JUSTUS-LIEBIG-UNIVERSITÄT GIESSEN
PROFESSUR BWL – WIRTSCHAFTSINFORMATIK
UNIV.-PROF. DR. AXEL SCHWICKERT

Fabian, S.; Schwickert, Axel

OpenID Connect und FIDO2 – Ein Vergleich

ARBEITSPAPIERE WIRTSCHAFTSINFORMATIK

Nr. 8 / 2021
ISSN 1613-6667

Arbeitspapiere WI Nr. 8 / 2021

Autoren: Fabian, S.; Schwickert, Axel

Titel: OpenID Connect und FIDO2 – Ein Vergleich

Zitation: Fabian, S.; Schwickert, Axel: OpenID Connect und FIDO2 – Ein Vergleich, in: Arbeitspapiere WI, Nr. 8/2021, Hrsg.: Professur BWL – Wirtschaftsinformatik, Justus-Liebig-Universität Gießen 2021, 87 Seiten, ISSN 1613-6667.

Kurzfassung: Die derzeit weit verbreitete Authentifizierungsmethode OpenID Connect hat zwei wesentliche Eigenschaften. Zum einen müssen die Benutzer-Daten bei einem Authentifizierungsvorgang zwischen dem Web-Browser des Users und dem Web-Server der Web-Anwendung über Netzwerkleitungen des offenen Internet übertragen werden. Zum anderen müssen die Benutzer-Daten auf Seiten des Benutzers und des Servers gespeichert werden. Die Benutzer-Daten müssen also beim User, beim Server und auf dem Weg dazwischen geschützt werden. Genau diese beiden Eigenschaften sind die Ursache für die allseits bekannten Sicherheitsprobleme von geschützten Web-Anwendungen im offenen Internet. FIDO2 (Fast Identity Online) ist eine neuartige Authentifizierungsmethode, die diese Sicherheitsprobleme wesentlich verringern kann. Mit FIDO2 müssen keine kritischen Benutzer-Daten auf der Server-Seite gespeichert oder über Netzwerkleitungen des offenen Internet übertragen werden. Die Sicherheit der Benutzer-Daten auf der User-Seite wird durch spezielle technische Vorrichtungen kategorial erhöht. Ziel der vorliegenden Arbeit ist es, OpenID Connect und FIDO2 vergleichend zu untersuchen. Zu jeder Authentifizierungsmethode wird ein statisches Modell der Akteure und ein dynamisches Modell der Abläufe mit den jeweiligen spezifischen Eigenschaften beschrieben. Die beiden Authentifizierungsmethoden werden abschließend mit allen Eigenschaften vergleichend nebeneinander gestellt.

Schlüsselwörter: OpenID Connect, OIDC, FIDO2, OAuth, Authentifizierungsmethode, Auth-Server, Access-Token, 1-Faktor, 2-Faktor, WebAuthn, Authentifikator, Trusted Platform Module, Secure Element, CATP2, Challenge Response, Signatur, Kryptographie, asymmetrische Verschlüsselung

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

API ..... Application Programming Interface
Auth-Server ..... Authentifizierungs-Server
CTAP ..... Client to Authenticator Protocol
DE-L ..... Digital Enterprise Labs
FIDO ..... Fast Identity Online
FRR ..... False-Reject-Rate
HTTP ..... Hypertext Transfer Protocol
HTTPS ..... Hypertext Transfer Protocol Secure
H2M ..... Human-to-Machine
HW ..... Hardware
IAM ..... Identity- & Access-Management
ID ..... Identifikationsnummer
Inc ..... Incorporated
IETF ..... Internet Engineering Task Force
IT-System ..... Informationstechnologie-System
JSON ..... JavaScript Object Notation
JWT ..... JSON Web Token
M2M ..... Machine-to-Machine
MitM ..... Man-in-the-Middle
OAuth ..... Open Authorization
oHG ..... offene Handelsgesellschaft
OIDC ..... OpenID Connect
OS ..... Operating System
PIN ..... Personal Identification Number
SAML ..... Security Assertion Markup Language
SMS ..... Short Message Service
SW ..... Software
TAