technisch die Funktion eines Superadministrators ein und kann vorteilhaft ungewollte Zugriffsverluste auf Tresore verhindern, indem, bevorzugt automatisiert, jeder Tresor durch dieses Profil automatisiert verschlüsselt wird. Auf dieses Profil haben wiederum zumindest ein, aber bevorzugt alle Nutzerprofile von Administratoren der Organisation Zugriff. Soll eine Administratorberechtigung aufgehoben werden, kann vorteilhaft simpel die Verschlüsselung des ersten Profils mit dem zweiten Profil dieses Administrators gelöscht werden, sodass das zweite Profil nicht mehr in der Lage ist, das erste Profil zu entschlüsseln.

In einer besonders bevorzugten Ausführungsform des sequenziellen Einsetzens zweier Profile zum Umsetzen einer Berechtigungsstruktur für zumindest einen Administrator der geheimen Daten einer Organisation ist der Administrator der Organisation in der Lage, sämtliche Verschlüsselungen entsprechend Verfahrensschritt S03) seiner Organisation zu löschen, durch Platzhalter zu ersetzen sowie neu zu generieren. Dieses Ausführungsform erlaubt es einem Administrator, Profile bestimmter Nutzer, bestimmte Push-Authentifikatoren oder bestimmte Verschlüsselungen (und damit Zugriffsberechtigungen) aufzuheben. Vorteilhafte Nutzungsfälle hiervon sind das On- und Offboarding von Mitarbeitern einer Organisation, das Erstellen und Aufheben bestimmter (Administrator-)Berechtigungen, sowie das Sperren und Wiederherstellen des Zugriffs beim Verlust oder Wechsel von Geräten, die als Push-Authentifikator eingesetzt werden können.

In einer ganz besonders bevorzugten Ausführungsform ist der Administrator der Organisation in der Lage, das Auflösen einer Verschlüsselung und damit der entsprechenden Zugriffsberechtigung wie folgt durchzuführen. Die Verschlüsselung eines Profils oder des symmetrischen Schlüssels eines Tresors wird gelöscht. Im Falle der Verschlüsselung eines Profils wird das Profil als freies Profil weitergeführt, wobei Platzhalter als Verschlüsselungen eingesetzt werden. Bei Rückkehr oder Übergabe des Profils werden neue Verschlüsselungen erstellt und die Schlüssel oder der Profilstartwert an den Push-Authentifikator des neuen, ersetzten oder zurückgekehrten Nutzers verschlüsselt und gesendet. Vorteilhaft kann hierdurch ein flexibler und sicherer Onboardingprozess gestartet

werden. Besonders bevorzugt generiert ein neuer Push-Authentifikator die Schlüssel seines neu zugewiesenen Profils nach dem Onboarding neu, sodass datenschutztechnisch vorteilhaft nur er, und nicht mehr der Administrator, Zugriff darauf hat.

In einer bevorzugten Ausführungsvariante des erfindungsgemäßen Verfahrens umfasst ein Tresor einen sensiblen Datenbereich und eine Metadatenbereich. Der sensible Datenbereich ist innerhalb des Tresors zusätzlich durch einen symmetrischen Hochsicherheitsschlüssel verschlüsselt. Das bedeutet, dass zum Zugriff auf diesen sensiblen Datenbereich sowohl der symmetrische Schlüssel zum Entschlüsseln des gesamten Tresors als auch der symmetrische Hochsicherheitsschlüssel zum Entschlüsseln des sensiblen Datenbereichs benötigt wird. Der symmetrische Hochsicherheitsschlüssel wird wiederum mit einem öffentlichen Hochsicherheitsschlüssels eines asymmetrischen Hochsicherheitsschlüsselpaars eines Push-Authentifikators oder eines öffentlichen Profilhochsicherheitsschlüssels eines asymmetrischen Profilhochsicherheitsschlüsselpaars eines Profils verschlüsselt.

In einer bevorzugten Ausführung entsprechend der Nutzung des erfindungsgemäßen Verfahrens als Passwortmanager, können in dem Metadatenbereich geheime, aber nicht hochsensible Daten wie Benutzernamen und Mailadressen gespeichert werden, während im sensiblen Datenbereich die zugehörigen Passwörter und weiteren geheimen Daten wie Antworten auf Sicherheitsfragen gespeichert werden. Bei einer alternativen Anwendung des erfindungsgemäßen Verfahrens für den Onlinehandel könnten im Metadatenbereich beispielsweise Rechnungsadressen und im sensiblen Datenbereich Kreditkarteninformationen aufbewahrt werden.

Durch den Zugriff oder fehlenden Zugriff auf den privaten Hochsicherheitsschlüssels bzw. den privaten Profilhochsicherheitsschlüssels kann somit stufenweise der Zugriff auf einen zweiten Bereich mit sensibleren Daten bestimmt werden. Das Hochsicherheitsschlüsselpaar bzw. das Profilhochsicherheitsschlüsselpaar gehört jeweils zu einem Push-Authentifikator bzw. Profil und wird entsprechend neben den bereits beschriebenen zu einem Push-Authentifikator oder Profil gehörigen Schlüsselpaaren auf dem entsprechenden Authentifikatorclient erzeugt und der jeweilige private Schlüssel dort gespeichert.

In einer besonders bevorzugten Ausführungsvariante können das Hochsicherheitsschlüsselpaar sowie das Profilhochsicherheitsschlüsselpaar nicht auf einem dauerhaften Speicher eines Clients gespeichert werden. Ein dauerhafter Speicher ist ein Speicher, welcher Daten auch nach dem Abschalten der Stromversorgung noch verfügbar macht. Dies bedeutet, dass ein Client diese Schlüssel nur in einem Arbeitsspeicher, maximal bis zur nächsten Abschaltung dieses Clients, verfügbar hat. Danach müssen sie neu

abgerufen werden. Damit der Zugriff auf diese Schlüssel in dieser Ausführungsvariante nicht verloren geht, muss zumindest auf dem Authentifikatorclient, welcher das Hochsicherheitsschlüsselpaar bzw. das Profilhochsicherheitsschlüsselpaar erzeugt hat, dieses Erzeugen reproduzierbar sein. Hierfür eignet sich bevorzugt das Erzeugen aus einem zufälligen Startwert, welcher gespeichert wird. Ein solcher Startwert wird im weiteren Verlauf der Beschreibung näher beschrieben.

Durch die nicht dauerhaft speicherbare Ausführungsvariante der privaten Hochsicherheitsschlüssel bzw. privaten Profilhochsicherheitsschlüssel wird vorteilhaft sichergestellt, dass ein Applikationsclient nie dauerhaft Zugriff auf den sensiblen Datenbereich des Tresors erhält, sondern dieser Zugriff immer vom Erzeugen aus dem zufälligen Startwert bzw. Abrufen der privaten (Profil-)Hochsicherheitsschlüssel vom Authentifikatorclient abhängig ist. Somit ist gegeben, dass der Authentifikatorclient den Zugriff auf den sensiblen Datenbereich nicht nur beim Koppeln von Applikationsclients, sondern jederzeit, kontrolliert.

In einer weiterhin bevorzugten Variante ist auch das Loginschlüsselpaar eines Authentifikatorclients nicht auf dauerhaftem Speicher speicherbar, sodass kein Applikationsclient sich dauerhaft auf dem Server anmelden kann. Damit wird vorteilhaft der Zugriff auf den Account ebenfalls dauerhaft über den Authentifikatorclient gesteuert.

In einer bevorzugten Ausführungsvariante der Nutzung der Zugriffssteuerung eines Metadatenbereichs und eines sensiblen Datenbereichs des Tresors kann ein gekoppelter Applikationsclient in einen offenen oder geschlossenen Zustand versetzt werden, wodurch dieser Applikationsclient Zugriff oder keinen Zugriff auf den privaten Hochsicherheitsschlüssel erhält. Damit wird vorteilhaft erreicht, dass ein Zugriff dieses Applikationsclients auf den sensiblen Datenbereich geregelt werden kann, wobei ein Applikationsclient im offenen Zustand Zugriff erhält, während ein Applikationsclient im geschlossenen Zustand nur Zugriff auf den Metadatenbereich erhält. Dieses Regeln bzw. das Versetzen in den offenen oder geschlossenen Zustand durch den Authentifikatorclient durchgeführt wird. In einer besonders bevorzugten Ausführungsvariante wird das Versetzen in den offenen oder geschlossenen Zustand durch das Umstellen eines binären Schalters innerhalb einer mobilen Applikation erreicht, wobei die mobile Applikation einen solchen binären Schalter für jeden gekoppelten Applikationsclient in einer Liste anzeigt. Weiterhin bevorzugt werden diese binären Schalter zum Einstellen des Zustands innerhalb der bereits beschriebenen Listenübersicht über alle Clients auf dem Authentifikatorclient dargestellt.

In einer besonders bevorzugten Ausführungsvariante der Steuerung des offenen oder geschlossenen Zustands eines Applikationsclients wird der Zustand durch die Existenz des

mit dem öffentlichen Sitzungsschlüssel des Applikationsclients verschlüsselten Hochsicherheitsschlüssels oder Profilhochsicherheitsschlüssels in einer Sitzung des Applikationsclients auf dem Server bestimmt. Existiert ein so verschlüsselter Hochsicherheitsschlüssel, so kann nur der Applikationsclient, der über den privaten Sitzungsschlüssel verfügt, diesen Hochsicherheitsschlüssel entschlüsseln und mit diesem auf den Tresor zugreifen. Der zum Hochsicherheitsschlüssel zugehörige Authentifikatorclient kann durch das Entfernen dieses Schlüssels vom Server den Applikationsclient vom offenen in den geschlossenen Zustand versetzen.

Durch die Möglichkeit, einen Applikationsclient in einen offenen Zustand zu versetzen, kann die Effizienz des Abrufens und Speicherns von geheimen Daten für ein besonders vertrauenswürdiges Gerät erhöht werden. Durch die Möglichkeit, einen Applikationsclient in einen geschlossenen Zustand zu versetzen, kann und muss jedes Abrufen oder Speichern von geheimen Daten durch den vertrauenswürdigen Authentifikatorclient autorisiert werden. Gleichzeitig kann der private Codierungsschlüssel bzw. private Profilschlüssel, im Gegensatz zu den äquivalenten Hochsicherheitsschlüsseln, auf dem Applikationsclient gespeichert werden, sodass der Applikationsclient, solange er eine aktive Sitzung auf dem Server besitzt, zumindest Zugriff auf den Metadatenbereich besitzt. Vorteilhaft kann der Applikationsclient dadurch zwar insbesondere keine geheimen Daten aus dem sensiblen Datenbereich für die Webanwendungen abrufen, aber zumindest über eine gespeicherte URL der Webanwendung oder einen gespeicherten Benutzernamen erkennen, dass geheime Daten für die Webanwendung im sensiblen Datenbereich im Tresor vorliegen.

In einer weiterhin bevorzugten Ausführungsvariante des erfindungsgemäßen Verfahrens wird die Integrität der auf dem Server gespeicherten öffentlichen Codierungsschlüssel, Hochsicherheitsschlüssel, Profilschlüssel, Profilhochsicherheitsschlüssel sowie Sitzungsschlüssel durch einen Signierprozess sichergestellt. Für den Signierprozess kommt ein privater Identifikationsschlüssels eines Identifikationsschlüsselpaars des zu dem jeweiligen öffentlichen, auf dem Server gespeicherten Schlüssel zugehörigen Push-Authentifikators oder Profils zum Einsatz. Der öffentliche Identifikationsschlüssel wird auf dem Server gespeichert. Bei dem Signierprozess wird aus dem privaten Identifikationsschlüssel und dem jeweiligen öffentlichen Schlüssel eine Signatur erzeugt, welche gemeinsam mit dem jeweiligen öffentlichen Schlüssel auf dem Server gespeichert wird. Mithilfe des öffentlichen Identifikationsschlüssels und der Signatur können der Server bzw. Dritte zweifelsfrei feststellen, dass die Signatur von dem Besitzer des privaten Identifikationsschlüssels kommt, in diesem Fall von dem Authentifikatorclient. Gleichzeitig wird die Integrität der signierten öffentlichen Schlüssel garantiert. Dies gilt, solange auch der öffentliche Identifikationsschlüssel auf dem Server korrekt und unverändert ist. Ist er das

jedoch nicht mehr, kann dieses Problem vorteilhaft wiederum durch den Authentifikatorclient festgestellt werden.

In einer weiterhin bevorzugten Ausführungsvariante des erfindungsgemäßen Verfahrens werden die zu einem Push-Authentifikator gehörigen Schlüsselpaare Loginschlüsselpaar, Identifikationsschlüsselpaar, Codierungsschlüsselpaar und Hochsicherheitsschlüsselpaar aus einem zufälligen Authentifikatorstartwert abgeleitet. Weiterhin werden die zu einem Profil gehörigen Schlüsselpaare Identifikationsschlüsselpaar, Profilschlüsselpaar und Profilhochsicherheitsschlüsselpaar ebenfalls aus einem zufälligen Authentifikatorstartwert abgeleitet. Dieses Ableiten geschieht besonders bevorzugt durch kollisionsfreie kryptografische Einwegfunktionen.

Das Ableiten der Schlüsselpaare hat den Vorteil, dass ein Push-Authentifikator mitsamt all seinen Schlüsseln aus einem einzigen zufälligen Startwert abgeleitet werden kann. Beim Autorisieren anderer Geräte durch das Übertragen von mehreren privaten Schlüsseln an diese, kann stattdessen einmalig der zufällige Authentifikator- oder Profilstartwert übertragen werden. Weitere bevorzugte Eigenschaften von Schlüsseln, dass beispielsweise Hochsicherheits- und Loginschlüssel aus Sicherheitsgründen nicht dauerhaft auf einem Applikationsclient gespeichert werden dürfen, werden den Schlüsseln bevorzugt nach dem Ableiten zugeordnet. Jeder zufällige Authentifikator- bzw. Profilstartwert, welcher Schlüssel Ableiten kann, welche nicht dauerhaft auf einem Applikationsclient gespeichert werden dürfen, darf selbst ebenfalls nicht dauerhaft auf einem Applikationsclient gespeichert werden. In einer besonders bevorzugten Ausführung werden die zufälligen Startwerte für Push-Authentifikatoren und Profile auf einem Authentifikatorclient auf dem Schlüsselelement des Authentifikatorclients gespeichert, vorteilhaft verschlüsselt und vor Angriffen geschützt.

Diese Ausführungsvariante hat zur Folge, dass anstelle des verschlüsselten Übertragens privater Schlüssel im erfindungsgemäßen Verfahren in der Regel der zufällige Authentifikator- oder Profilstartwert verschlüsselt übertragen wird. So wird insbesondere beim Autorisieren des Koppelns eines Applikationsclients der zufällige Authentifikatorstartwert anstelle des privaten Loginschlüssels verschlüsselt übertragen. Außerdem wird beim Autorisieren eines Speicherns oder Abrufens von geheimen Daten durch den Applikationsclient anstelle der privaten Codierungs- und Hochsicherheitsschlüssel bzw. Profil und Profilhochsicherheitsschlüssel der zufällige Authentifikatorstartwert bzw. Profilstartwert übertragen. In einer besonders bevorzugten Ausführungsvariante wird das Versetzen eines Applikationsclients in einen offenen oder geschlossenen Zustand gesteuert durch das Hinzufügen des mit dem öffentlichen Sitzungsschlüssel des Applikationsclients verschlüsselten zufälligen Authentifikator- bzw. Profilstartwerts, anstelle der in diesen zufälligen Startwerten enthaltenen privaten Hochsicherheitsschlüsseln, zu einer Sitzung.

In einer besonders bevorzugten Ausführungsvariante wird ein Profil nicht aus einem, sondern aus zwei zufälligen Profilstartwerten erzeugt, wobei aus einem zufälligen Profilstartwert das Profilschlüsselpaar und aus einem anderen zufälligen Profilhochsicherheitsstartwert das Identifikationsschlüsselpaar und das Profilhochsicherheitsschlüsselpaar erzeugt werden. Sind der symmetrische Schlüssel und symmetrische Hochsicherheitsschlüssel eines Tresors mithilfe eines Profils verschlüsselt, erlaubt diese Teilung bei einem gekoppelten Applikationsclient ebenfalls zwischen offenem und geschlossenem Zustand zu unterteilen, indem im Fall des geschlossenen Zustands der (nicht dauerhaft speicherbare) zufällige Profilhochsicherheitsstartwert nicht in der Sitzung zur Verfügung gestellt wird und somit durch den Applikationsclient keine sensiblen Daten aus dem Tresor abgerufen werden können.

In einer weiterhin bevorzugten Ausführungsvariante des erfindungsgemäßen Verfahrens werden die zu einem Push-Authentifikator gehörigen Schlüsselpaare Identifikationsschlüsselpaar, Codierungsschlüsselpaar und Hochsicherheitsschlüsselpaar (ausschließlich das Loginschlüsselpaar) zusätzlich zu dem zufälligen Authentifikatorstartwert aus einem zufälligen Saltwert abgeleitet. Weiterhin werden die zu einem Profil gehörigen Schlüsselpaare Identifikationsschlüsselpaar, Profilschlüsselpaar und Profilhochsicherheitsschlüsselpaar zusätzlich zu dem zufälligen Profilstartwert und Profilhochsicherheitsstartwert aus einem zufälligen Profilsaltwert abgeleitet. Saltwert und Profilsaltwert werden unverschlüsselt auf dem Server gespeichert. Das Ableiten von Schlüsseln aus zufälligen Startwerten kombiniert mit zufälligen Saltwerten hat den Vorteil, dass ein Angreifer keine privaten Schlüssel herausfinden kann, indem er verschiedene Startwerte, mittels sogenannter Rainbow Tables auch systematisch, durchtestet und bei gleichem Ergebnis eines Schlüssels den Startwert ableiten kann. Saltwerte erhöhen somit vorteilhaft die kryptografische Sicherheit.

In einer weiterhin bevorzugten Ausführungsvariante des erfindungsgemäßen Verfahrens umfasst das Verfahren eine Recovery-Funktion, welche es erlaubt, einen neuen Authentifikatorclient sowie die durch jeglichen Push-Authentifikator oder Profil des initialen Authentifikatorclients verschlüsselten geheimen Daten zu nutzen, ohne auf den initialen, also den ersten für diesen Account verwendeten, Authentifikatorclient Zugriff zu haben. Dafür werden Backup-Authentifikatoren genutzt, welche durch die Recovery-Funktion vorteilhaft auf dem neuen Authentifikatorclient initialisiert werden, um Zugriff auf die geheimen Daten des initialen Authentifikatorclients zu erhalten.

In einer besonders bevorzugten Ausführungsvariante der Recovery-Funktion wird beim Initialisieren des Push-Authentifikators auf dem initialen Authentifikatorclient zusätzlich ein weiterer Backup-Authentifikator erstellt, welcher eine Kopie der gleichen geheimen Daten

wie der Push-Authentifikator des initialen Authentifikatorclients verschlüsselt. Dabei wird der entsprechende zufällige Authentifikatorstartwert des Backup-Authentifikators, aus welchem sich jegliche Schlüssel dieses Backup-Authentifikators ableiten lassen, besonders bevorzugt in einer verschlüsselten und durch weitere Sicherheitsmaßnahmen geschützten Cloud gespeichert. Beispielhaft, aber nicht einschränkend, bieten iOS und Android für ihre mobilen Anwendungen verschlüsselte Cloud-Backups, die für den Backup-Authentifikator genutzt werden.

In einer alternativen, besonders bevorzugten Variante kann ein Backup-Code vom Nutzer aus der mobilen Applikation abgerufen werden. Für diesen Backup-Code wird beim erstmaligen Abrufen ein weiterer Backup-Authentifikator erstellt und die geheimen Daten des Nutzers mit diesem Backup-Authentifikator verschlüsselt. Aus dem Backup-Code wird besonders bevorzugt mithilfe eines Backup-Saltwerts, welcher auf dem Server gespeichert wird, ein Authentifikatorstartwert mithilfe einer Hashfunktion abgeleitet. Der Backup-Saltwert erhöht vorteilhaft die Sicherheit dieser Ausführung der Recovery-Funktion. Für die Sicherheit gegen Ausspähen, Manipulieren und Verlust des Backup-Codes ist in dieser Ausführungsform der Recovery-Funktion der Nutzer selbst verantwortlich.

In einer weiterhin besonders bevorzugten Ausführungsvariante der Recovery-Funktion werden beim Initialisieren des neuen Authentifikatorclients durch eine der beschriebenen oder eine andere Recovery Funktion in allen durch den jeweiligen Backup-Authentifikator des initialen Authentifikatorclients verschlüsselten Tresoren die geheimen Daten in eine einzelne Übergabe zusammengefasst. Der oder die Tresore werden dann mittels eines oder bevorzugt zwei neuen symmetrischen Schlüssels verschlüsselt. Diese symmetrischen Schlüssel werden durch neue öffentliche Codierungsschlüssel und bevorzugt neue öffentliche Hochsicherheitsschlüssel eines neu initialisierten Push-Authentifikators auf dem neuen Authentifikatorclient verschlüsselt. Alle Schlüssel des initialen Authentifikatorclients, einschließlich des genutzten Backup-Authentifikators, werden gelöscht, sodass alle Tresore vorteilhaft neu verschlüsselt sind und langfristig angelegte Angriffe auf die geheimen Daten über die Recovery-Funktion spätestens ab diesem Zeitpunkt verhindert werden. Bevorzugt werden auf dem neuen Authentifikatorclient neue Recovery-Funktionen durch die beschriebenen Backup-Authentifikatoren angelegt.

In einer weiterhin bevorzugten Ausführungsvariante des erfindungsgemäßen Verfahrens umfasst ein Applikationsclient eine zugehörige, mit und auf dem Applikationsclient lokal verbundene Applikationsclienterweiterung. In einer besonders bevorzugten Ausführung ist die Applikationsclienterweiterung eine Browsererweiterung in einem Browser auf dem Applikationsclient. Die Applikationsclienterweiterung dient dazu, Metadaten der geheimen Daten eines Tresors, zu dem die Sitzung dieses Applikationsclients zugeordnet ist, zu

speichern und abzurufen. Dadurch können durch die Applikationsclienterweiterung vorteilhaft insbesondere Webseiten, die im Browser aufgerufen werden, als bekannte Webseiten identifiziert werden, für welche bereits geheime Daten vorliegen.

In einer besonders bevorzugten Ausführungsvariante generiert die Applikationsclienterweiterung Overlays, insbesondere für die Formulare zum Login bei bekannten Webseiten. Die Overlays werden besonders vorteilhaft anstelle der originalen Formulare eingeblendet, indem der Quellcode der Webseite entsprechend angepasst wird. Somit kann ein Nutzer vorteilhaft sofort erkennen, dass bereits geheime Daten mithilfe des erfindungsgemäßen Verfahrens gespeichert wurden und diese entsprechend abrufen. Die Overlays fragen weiterhin bei dem Nutzer ab, ob die bereits eingegebenen geheimen Daten in dem entsprechenden Tresor gespeichert werden sollen, beispielsweise wenn bei einem erfolgreichen Loginversuch auf einer Webseite keine bisher gespeicherten Logindaten für diese Webseite vorliegen. In einer weiteren Ausführungsvariante generiert die Applikationsclienterweiterung Overlays auch für Formulare, welche den Nutzer zur Eingabe von insbesondere, aber nicht ausschließlich Adress-, Zahlungs- und Kreditkarteninformationen auffordern. In einer weiteren Ausführungsvariante generiert die Applikationsclienterweiterung Overlays, wenn der Nutzer oder ein anderer Nutzer für eine bestimmte URL eine geheime Notiz hinterlassen hat und fragt deren Anzeige ab. Damit können vorteilhaft sicher Notizen speziell für bestimmte Webanwendungen gespeichert und geteilt werden.

In einer weiterhin besonders bevorzugten Ausführungsvariante führt die Applikationserweiterung das Erkennen bekannter Webseiten, für welche geheime Daten vorliegen, das Generieren und Einblenden von Overlays sowie ein automatisches Ausfüllen der Formulare, insbesondere Formulare zum Login, mit allen im aktuellen Zustand des Applikationsclients verfügbaren geheimen Daten, automatisch aus. Für den Nutzer bleibt nur noch die Aufgabe übrig, bei einem geschlossenen Zustand des Applikationsclients das Speichern bzw. Abrufen von geheimen Daten mittels des Authentifikatorclients zu bestätigen, wobei die Applikationsclienterweiterung auch für diese Aufgabe ein spezielles Overlay generiert und einblendet. Somit kann vorteilhaft eine Nutzererfahrung gewährleistet werden, die ähnlich komfortabel wie eine SSO-Lösung ist.

In einer weiterhin bevorzugten Ausführungsvariante des erfindungsgemäßen Verfahrens wird jegliches Autorisieren des Koppelns von Applikationsclients sowie des Speicherns und Abrufens von geheimen Daten durch einen Authentifikatorclient durch eine Wischgeste erreicht, die auf dem in dieser Ausführungsvariante als Authentifikatorclient eingesetzten Smartphone durchgeführt wird. In einer besonders bevorzugten Variante ist auf dem Authentifikatorclient eine bereits beschriebene mobile Anwendung installiert, die in dieser

Ausführungsvariante im Fall einer benötigten Autorisierung den Nutzer mithilfe einer einfachen grafischen Benutzeroberfläche zu der Wischgeste auffordert. In einer besonders bevorzugten Ausführungsvariante wird zusätzlich zum Durchführen der Wischgeste die Sicherheitsfunktion des Smartphones, beispielsweise eine biometrische Authentifizierung mittels Fingerabdrucks oder Gesichtsidentifikation, abgefragt. Somit wird vorteilhaft eine 2FA erreicht mithilfe einer für den Nutzer intuitiven und bekannten Authentifizierung. Die auf dem Smartphone ist ebenfalls eine besonders einfache und intuitive Art der Autorisierung und erhöht vorteilhaft die Benutzerfreundlichkeit des gesamten Verfahrens.

In einer weiterhin bevorzugten Ausführungsvariante des erfindungsgemäßen Verfahrens umfasst eine Sitzung eines mit dem Server gekoppelten Clients einen Sitzungstoken, welcher neben dem öffentlichen Sitzungsschlüssel auf dem Server gespeichert wird. Der vorhandene Sitzungstoken erlaubt es vorteilhaft dem Client, Anfragen an den Server zu senden. Das Entfernen des Sitzungstokens erlaubt es in dieser Ausführungsvariante, Sitzungen von Applikationsclients, aber auch Authentifikatorclients zu kontrollieren, indem durch Entfernen des Sitzungstokens deren Anfragen an den Server abgelehnt werden. Ein Client, dessen Anfrage an den Server abgelehnt wird, löscht in einer besonders bevorzugten Ausführung seine gespeicherten Schlüssel. Der Sitzungstoken ist somit eine besonders einfache Art, die Kontrolle über Sitzungen von Clients durch einen Authentifikatorclient auszuüben.

In einer weiterhin bevorzugten Ausführungsvariante des erfindungsgemäßen Verfahrens wird das Koppeln eines Applikationsclients in Schritt S01) initialisiert durch das Eingeben eines öffentlichen Austauschschlüssels eines von diesem Applikationsclient generierten asymmetrischen Austauschschlüsselpaars in den Authentifikatorclient. Das Eingeben umfasst dabei jegliche Form der Datenerfassung durch den Authentifikatorclient, insbesondere aber nicht einschränkend durch physisches Eintippen durch den Nutzer oder Erfassen von Bild-, Funk oder Toninformationen. Der Authentifikatorclient verschlüsselt seinen privaten Loginschlüssel oder bevorzugt seinen zufälligen Authentifikatorstartwert mit diesem öffentlichen Austauschschlüssel und sendet das verschlüsselte Resultat an den Applikationsclient, welcher den privaten Loginschlüssel oder bevorzugt den zufälligen Authentifikatorstartwert mit dem privaten Austauschschlüssel entschlüsseln kann. Vorteilhaft wird durch das Verschlüsseln eine Sicherheit der Übertragung gewährleistet. Das Senden des verschlüsselten privaten Loginschlüssels oder zufälligen Authentifikatorstartwerts geschieht bevorzugt mithilfe des Servers als Proxy, also in der Funktion der Weiterleitung. Besonders bevorzugt wird die Integrität des versendeten verschlüsselten privaten Loginschlüssels oder zufälligen Authentifikatorstartwerts mittels einer Signatur des Authentifikatorclients, besonders bevorzugt mittels dessen Identifikationsschlüssels,

sichergestellt. Der Applikationsclient kann Authentifikatorclient als Absender sowie die Integrität des versendeten verschlüsselten privaten Loginschlüssels oder zufälligen Authentifikatorstartwerts feststellen, wenn der Server dem Applikationsclient den öffentlichen Identifikationsschlüssel zusätzlich mitsendet.

In einer besonders bevorzugten Ausführungsvariante wird das Austauschschlüsselpaar nur einmalig verwendet, sodass vorteilhaft kein Spam, Schadcode oder fehlerhaften Informationen mithilfe der dauerhaft gleichen Verschlüsselung an den Applikationsclient gesendet werden können. Besonders bevorzugt wird das Austauschschlüsselpaar zeitgesteuert, besonders bevorzugt aller 30 Sekunden, 60 Sekunden oder 5 Minuten, erneuert.

In einer besonders bevorzugten Ausführungsvariante des Koppelns eines Applikationsclients mittels eines asymmetrischen Austauschschlüsselpaars ist der öffentliche Austauschschlüssel in einem vom Applikationsclient generierten und angezeigten Austausch-QR-Code enthalten, welcher vom Authentifikatorclient visuell und besonders bevorzugt per Kamera erfasst wird. Diese Ausführung erlaubt vorteilhaft eine besonders hohe Benutzerfreundlichkeit. In einer weiterhin besonders bevorzugten Ausführungsvariante ist eine mobile Anwendung, die das erfindungsgemäße Verfahren auf einem als Authentifikatorclient genutzten Smartphones ausführt, beim Betriebssystem des Smartphones als Verarbeiter von zumindest einer solchen bestimmten Art von QR-Codes, welche für das Koppeln von Applikationsclients verwendet wird, registriert. Dadurch wird diese mobile Anwendung automatisch gestartet, wenn ein solcher QR-Code durch eine beliebige Kameraanwendung des Smartphones erfasst wird. Dies erhöht wiederum die zeitliche Effizienz des Verfahrens und damit vorteilhaft die Benutzerfreundlichkeit.

Allgemein können Ausführungsbeispiele der vorliegenden Erfindung als Programm, Firmware, Computerprogramm oder Computerprogrammprodukt mit einem Programmcode oder als Daten implementiert sein, wobei der Programmcode oder die Daten dahin gehend wirksam ist bzw. sind, eines der Verfahren durchzuführen, wenn das Programm auf einer Recheneinheit abläuft. Der Programmcode, das Computerprogrammprodukt oder die Daten kann bzw. können beispielsweise auch auf einem maschinenlesbaren Träger oder Datenträger gespeichert sein. Der Programmcode oder die Daten können unter anderem als Quellcode, Maschinencode oder Bytecode sowie als anderer Zwischencode vorliegen.

Eine Recheneinheit kann durch einen Prozessor, einen Computerprozessor (CPU = Central Processing Unit), einen Grafikprozessor (GPU = Graphics Processing Unit), einen Computer, ein Computersystem, einen anwendungsspezifischen integrierten Schaltkreis (ASIC = Application-Specific Integrated Circuit), einen integrierten Schaltkreis (IC = Integrated

Circuit), ein Ein-Chip-System (SOC = System on Chip), ein programmierbares Logikelement oder ein feldprogrammierbares Gatterarray mit einem Mikroprozessor (FPGA = Field Programmable Gate Array) gebildet sein.

Die vorliegende Erfindung umfasst neben dem umfänglich beschriebenen Verfahren außerdem den Server, welcher zumindest eine Recheneinheit zur Ausführung von Computerprogrammen, Netzwerkanschlüsse zum Austausch von Daten zumindest mit allen mit dem Server verbundenen Clients in einem Netzwerk, sowie zumindest ein Speichermedium zum Speichern von Computerprogrammen sowie Schlüsseln, Werten, Sitzungen und Tresoren. In bevorzugter Ausführung ist die Recheneinheit eine CPU (central processing unit), welche vielseitig für Operationen anwendbar ist, aber nicht auf besonders spezialisierte Rechenoperationen angepasst ist. In bevorzugter Ausführung umfasst der Server weiterhin einen anwendungsspezifischen integrierten Schaltkreis (ASIC) zur effizienten Verschlüsselung, zum Prüfen von Signaturen oder Zertifikaten und dem Ausführen von Hashfunktionen.

Die vorliegende Erfindung umfasst neben dem umfänglich beschriebenen Verfahren und dem Server ein Computerprogrammprodukt A. Dieses umfasst Befehle, die bei der Ausführung durch eine Recheneinheit auf einem Authentifikatorclient diesen dazu veranlassen, das erfindungsgemäße Verfahren auszuführen. Dies betrifft im Speziellen die Verfahrensschritte und -merkmale, die auf dem Authentifikator ausgeführt werden. Hierzu gehören insbesondere, aber nicht einschränkend, folgende Verfahrensschritte und -merkmale:

- Das Generieren bzw. Ableiten aller asymmetrischer Schlüsselpaare auf einem Authentifikatorclient,
- Das Erstellen und Senden sowie Empfangen und Auswerten von Anfragen zum oder vom Server
- Zugreifen auf Schnittstellen zur Nutzung von Kamera, Schutzelement, Schutzfunktion und Speicher des Authentifikatorclients
- Speichern von privaten Schlüsseln und Werten
- Das Bereitstellen einer Benutzeroberfläche, welche Funktionen zum Autorisieren (z.B. mittels Wischgeste), Auflisten aller gekoppelten Clients, Ändern des Status und Entfernen der Sessions von Clients, Erstellen und Teilen eines Profils, Eingeben eines öffentlichen Austauschschlüssels zum Koppeln eines Applikationsclients (insbesondere per Kamera und QR-Code), Auffordern des Nutzers zum Verifizieren über die Schutzfunktion des Smartphones als Authentifikatorclient, bereitstellt.

Die vorliegende Erfindung umfasst neben dem umfänglich beschriebenen Verfahren, dem Server und einem Computerprogrammprodukt A auch ein Computerprogrammprodukt B. Dieses umfasst Befehle, die bei der Ausführung durch eine Recheneinheit auf einem Applikationclient diesen dazu veranlassen, das erfindungsgemäße Verfahren auszuführen. Dies betrifft im Speziellen die Verfahrensschritte und -merkmale, die auf dem Authentifikator ausgeführt werden. Hierzu gehören insbesondere, aber nicht einschränkend, folgende Verfahrensschritte und -merkmale:

- Das Generieren bzw. Ableiten aller asymmetrischer Schlüsselpaare auf einem Applikationsclient,
- Das Erstellen und Senden sowie Empfangen und Auswerten von Anfragen zum oder vom Server
- Zugreifen auf Schnittstellen zur Nutzung von Schutzelement und Speicher des Applikationsclients
- Speichern von privaten Schlüsseln und Metadaten von geheimen Daten
- Zugriff auf Schnittstelle zu lokaler Kommunikation mit einer zugehörigen Applikationsclienterweiterung
- Das Bereitstellen zumindest einer Benutzeroberfläche, welche Funktionen zum Abrufen des Sitzungsstatus, Erstellen und Anzeigen eines öffentlichen Austauschschlüssels zum Koppeln mit einem Server und Authentifikatorclient (insbesondere per QR-Code), Erkennen des Bekanntheitsstatus einer Webanwendung sowie Erstellen und Einblenden von Overlays, Anzeigen einer Autorisierungsaufforderung für den Authentifikatorclient, automatisches Ausfüllen von geheimen Daten, Speichern von geheimen Daten, bereitstellt.

Das Computerprogrammprodukt B besteht aus bis zu zwei Teilen, einem primären Computerprogrammprodukt des Applikationsclients und einer Applikationsclienterweiterung. In einer bevorzugten Ausführungsvariante ist das primäre Computerprogrammprodukt des Applikationsclients eine Webanwendung, während die Applikationsclienterweiterung als Browser-Addin ausgeführt ist. In dieser bevorzugten Ausführung wird das Koppeln einschließlich Anzeigen des öffentlichen Austauschschlüsselpaars sowie Funktionen wie ein Abruf des Status des Applikationsclients in der Webanwendung durchgeführt. Das Browser-Addin übernimmt dagegen alle Funktionen und Schritte des Verfahrens bezüglich des Speicherns von Metadaten von geheimen Daten, des Erstellens und Anzeigens von Overlays, des Ausfüllens von Formularen sowie des Speicherns und Abrufens von geheimen Daten auf und vom Server.

Die vorliegende Erfindung umfasst neben dem umfänglich beschriebenen Verfahren, dem Server und den Computerprogrammprodukten A und B auch ein Computerprogrammprodukt C. Dieses umfasst Befehle, die bei der Ausführung durch eine Recheneinheit auf dem Server diesen dazu veranlassen, das erfindungsgemäße Verfahren auszuführen. Dies betrifft im Speziellen die Verfahrensschritte und -merkmale, die auf dem Server ausgeführt werden. Hierzu gehören insbesondere, aber nicht einschränkend, folgende Verfahrensschritte und -merkmale:

- Das Generieren bzw. Ableiten aller symmetrischer Schlüssel,
- Zugreifen auf die Netzwerkanschlüsse zum Erstellen und Senden sowie Empfangen und Auswerten von Anfragen zu oder von den Clients, insbesondere das Agieren als Proxy zur Weiterleitung der Kommunikation zwischen Clients,
- Zugreifen auf den Speicher sowie spezielle ASICs zur Verschlüsselung, Überprüfung von Signaturen und Zertifikaten oder Ausführung von Hashfunktionen,
- Speichern von verschlüsselten privaten Schlüsseln, öffentlichen Schlüsseln und geheimen Daten in Tresoren, insbesondere Generieren einer systematischen datenbankähnlichen Datenverwaltung, sodass Tresore, Schlüssel, Werte, Sitzungen, Tokens und Accountdaten einander und den jeweiligen Nutzern und Clients korrekt zugeordnet werden,

Die vorliegende Erfindung umfasst neben dem umfänglich beschriebenen Verfahren, dem Server und den Computerprogrammprodukten A, B und C auch ein System, welches aus der Kombination der Computerprogrammprodukte A, B und C sowie dem Server besteht. Dieses System ist in der Lage, das vorliegende erfindungsgemäße Verfahren vollumfänglich auszuführen. Voraussetzung ist hierfür die Verwendung von geeigneten bereits spezifizierten, aber nicht von der vorliegenden Erfindung umfassten Authentifikatorclients bzw. Applikationsclients, welche dafür eingerichtet sind, zumindest die Computerprogrammprodukte A bzw. B auszuführen.

Weitere Vorteile, Merkmale und Anwendungsmöglichkeiten der vorliegenden Erfindung ergeben sich auch aus der nachfolgenden Beschreibung von Ausführungsbeispielen und den Zeichnungen. Dabei können die in den Ansprüchen und in der Beschreibung erwähnten Merkmale jeweils einzeln für sich oder in beliebiger Kombination erfindungswesentlich sein. Zudem ist darauf hinzuweisen, dass der Fachmann zweifelsohne erkennt, dass sich die einzelnen Merkmale, die in den vorstehenden konkreten Ausführungsformen beschrieben sind, auf angemessene Weise miteinander kombinieren lassen, soweit kein Widerspruch vorliegt, wobei zum Vermeiden unnötiger Wiederholung auf eine separate Beschreibung verschiedener möglicher Kombinationen verzichtet wird.

AUSFÜHRUNGSBEISPIELE

Die oben beschriebenen Eigenschaften, Merkmale und Vorteile dieser Erfindung sowie die Art und Weise, wie diese erreicht werden, werden klarer und deutlicher verständlich im Zusammenhang mit den folgenden Zeichnungen und Beschreibungen der Grundprinzipien und Ausführungsbeispiele der vorliegenden Erfindung.

Dabei wird die vorliegende Erfindung näher erläutert, ohne die Erfindung auf diese Zeichnungen und Beschreibungen zu beschränken. Sofern in dieser Anmeldung der Begriff "kann" verwendet wird, handelt es sich sowohl um die technische Möglichkeit als auch um die tatsächliche technische Umsetzung. In den einzelnen Figuren gezeigte und zu dem jeweiligen Beispiel beschriebene Merkmale sind nicht auf das jeweilige Einzelbeispiel beschränkt. Der Singular schließt den Plural ein, es sei denn, aus dem Kontext geht eindeutig etwas anderes hervor.

In den nachfolgenden Zeichnungen sind alle kryptografischen Schlüssel als Farbkästen dargestellt und mit einem Schlüsselsymbol versehen. Alle Schlüssel oder Werte, die in der hellgrauen Farbe hinterlegt sind, sind geheime Schlüssel oder Werte, welche nur auf dem jeweiligen Gerät vorhanden sind, zu dem sie gehören. Weiß hinterlegte Schlüssel oder Werte sind ebenfalls geheim, liegen aber auf dem Server, jedoch ausschließlich in verschlüsselter Form, vor. Dunkelgrau hinterlegte Schlüssel oder Werte sind öffentliche Schlüssel oder Werte und können serverseitig unverschlüsselt vorliegen.

Nachfolgend zeigt:

Fig. 1: Das Wirkprinzip der Bedienoberfläche des erfindungsgemäßen Verfahrens in Schritt S03). Dabei speichert oder ruft der Nutzer geheime Daten auf einem Applikationsclient (1.3), hier ein Laptop, ab. Diese Funktion wird in der hier dargestellten, bevorzugten Ausgestaltung über Overlays (1.4.2) durch die Applikationsclienterweiterung (1.4.1) bereitgestellt, in dieser bevorzugten Ausführung als Browser-Addin ausgeführt. Verschiedenartige Formulare (1.4.3) verschiedener Webanwendungen können durch die Applikationsclienterweiterung (1.4.1) automatisch erkannt und das erfindungsgemäße Overlay (1.4.2) angezeigt werden. Speichert oder ruft der Nutzer geheime Daten ab, wird er auf seinem Authentifikatorclient (1.2), in dieser bevorzugten Ausführungsvariante ein Smartphone, aufgefordert, diesen Schritt zu bestätigen.

Fig. 2: Die zugrundeliegende Architektur des Systems der am Verfahren beteiligten Geräte. Die Pfeile stellen die verfahrensspezifischen Kommunikationskanäle

dar. Ein Authentifikatorclient (1.2) kommuniziert über einen Server (1.1) mit einem Applikationsclient (1.3) bzw. dessen zugehöriger Applikationsclienterweiterung (1.4.1). Sowohl der Authentifikatorclient (1.2) als auch der Applikationsclient (1.3) und dessen Applikationsclienterweiterung (1.4.1) kommunizieren mit dem Server über verschlüsselte Kanäle, dargestellt durch die durchgezogenen Pfeile, in dieser bevorzugten Ausführung über Kanäle mit Verschlüsselung nach TLS-Protokoll. Zusätzlich dient der Server (1.1) als ein Kommunikationsproxy zwischen den Clients, diese Funktionsweise wird dargestellt mit den gepunkteten Pfeilen. In diesem Fall kommunizieren die Clients mittels Ende-zu-Ende verschlüsselter Nachrichten innerhalb der verschlüsselten Kanäle. Nur zwischen Applikationsclient (1.3) und dessen Applikationsclienterweiterung (1.4.1) kommt es zu einer lokalen, unverschlüsselten Kommunikation auf demselben Gerät, z.B. zwischen dem Gerätespeicher und dem Browser-Addin.

Fig. 3: Ein Vergleich der verschiedenen Schlüssel, welche ein Push-Authentifikator (1.5) und ein Profil (1.7) besitzen. Ein Profil (1.7) besitzt im Gegensatz zum Push-Authentifikator (1.5) kein Loginschlüsselpaar aus öffentlichem (2.1.2) und privatem (2.1.3) Loginschlüssel, um sich selbstständig gegen den Server zu authentifizieren. Sowohl Profil (1.7) als auch Push-Authentifikator (1.5) besitzen ein Identifikationsschlüsselpaar aus öffentlichem (2.2.2) und privatem (2.2.3) Identifikationsschlüssel, wobei alle weiteren beschriebenen öffentlichen Schlüssel des Profils (1.7) bzw. Push-Authentifikators (1.5), welche auf dem Server gespeichert werden, mittels des privaten Identifikationsschlüssels (2.2.3) signiert werden, wodurch mittels des Signierprozesses (2.2.4) die Integrität der signierten öffentlichen Schlüssel sichergestellt werden kann. Profil (öffentlicher (2.5.2) und privater (2.5.3) Profilschlüssel und öffentlicher (2.6.2) und privater (2.6.3) Profilhochsicherheitsschlüssel) und Push-Authentifikator (öffentlicher (2.3.2) und privater (2.3.3) Codierungsschlüssel und öffentlicher (2.4.2) und privater (2.4.3) Hochsicherheitsschlüssel) besitzen unterschiedlich benannte Schlüssel zum Verschlüsseln von zum Profil bzw. Push-Authentifikator gehörigen Tresoren. Das Codierungsschlüsselpaar (2.3.2 und 2.3.3) und Hochsicherheitsschlüsselpaar (2.4.2 und 2.4.3) des Push-Authentifikators (1.5) dienen allerdings außerdem dazu, jegliche Schlüssel eines diesem Push-Authentifikator zugeordneten Profils (1.7) zu entschlüsseln, um Zugriff auf dieses Profil zu erhalten. In der dargestellten Ausführungsvariante

werden jegliche Schlüssel des Profils bzw. des Push Authentifikators aus dem zufälligen Profilstartwert (4.2.1) und dem zufälligen Profilhochsicherheitsstartwert (4.2.2), bzw. dem zufälligen Authentifikatorstartwert (4.1) abgeleitet. Dabei existieren sowohl Profilstartwert (4.2.1) als auch Profilhochsicherheitsstartwert (4.2.2), um von einem Profil das Profilschlüsselpaar (2.5.2 und 2.5.3) und das Profilhochsicherheitsschlüsselpaar (2.6.2 und 2.6.3) einzeln abrufen zu können, z.B. je nachdem, ob das Profil in offenem oder geschlossenem Zustand ist. In dieser Ausführungsvariante werden alle Schlüssel von dem Push-Authentifikator (1.5) bzw. dem Profil (1.7) zusätzlich aus einem zufälligen Saltwert (4.3) bzw. einem zufälligen Profilsaltwert (4.4) abgeleitet.

Fig. 4: Die Verschlüsselung eines Tresors (3.3.1) auf einem Server (1.1), in dieser Ausführungsvariante mittels eines Profils (1.7), wobei angemerkt sei, dass eine solche Verschlüsselungsvariante in einer Ausführungsvariante ohne Profil auch direkt durch einen Push-Authentifikator vorgenommen werden kann. Die auf dem Authentifikatorclient (1.2) erzeugten und zum Profil gehörenden öffentlichen (2.2.2, 2.5.2, 2.6.2) und privaten (2.2.3, 2.5.3, 2.6.3) Schlüssel werden aus zumindest einem zufälligen Profilstartwert (4.2.1), einem zufälligen Profilhochsicherheitsstartwert (4.2.2) und einem zufälligen Profilsaltwert (4.4) erzeugt. Die öffentlichen Schlüssel (2.2.2, 2.5.2, 2.6.2) werden auf dem Server gespeichert, wobei der öffentliche Profilhochsicherheitsschlüssel (2.6.2) genutzt wird, um den symmetrischen Hochsicherheitsschlüssel (3.2) zu verschlüsseln und der öffentliche Profilschlüssel (2.5.2) genutzt wird, um zumindest den symmetrischen Schlüssel (3.1) zu verschlüsseln. In einer bevorzugten Variante werden durch den öffentlichen Profilschlüssel (2.5.2) zusätzlich beide symmetrischen Schlüssel (3.1, 3.2) verschlüsselt. Mit den symmetrischen Schlüsseln können wiederum geheime Daten aus dem Tresor (3.3.1) abgerufen werden, welche dort systematisch und zeitlich sortiert in einzelnen Übergaben (3.3.7) für einzelne Webanwendungen gespeichert sind.

Fig. 5: Die Zugriffsbeschränkung auf ein Profil (1.7) durch Verschlüsselungen durch einen Push-Authentifikator (1.5) und einen zweiten, in dieser Ausführungsvariante Backup-Authentifikator (1.6). Im Profil links unten werden wie in Fig.4 aus einem zufälligen Profilstartwert (4.2.1), einem zufälligen Profilhochsicherheitsstartwert (4.2.2) und einem zufälligen Profilsaltwert (4.4) zum Profil zugehörige Schlüsselpaare erzeugt, wobei

diese wie in Fig.4 zum Verschlüsseln eines serverseitigen Tresors genutzt werden können. Während der zufällige Profilsaltwert (4.4) öffentlich auf dem Server gespeichert wird, werden der zufällige Profilstartwert (4.2.1) zwar verschlüsselt durch den öffentlichen Codierungsschlüssel (2.3.2) eines Push-Authentifikators (1.5) und der zufällige Profilhochsicherheitsstartwert (4.2.2) verschlüsselt durch zumindest den öffentlichen Hochsicherheitsschlüssel (2.4.2) des Push-Authentifikators (1.5) auf dem Server gespeichert. Dadurch kann nur dieser Push-Authentifikator (1.5) Zugriff auf dieses Profil erlangen, indem er die zufälligen Startwerte (4.2.1 und 4.2.2) des Profils entschlüsselt. Zusätzlich ist in dieser Ausführungsvariante ein Backup-Authentifikator (1.6) eingerichtet, mit dem die zufälligen Startwerte (4.2.1 und 4.2.2) des Profils noch einmal in gleicher Weise verschlüsselt auf dem Server gelagert werden. Dieser Backup-Authentifikator(1.6) kann durch Übertragen des zufälligen Authentifikatorstartwerts (4.1) und des zufälligen Saltwerts (4.3) auf einem anderen Authentifikatorclient erstellt werden und dort im Fall eines Verlusts des initialen Authentifikatorclients (1.2) dort alle Schlüssel (2.1.2, 2.1.3, 2.2.2, 2.2.3, 2.3.2, 2.3.3, 2.4.2, 2.4.3) erzeugen, um das Profil (1.7) zu entschlüsseln und damit Zugriff zu erhalten.

Fig. 6: Die öffentlichen und privaten Schlüssel, welche einem Profil (1.7) und einem Push-Authentifikator (1.5) auf einem Applikationsclient in nicht gekoppeltem, gekoppeltem aber geschlossenen, sowie gekoppeltem und offenen Zustand zur Verfügung stehen. In nicht gekoppeltem Zustand generiert der Applikationsclient nur einen öffentlichen (2.8.2) und einen privaten (2.8.3) Austauschschlüssel, um von einem Authentifikatorclient Schlüssel oder Startwerte verschlüsselt übertragen bekommen zu können. Im gekoppelten Zustand besitzt der Applikationsclient eine Sitzung (2.7.4) mit einem Sitzungstoken (2.7.5) und einem öffentlichen (2.7.2) und einem privaten (2.7.3) Sitzungsschlüssel. Ein Push- oder Backup-Authentifikator (1.5 oder 1.6) besitzt außerdem ein Loginschlüsselpaar aus öffentlichem (2.1.2) und privatem (2.1.3) Loginschlüssel, um sich gegen den Server zu authentifizieren, was ein Profil nicht kann. Darüber hinaus besitzt ein Profil (1.7) oder Push-/Backup-Authentifikator (1.5 / 1.6) auf einem Applikationsclient alle insbesondere in Fig. 3 beschrieben Schlüssel. Allerdings kann durch den offenen Zustand gesteuert werden, ob Push-/Backup-Authentifikator (1.5 / 1.6) auf einem Applikationsclient Zugriff auf dessen privaten Hochsicherheitsschlüssel (2.4.3), Identifikationsschlüssel (2.2.3) und Loginschlüssel (2.1.3) bzw. ein Profil (1.7) Zugriff auf dessen

privaten Identifikationsschlüssel und Profilhochsicherheitsschlüssel hat. Dies erlaubt oder verbietet entsprechend den Zugriff auf sensible Daten innerhalb der geheimen Daten und kann über einen Authentifikatorclient gesteuert werden.

Fig. 7: Eine Illustration der Zugriffsberechtigung auf einen Tresor (3.3.1) und die enthaltenen geheimen Daten durch den Besitz und den Einsatz bestimmter Schlüssel. In dieser Ausführungsvariante greift ein Push-Authentifikator direkt auf den Tresor (3.3.1) zu, anders als in der ebenso möglichen Ausführungsvariante in Fig.4, in der ein Profil als Indirektionsebene für den Push-Authentifikator auf den Tresor (3.3.1) zugreift. Der Push-Authentifikator ist im Besitz des Codierungsschlüsselpaars (2.3.1), bestehend aus einem privaten (2.3.3) und einem öffentlichen (2.3.2) Codierungsschlüssel. Mit diesen Schlüsseln kann der Push-Authentifikator den symmetrischen Schlüssel (3.1) ver- und entschlüsseln, um Zugriff auf den durch diesen symmetrischen Schlüssel (3.1) verschlüsselten Tresor zu erhalten. In dieser bevorzugten Ausführungsvariante ist der Tresor allerdings in einen Metadatenbereich (3.3.5) und einen sensiblen Datenbereich (3.3.3) unterteilt, wobei der sensible Datenbereich (3.3.3) zusätzlich durch einen symmetrischen Hochsicherheitsschlüssel (3.2) verschlüsselt ist. Mithilfe des symmetrischen Schlüssels (3.1) allein kann der Push-Authentifikator nur auf den Metadatenbereich (3.3.5) und die darin enthaltenen Metadaten (3.3.6) zugreifen. Für den Zugriff auf die sensiblen Daten (3.3.4) im sensiblen Datenbereich (3.3.3) benötigt der Push-Authentifikator das Hochsicherheitsschlüsselpaar (2.4.1) aus einem privaten (2.4.3) und einem öffentlichen (2.4.2) Hochsicherheitsschlüssel zum Ver- und Entschlüsseln des symmetrischen Hochsicherheitsschlüssels (3.2). Der private Hochsicherheitsschlüssel (2.4.3) kann vom Push-Authentifikator nicht dauerhaft gespeichert werden, sondern muss, siehe Fig.6, mittels des zufälligen Authentifikatorstartwerts generiert bzw. abgeleitet werden, was wiederum nur im offenen Zustand des Push-Authentifikators möglich ist.

Fig. 8: Eine Illustration einer bevorzugten Ausführungsvariante der Applikationsclienterweiterung (1.4.1) mit deren Funktionen und den verschiedenen Overlays (1.4.2). Das erfindungsgemäße Verfahren wird hier zum passwortbasierten Login auf eine Webanwendung eingesetzt. Im ersten Zustand links oben zeigt eine Webanwendung, hier beispielsweise Github.com, deren standardmäßiges Formular (1.4.3) zum Login. Daher

muss die der Applikationsclient und die Applikationsclienterweiterung (1.4.1) mit einem Authentifikatorclient gekoppelt werden, indem in dieser bevorzugten Ausführungsvariante der von der Applikationsclienterweiterung (1.4.1) dargestellte QR-Code (2.8.4) durch den Authentifikatorclient gescannt wird. Dieser QR-Code (2.8.4) enthält einen öffentlichen Austauschschlüssel (2.8.2), wodurch Loginschlüssel oder zufällige Startwerte verschlüsselt und sicher zwischen dem Authentifikatorclient und dem Applikationsclient ausgetauscht und das Koppeln des Applikationsclients durchgeführt werden kann. Im gekoppelten Zustand erkennt die Applikationserweiterung (1.4.1) das bekannte Formular (1.4.3) zum Login und blendet automatisch ein Overlay (1.4.2) zum Login mit dem erfindungsgemäßen Verfahren anstelle des Formulars ein, siehe Darstellung in der Mitte oben von Fig.8. Initialisiert der Nutzer den Login, so zeigt das Overlay (1.4.2) in der Darstellung unten mittig an, dass der Nutzer den Zugriff auf die geheimen Daten dieser Webanwendung mittels des Authentifikatorclients autorisieren soll – dies gilt zumindest für den Fall, dass der Applikationsclient nicht in offenen Zustand gesetzt ist, denn dann könnte er auch die sensiblen geheimen Daten ohne Autorisieren abrufen. Am Ende des Prozesses zeigt das Overlay (1.4.2) den erfolgreichen Login in die Webanwendung an und wird danach ausgeblendet.

In den unterschiedlichen Figuren sind hinsichtlich ihrer Funktion gleichwertige Teile stets mit denselben Bezugszeichen versehen, sodass diese in der Regel auch nur einmal beschrieben werden.

Die gezeigten Ausführungsformen der Erfindung sind jeweils beispielhaft und nicht beschränkend zu verstehen. Die Erfindung kann auch auf hiervon abweichende Weise ausgeführt werden. Daher sind alle gleichwertigen strukturellen Änderungen, die durch Anwendung der Beschreibung der Erfindung vorgenommen werden, in den Umfang der Patentanmeldung einzubeziehen.

BEZUGSZEICHENLISTE

(1.1)	Server
(1.2)	Authentifikatorclient
(1.3)	Applikationsclient
(1.4.1)	Applikationsclienterweiterung
(1.4.2)	Overlay
(1.4.3)	Formular
(1.5)	Push-Authentifikator
(1.6)	Backup-Authentifikator
(1.7)	Profil
(2.1.1)	Loginschlüsselpaar
(2.1.2)	Öffentlicher Loginschlüssel
(2.1.3)	Privater Loginschlüssel
(2.2.1)	Identifikationsschlüsselpaar
(2.2.2)	Öffentlicher Identifikationsschlüssel
(2.2.3)	Privater Identifikationsschlüssel
(2.2.4)	Signierprozess
(2.3.1)	Codierungsschlüsselpaar
(2.3.2)	Öffentlicher Codierungsschlüssel
(2.3.3)	Privater Codierungsschlüssel
(2.4.1)	Hochsicherheitsschlüsselpaar
(2.4.2)	Öffentlicher Hochsicherheitsschlüssel
(2.4.3)	Privater Hochsicherheitsschlüssel
(2.5.1)	Profilschlüsselpaar
(2.5.2)	Öffentlicher Profilschlüssel
(2.5.3)	Privater Profilschlüssel

(2.6.1) Profilhochsicherheitsschlüsselpaar

(2.6.2) Öffentlicher Profilhochsicherheitsschlüssel

(2.6.3) Privater Profilhochsicherheitsschlüssel

(2.7.1) Sitzungsschlüsselpaar

(2.7.2) Öffentlicher Sitzungsschlüssel

(2.7.3) Privater Sitzungsschlüssel

(2.7.4) Sitzung

(2.7.5) Sitzungstoken

(2.8.1) Austauschschlüsselpaar

(2.8.2) Öffentlicher Austauschschlüssel

(2.8.3) Privater Austauschschlüssel

(2.8.4) Austausch-QR-Code

(3.1) Symmetrischer Schlüssel

(3.2) Symmetrischer Hochsicherheitsschlüssel

(3.3.1) Tresor

(3.3.2) geheime Daten

(3.3.3) Sensibler Datenbereich

(3.3.4) Sensible Daten

(3.3.5) Metadatenbereich

(3.3.6) Metadaten

(3.3.7) Übergabe

(4.1) Zufälliger Authentifikatorstartwert

(4.2.1) Zufälliger Profilstartwert

(4.2.2) Zufälliger Profilhochsicherheitsstartwert

(4.3) Zufälliger Saltwert

(4.4) Zufälliger Profilsaltwert

PATENTANSPRÜCHE

1. **Computerimplementiertes Verfahren** für ein gesichertes Management von geheimen Daten (3.3.2) auf einem Server (1.1) sowie die gesicherte Nutzung der geheimen Daten (3.3.2), umfassend die Schritte:

S01) Initialisieren von zumindest einem Push-Authentifikator (1.5) auf einem initialen Authentifikatorclient (1.2), umfassend das

Generieren von zumindest zwei zum Push-Authentifikator (1.5) gehörigen asymmetrischen Schlüsselpaaren, einem Loginschlüsselpaar (2.1.1) aus einem privaten Loginschlüssel (2.1.3) und einem öffentlichen Loginschlüssel (2.1.2) und einem Codierungsschlüsselpaar (2.3.1) aus einem privaten Codierungsschlüssel (2.3.3) und einem öffentlichen Codierungsschlüssel (2.3.2) und

Speichern des öffentlichen Loginschlüssels (2.1.2) und des öffentlichen Codierungsschlüssels (2.3.2) des Push-Authentifikators (1.5) auf dem Server (1.1),

S02) Koppeln zumindest eines Applikationsclients (1.3) mit dem Server (1.1), umfassend das

Autorisieren durch den Authentifikatorclient (1.2) mittels verschlüsselten Austauschs des privaten Loginschlüssels (2.1.3) vom Push-Authentifikator (1.5) mit dem Applikationsclient (1.3),

Authentifizieren des Applikationsclients (1.3) gegen den Server (1.1) mittels des Loginschlüsselpaars (2.1.1) und

das Erstellen einer Sitzung (2.7.4) für den Applikationsclient (1.3) auf dem Server (1.1) durch das Hinterlegen eines öffentlichen Sitzungsschlüssels (2.7.2) eines vom Applikationsclient (1.3) generierten Sitzungsschlüsselpaars (2.7.1),

S03) Speichern bzw. Abrufen von geheimen Daten (3.3.2) durch den Applikationsclient (1.3) auf bzw. von einem auf dem Server (1.1) befindlichen Tresor (3.3.1), umfassend das

Verschlüsseln des Tresors (3.3.1) beim Speichern sowie Entschlüsseln des Tresors (3.3.1) beim Abrufen, wobei das Verschlüsseln oder Entschlüsseln durch einen auf dem Server (1.1) gespeicherten symmetrischen Schlüssel (3.1) erreicht wird,

Verschlüsseln des symmetrischen Schlüssels (3.1) mit dem öffentlichen Codierungsschlüssel (2.3.2) des Push-Authentifikators (1.5) und

Speichern bzw. Abrufen mittels Autorisierens eines verschlüsselten Abrufens des privaten Codierungsschlüssels (2.3.3) des Push-Authentifikators (1.5) durch den Authentifikatorclient (1.3) zum Entschlüsseln des symmetrischen Schlüssels (3.1),

dadurch gekennzeichnet dass

zumindest ein Profil (1.7) als Indirektionsebene zusätzlich zu dem Push-Authentifikator (1.5) eingesetzt wird, wobei der symmetrische Schlüssel (3.1) durch einen öffentlichen Profilschlüssel (2.5.2) eines Profilschlüsselpaars (2.5.1) des Profils (1.7) verschlüsselt wird,
wobei das Profil (1.7) durch den öffentlichen Codierungsschlüssel (2.3.2) des Push-Authentifikators (1.5) verschlüsselt wird.

2. Verfahren nach Anspruch 1, wobei der symmetrische Schlüssel (3.1) durch einen oder mehrere verschiedene öffentliche Profilschlüssel (2.5.2) oder öffentliche Codierungsschlüssel (2.3.2) von mehreren Profilen (1.7) oder Push-Authentifikatoren (1.5) verschlüsselt wird.

3. Verfahren nach einem der Ansprüche 1 bis 2, wobei zwei Profile sequenziell eingesetzt wobei werden, sodass zumindest ein erstes Profil (1.7) direkt durch den öffentlichen Profilschlüssel (2.5.2) des Profilschlüsselpaars (2.5.1) zumindest eines zweiten Profils (1.7) verschlüsselt wird.

4. Verfahren nach Anspruch 3, wobei das erste Profil (1.7) sämtliche zu einer Organisation gehörigen Tresore (3.3.1) verschlüsselt und wobei das zumindest zweite Profil (1.7) von den Push-Authentifikatoren (1.5) eines Administrators der Organisation verwaltet wird.

5. Verfahren nach Anspruch 4, wobei der Administrator der Organisation in der Lage ist, sämtliche Verschlüsselungen entsprechend Verfahrensschritt S03) seiner Organisation zu löschen, durch Platzhalter zu ersetzen sowie neu zu generieren.

6. Verfahren nach einem der Ansprüche 1 bis 5, wobei der Tresor (3.3.1) einen sensiblen Datenbereich (3.3.3) sowie einen Metadatenbereich (3.3.5) umfasst, wobei der sensible Datenbereich (3.3.3) zusätzlich durch einen auf dem Server gespeicherten symmetrischen Hochsicherheitsschlüssel (3.2) verschlüsselt ist, wobei der symmetrische Hochsicherheitsschlüssel (3.2) mittels eines öffentlichen Hochsicherheitsschlüssels (2.4.2) eines Hochsicherheitsschlüsselpaars (2.4.1) eines Push-Authentifikators (1.5) oder eines öffentlichen Profilhochsicherheitsschlüssels (2.6.2) eines Profilhochsicherheitsschlüsselpaars (2.6.1) eines Profils (1.7) verschlüsselt wird.

7. Verfahren nach Anspruch 6, wobei das Hochsicherheitsschlüsselpaar (2.4.1), das Profilhochsicherheitsschlüsselpaars (2.6.1) und das Loginschlüsselpaar (2.1.1) nicht auf einem dauerhaften Speicher von Clients, das heißt Authentifikatorclients (1.2) oder Applikationsclients (1.3), gespeichert werden kann.

8. Verfahren nach einem der Ansprüche 6 bis 7, wobei ein gekoppelter Applikationsclient (1.3) in einen geschlossenen Zustand versetzt wird, in welchem dieser Applikationsclient (1.3) keinen Zugriff auf den privaten Hochsicherheitsschlüssel (2.4.3) erhält.

9. Verfahren nach einem der Ansprüche 6 bis 8, wobei ein gekoppelter Applikationsclient (1.3) in einen offenen Zustand versetzt wird, in welchem der Applikationsclient (1.3) Zugriff auf den privaten Hochsicherheitsschlüssel (2.4.3) erhält.

10. Verfahren nach einem der Ansprüche 1 bis 9, wobei die Integrität der auf dem Server (1.1) gespeicherten öffentlichen Codierungsschlüssel (2.3.2), Hochsicherheitsschlüssel (2.4.2), Profilschlüssel (2.5.2), Profilhochsicherheitsschlüssel (2.6.2) sowie Sitzungsschlüssel (2.7.2) durch einen Signierprozess (2.2.4) mittels eines privaten Identifikationsschlüssels (2.2.3) eines Identifikationsschlüsselpaars (2.2.1) des zugehörigen Push-Authentifikators (1.5) oder Profils (1.7) sichergestellt wird.

11. Verfahren nach einem der Ansprüche 1 bis 10, wobei die Schlüsselpaare (2.1.1, 2.2.1, 2.3.1, 2.4.1, 2.5.1, 2.6.1) eines Push-Authentifikators (1.5) oder eines Profils (1.7) aus zumindest einem zufälligen Authentifikatorstartwert (4.1) bzw. Profilstartwert (4.2.1, 4.2.2) abgeleitet werden.

12. Verfahren nach Anspruch 11, wobei zum Ableiten der Schlüsselpaare (2.1.1, 2.2.1, 2.3.1, 2.4.1, 2.5.1, 2.6.1) eines Push-Authentifikators (1.5) oder eines Profils (1.7) zusätzlich ein zufälliger Saltwert (4.3) oder Profilsaltwert (4.4) verwendet und für den Push-Authentifikator (1.5) bzw. das Profil (1.7) auf dem Server (1.1) gespeichert wird.

13. Verfahren nach einem der Ansprüche 1 bis 12, wobei das Verfahren eine Recovery-Funktion umfasst, welche ein Nutzen eines neuen Authentifikatorclients (1.2) und aller bisherigen geheimen Daten (3.3.2) ohne den initialen Authentifikatorclient (1.2) ermöglicht.

14. Verfahren nach einem der Ansprüche 1 bis 13, wobei zusätzlich zu dem Applikationsclient (1.3) eine zugehörige Applikationsclienterweiterung (1.4.1) gekoppelt wird, welche Metadaten (3.3.6) der Sitzung (2.7.4) speichert und abruft sowie Overlays (1.4.2) für gespeicherte geheime Daten (3.3.2) auf Internetseiten einblendet.

15. Verfahren nach Anspruch 14, wobei die Applikationserweiterung (1.4.1) verschiedene Formulare (1.4.3) einer Internetseite automatisiert erkennt und auf Wunsch ausfüllen kann.

16. Verfahren nach einem der Ansprüche 1 bis 15, wobei das Autorisieren des Applikationsclients (1.3) sowie das Speichern und Abrufen von geheimen Daten (3.3.2) durch den Applikationsclient (1.3) mithilfe einer einfachen Wischgeste auf einem Smartphone als Authentifikatorsclient (1.2) durchgeführt werden.

17. Verfahren nach einem der Ansprüche 1 bis 16, wobei eine Sitzung (2.7.4) einen vom Server (1.1) generierten Sitzungstoken (2.7.5) umfasst, der es zumindest dem zur Sitzung (2.7.4) gehörenden Client erlaubt, Anfragen an den Server (1.1) zu senden.

18. Verfahren nach einem der Ansprüche 1 bis 17, wobei das Koppeln des Applikationsclients in Schritt S02) mittels Eingebens eines von dem Applikationsclient (1.3) erstellten und ausgegebenen öffentlichen Austauschschlüssels (2.8.2) eines Austauschschlüsselpaars (2.8.1) in den Authentifikatorclient (1.2) initialisiert wird und wobei der private Loginschlüssel (2.1.3) beim Austausch vom Authentifikatorclient (1.2) zum Applikationsclient (1.3) mit dem öffentlichen Austauschschlüssel (2.8.2) verschlüsselt wird.

19. Verfahren nach Anspruch 18, wobei der öffentliche Austauschschlüssel (2.8.2) in einem vom Applikationsclient (1.3) generierten und angezeigten Austausch-QR-Code (2.8.4) enthalten ist, welcher vom Authentifikatorclient (1.2) erfasst wird.

20. **Server** (1.1), umfassend zumindest eine Recheneinheit zur Ausführung von Computerprogrammen, Netzwerkanschlüsse zum Austausch von Daten zumindest mit den zum Verfahren nach einem der Ansprüche 1 bis 19 gehörenden Clients in einem Netzwerk, sowie zumindest ein Speichermedium zum Speichern von zumindest Computerprogrammen, Schlüsseln, Werten, Sitzungen und Tresoren.

21. **Computerprogrammprodukt A**, umfassend Befehle, die bei der Ausführung durch einen Authentifikatorclient diesen dazu veranlassen, das Verfahren nach einem der Ansprüche 1 bis 19 auszuführen.

22. **Computerprogrammprodukt B**, umfassend Befehle, die bei der Ausführung durch einen Applikationsclient diesen dazu veranlassen, das Verfahren nach einem der Ansprüche 1 bis 19 auszuführen.

23. **Computerprogrammprodukt C**, umfassend Befehle, die bei der Ausführung durch den Server diesen dazu veranlassen, das Verfahren nach einem der Ansprüche 1 bis 19 auszuführen.

24. **System**, umfassend einen Server (1.1) nach Anspruch 20, ein Computerprogrammprodukt A nach Anspruch 21, ein Computerprogrammprodukt B nach Anspruch 22 und ein Computerprogrammprodukt C nach Anspruch 23.

FIGUREN

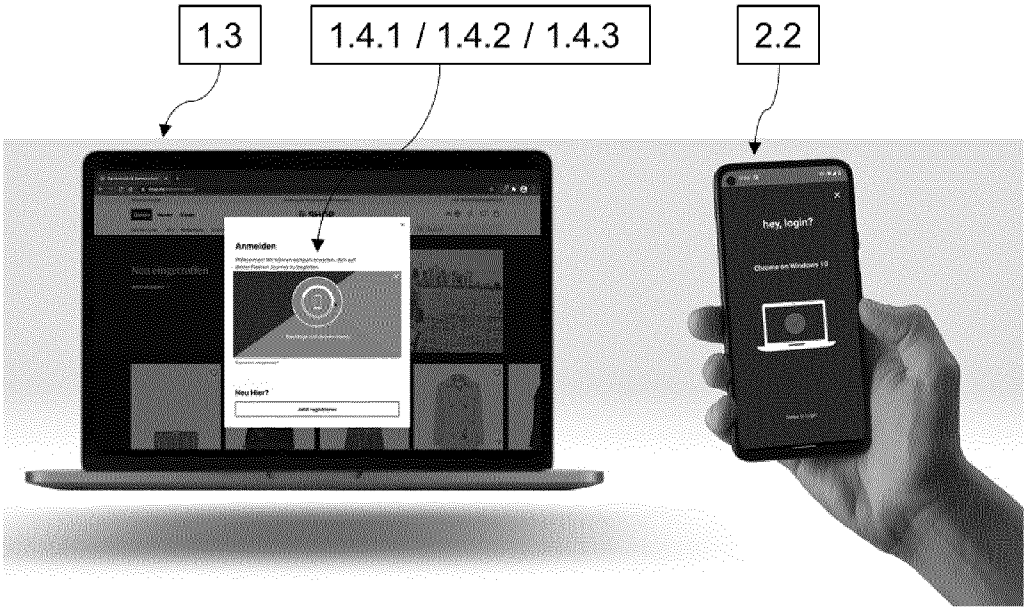

Fig. 1

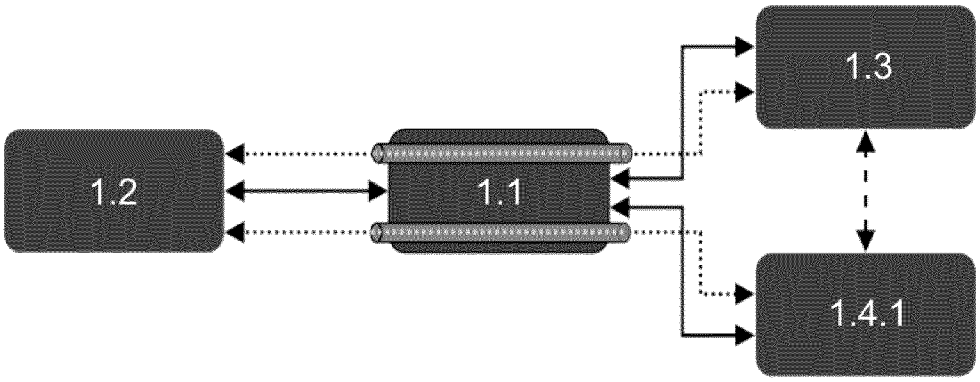

Fig. 2

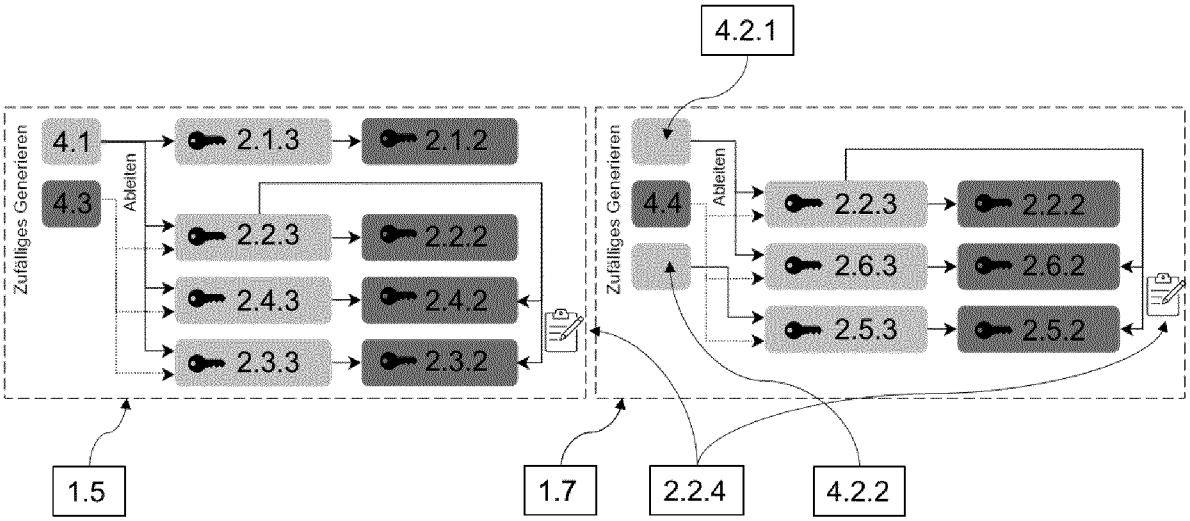
4.2.1
Zufälliges Generieren
4.1
4.3
Ableiten
2.1.3
2.1.2
2.2.3
2.2.2
2.4.3
2.4.2
2.3.3
2.3.2
Zufälliges Generieren
4.4
Ableiten
2.2.3
2.2.2
2.6.3
2.6.2
2.5.3
2.5.2
1.5
1.7
2.2.4
4.2.2

Fig. 3

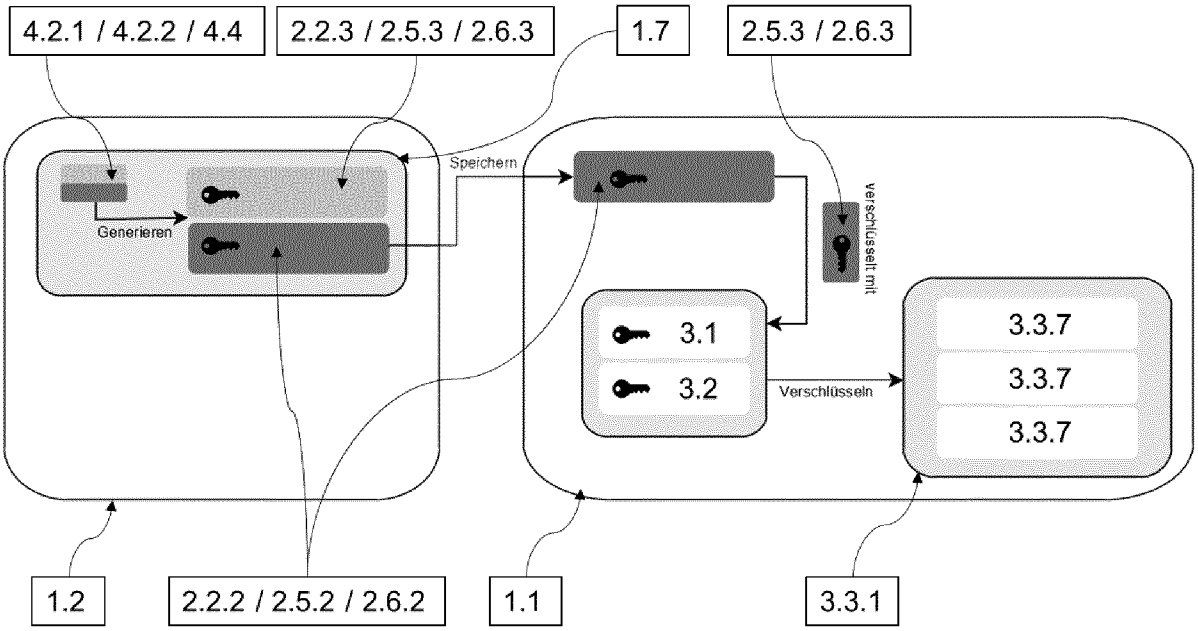
4.2.1 / 4.2.2 / 4.4
2.2.3 / 2.5.3 / 2.6.3
1.7
2.5.3 / 2.6.3
Generieren
Speichern
verschlüsselt mit
3.1
3.2
Verschlüsseln
3.3.7
3.3.7
3.3.7
1.2
2.2.2 / 2.5.2 / 2.6.2
1.1
3.3.1

Fig. 4

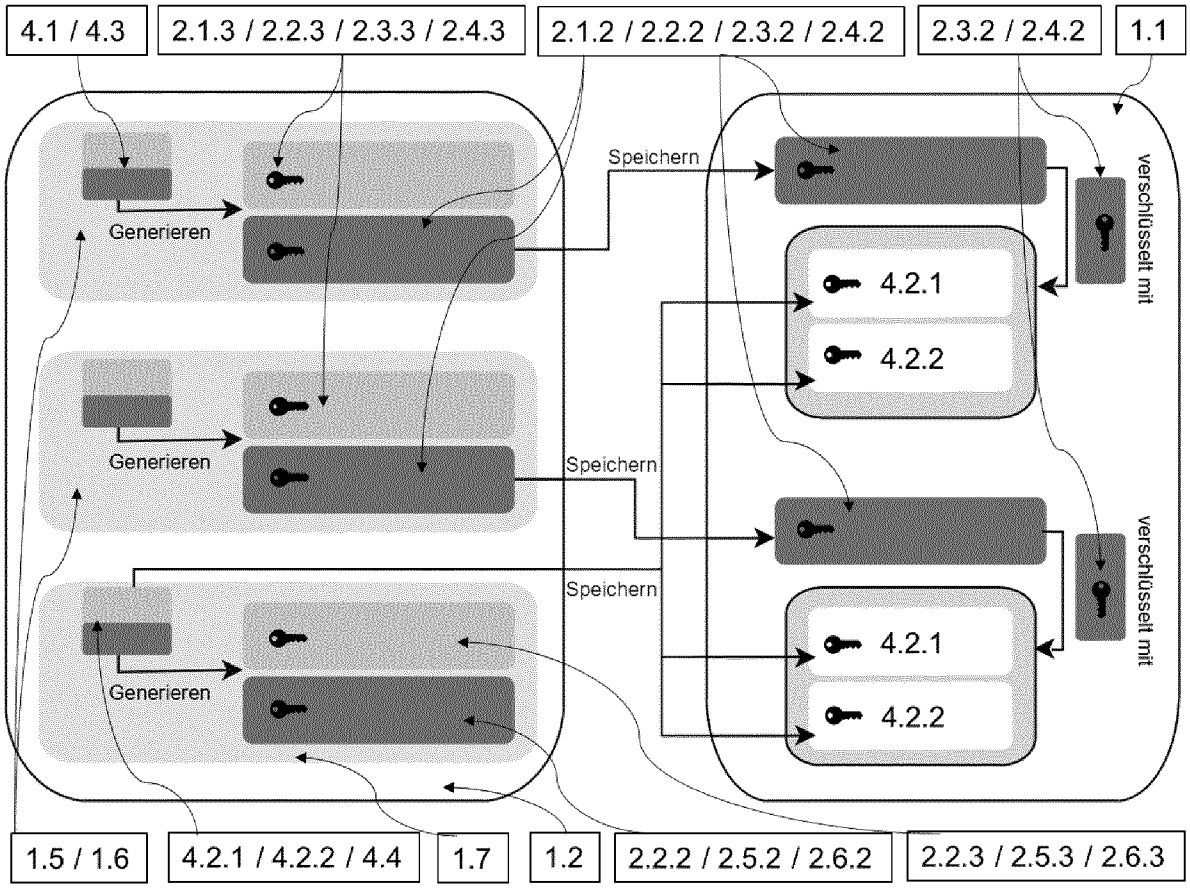
4.1 / 4.3
2.1.3 / 2.2.3 / 2.3.3 / 2.4.3
2.1.2 / 2.2.2 / 2.3.2 / 2.4.2
2.3.2 / 2.4.2
1.1
Generieren
Generieren
Generieren
Speichern
Speichern
Speichern
verschlüsselt mit
verschlüsselt mit
4.2.1
4.2.2
4.2.1
4.2.2
1.5 / 1.6
4.2.1 / 4.2.2 / 4.4
1.7
1.2
2.2.2 / 2.5.2 / 2.6.2
2.2.3 / 2.5.3 / 2.6.3

Fig. 5

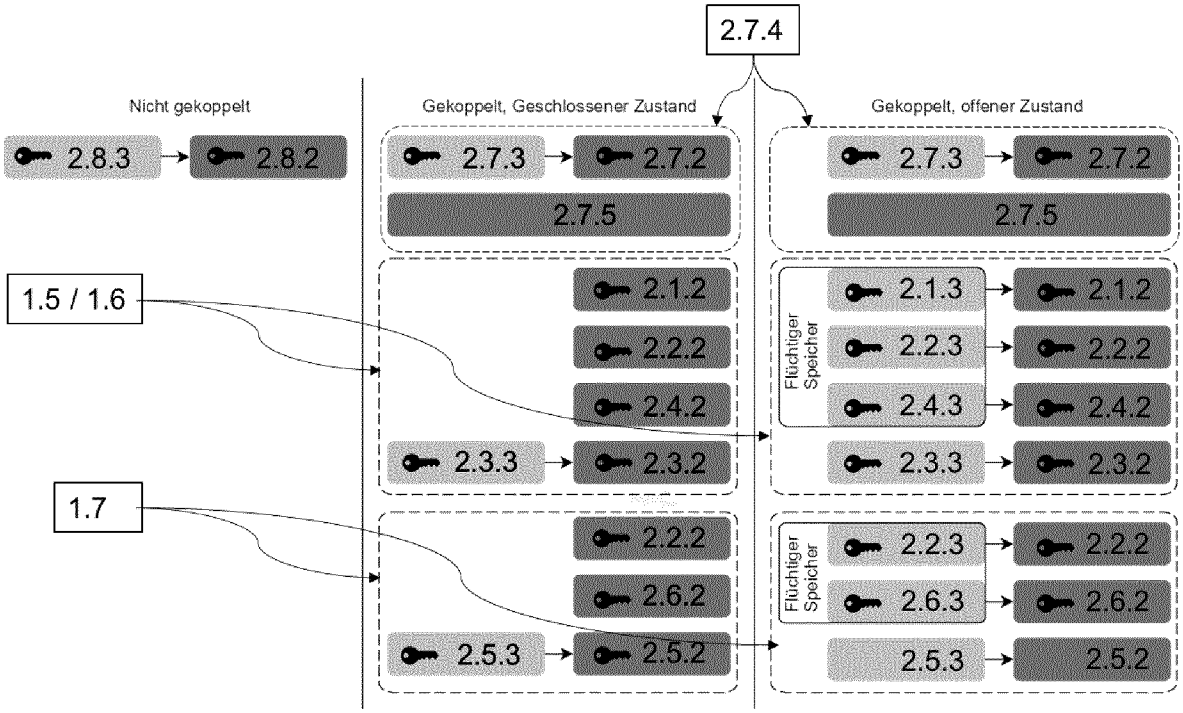
2.7.4
Nicht gekoppelt
Gekoppelt, Geschlossener Zustand
Gekoppelt, offener Zustand
2.8.3
2.8.2
2.7.3
2.7.2
2.7.5
2.7.3
2.7.2
2.7.5
1.5 / 1.6
2.1.2
2.2.2
2.4.2
2.3.3
2.3.2
Flüchtiger Speicher
2.1.3
2.2.3
2.4.3
2.1.2
2.2.2
2.4.2
2.3.3
2.3.2
1.7
2.2.2
2.6.2
2.5.3
2.5.2
Flüchtiger Speicher
2.2.3
2.6.3
2.2.2
2.6.2
2.5.3
2.5.2

Fig. 6

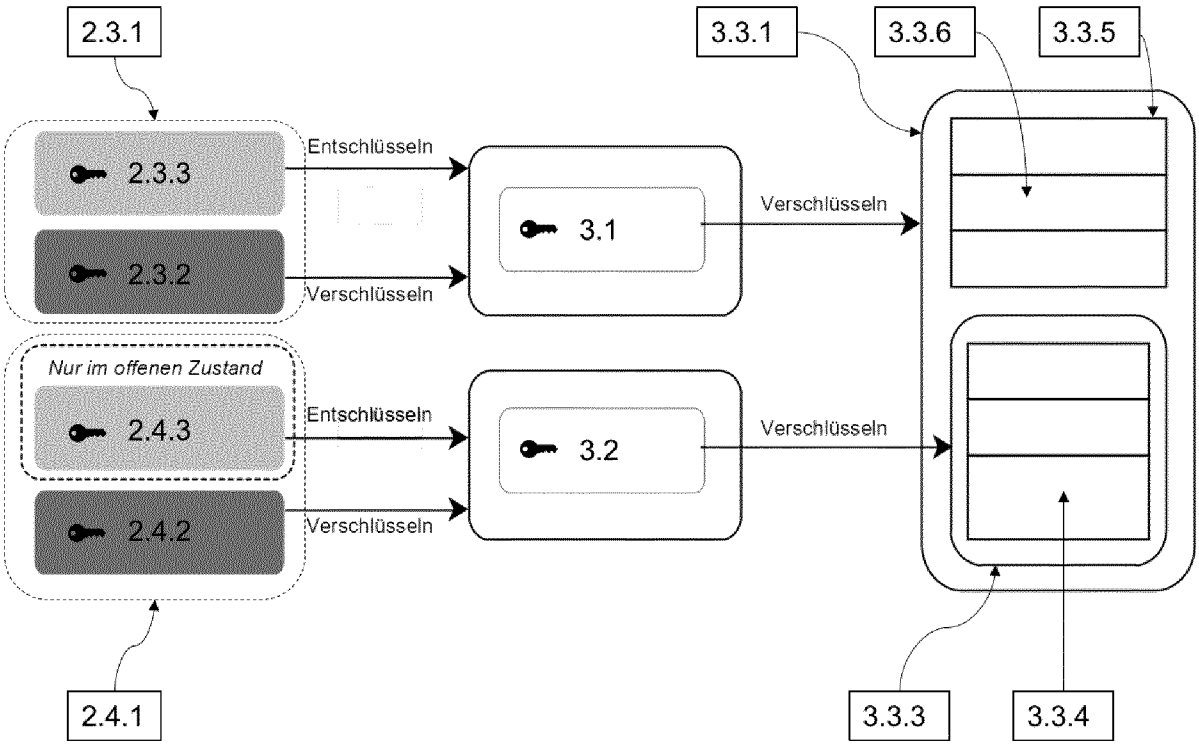
2.3.1
2.3.3
2.3.2
Entschlüsseln
Verschlüsseln
3.1
3.3.1
3.3.6
3.3.5
Verschlüsseln
Nur im offenen Zustand
2.4.3
2.4.2
Entschlüsseln
Verschlüsseln
3.2
Verschlüsseln
2.4.1
3.3.3
3.3.4

Fig. 7

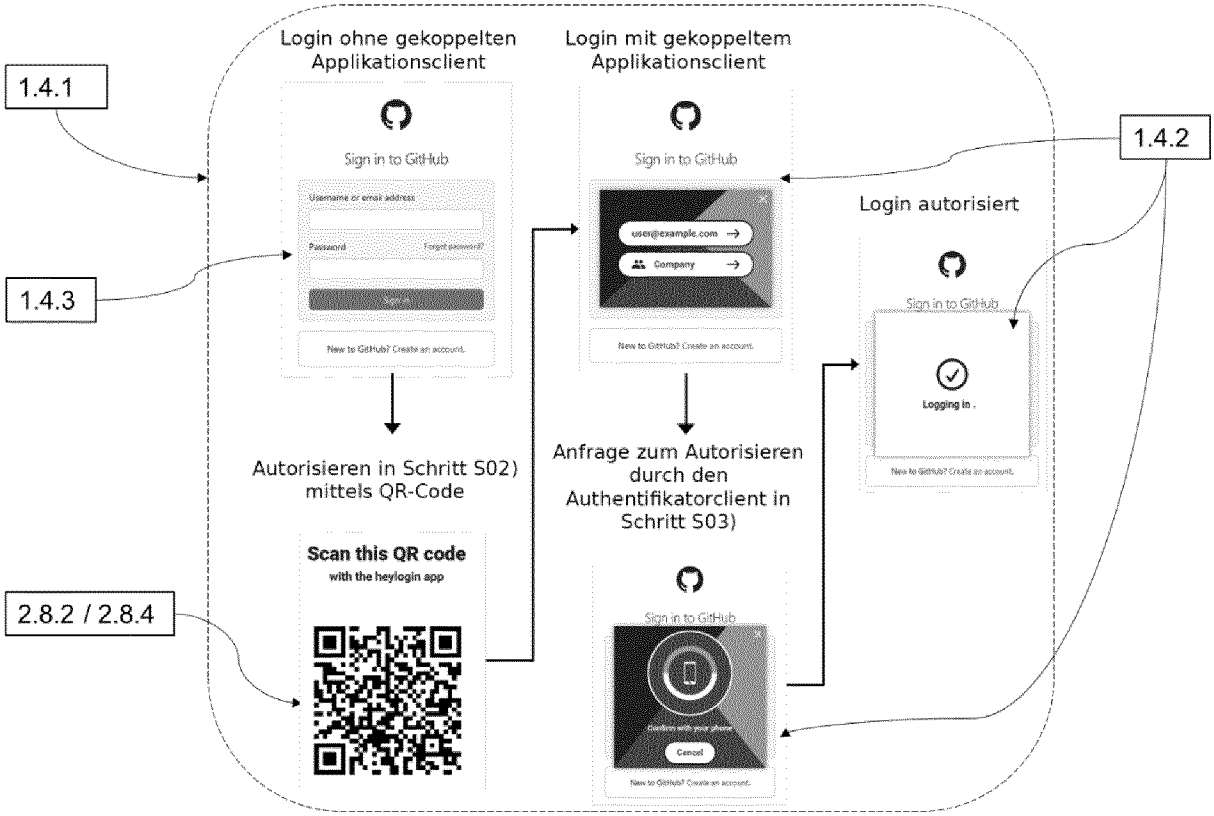
Login ohne gekoppelten Applikationsclient
Login mit gekoppeltem Applikationsclient
1.4.1
1.4.3
1.4.2
Sign in to GitHub
Login autorisiert
Autorisieren in Schritt S02) mittels QR-Code
Anfrage zum Autorisieren durch den Authentifikatorclient in Schritt S03)
Scan this QR code
with the heylogin app
2.8.2 / 2.8.4
Logging in .
Cancel

Fig. 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/EP2024/058675

A. CLASSIFICATION OF SUBJECT MATTER

H04L 9/32(2006.01)i; ***H04W 12/06***(2021.01)i; ***G06F 21/31***(2013.01)i; ***G06F 21/34***(2013.01)i; ***G06F 21/45***(2013.01)i; ***H04L 9/40***(2022.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04L; G06F; H04W

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Heylogin Gmbh. "heylogin White Paper" 06 April 2022 (2022-04-06), pages 1-16, Retrieved from the Internet: https://web.archive.org/web/20220528040652/https://www.heylogin.com/files/heylogin-security-whitepaper-en-v1-4.pdf [retrieved on 2023-09-14] XP093084091 the whole document	1-24
A	US 9003015 B2 (SEIFERT MICHAEL [DK]; KOSTENKO DMYTRO [DK]; SITECORE AS [DK]) 07 April 2015 (2015-04-07) abstract	1-24
A	US 2010169368 A1 (NEILL RICHARD W [US]) 01 July 2010 (2010-07-01) abstract	1-24

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
26 June 2024	**08 July 2024**

Name and mailing address of the ISA/EP	Authorized officer
European Patent Office **p.b. 5818, Patentlaan 2, 2280 HV Rijswijk** **Netherlands (Kingdom of the)** Telephone No. **(+31-70)340-2040** Facsimile No. **(+31-70)340-3016**	**Wolters, Robert** Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/EP2024/058675

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
US 9003015 B2	07 April 2015	DK 2469427 T3	16 October 2017
		EP 2469427 A1	27 June 2012
		US 2012158950 A1	21 June 2012
US 2010169368 A1	01 July 2010	US 2006136384 A1	22 June 2006
		US 2010169368 A1	01 July 2010
		US 2014074915 A1	13 March 2014
		US 2016294741 A1	06 October 2016

INTERNATIONALER RECHERCHENBERICHT

Internationales Aktenzeichen
PCT/EP2024/058675

A. KLASSIFIZIERUNG DES ANMELDUNGSGEGENSTANDES
INV. H04L9/32 H04W12/06 G06F21/31 G06F21/34 G06F21/45
H04L9/40
ADD.
Nach der Internationalen Patentklassifikation (IPC) oder nach der nationalen Klassifikation und der IPC

B. RECHERCHIERTE GEBIETE
Recherchierter Mindestprüfstoff (Klassifikationssystem und Klassifikationssymbole)
H04L G06F H04W

Recherchierte, aber nicht zum Mindestprüfstoff gehörende Veröffentlichungen, soweit diese unter die recherchierten Gebiete fallen

Während der internationalen Recherche konsultierte elektronische Datenbank (Name der Datenbank und evtl. verwendete Suchbegriffe)
EPO-Internal

C. ALS WESENTLICH ANGESEHENE UNTERLAGEN

Kategorie*	Bezeichnung der Veröffentlichung, soweit erforderlich unter Angabe der in Betracht kommenden Teile	Betr. Anspruch Nr.
A	Heylogin Gmbh: "heylogin White Paper", , 6. April 2022 (2022-04-06), Seiten 1-16, XP093084091, Gefunden im Internet: URL:https://web.archive.org/web/20220528040652/https://www.heylogin.com/files/heylogin-security-whitepaper-en-v1-4.pdf [gefunden am 2023-09-14] das ganze Dokument	1-24
A	US 9 003 015 B2 (SEIFERT MICHAEL [DK]; KOSTENKO DMYTRO [DK]; SITECORE AS [DK]) 7. April 2015 (2015-04-07) Zusammenfassung	1-24
A	US 2010/169368 A1 (NEILL RICHARD W [US]) 1. Juli 2010 (2010-07-01) Zusammenfassung	1-24

☐ Weitere Veröffentlichungen sind der Fortsetzung von Feld C zu entnehmen ☒ Siehe Anhang Patentfamilie

* Besondere Kategorien von angegebenen Veröffentlichungen :
"A" Veröffentlichung, die den allgemeinen Stand der Technik definiert, aber nicht als besonders bedeutsam anzusehen ist
"E" frühere Anmeldung oder Patent, die bzw. das jedoch erst am oder nach dem internationalen Anmeldedatum veröffentlicht worden ist
"L" Veröffentlichung, die geeignet ist, einen Prioritätsanspruch zweifelhaft erscheinen zu lassen, oder durch die das Veröffentlichungsdatum einer anderen im Recherchenbericht genannten Veröffentlichung belegt werden soll oder die aus einem anderen besonderen Grund angegeben ist (wie ausgeführt)
"O" Veröffentlichung, die sich auf eine mündliche Offenbarung, eine Benutzung, eine Ausstellung oder andere Maßnahmen bezieht
"P" Veröffentlichung, die vor dem internationalen Anmeldedatum, aber nach dem beanspruchten Prioritätsdatum veröffentlicht worden ist
"T" Spätere Veröffentlichung, die nach dem internationalen Anmeldedatum oder dem Prioritätsdatum veröffentlicht worden ist und mit der Anmeldung nicht kollidiert, sondern nur zum Verständnis des der Erfindung zugrundeliegenden Prinzips oder der ihr zugrundeliegenden Theorie angegeben ist
"X" Veröffentlichung von besonderer Bedeutung;; die beanspruchte Erfindung kann allein aufgrund dieser Veröffentlichung nicht als neu oder auf erfinderischer Tätigkeit beruhend betrachtet werden
"Y" Veröffentlichung von besonderer Bedeutung;; die beanspruchte Erfindung kann nicht als auf erfinderischer Tätigkeit beruhend betrachtet werden, wenn die Veröffentlichung mit einer oder mehreren Veröffentlichungen dieser Kategorie in Verbindung gebracht wird und diese Verbindung für einen Fachmann naheliegend ist
"&" Veröffentlichung, die Mitglied derselben Patentfamilie ist

Datum des Abschlusses der internationalen Recherche	Absendedatum des internationalen Recherchenberichts
26. Juni 2024	08/07/2024

Name und Postanschrift der Internationalen Recherchenbehörde	Bevollmächtigter Bediensteter
Europäisches Patentamt, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Wolters, Robert

1

INTERNATIONALER RECHERCHENBERICHT

Angaben zu Veröffentlichungen, die zur selben Patentfamilie gehören

Internationales Aktenzeichen

PCT/EP2024/058675

Im Recherchenbericht angeführtes Patentdokument	Datum der Veröffentlichung	Mitglied(er) der Patentfamilie	Datum der Veröffentlichung
US 9003015 B2	07-04-2015	DK 2469427 T3	16-10-2017
		EP 2469427 A1	27-06-2012
		US 2012158950 A1	21-06-2012
US 2010169368 A1	01-07-2010	US 2006136384 A1	22-06-2006
		US 2010169368 A1	01-07-2010
		US 2014074915 A1	13-03-2014
		US 2016294741 A1	06-10-2016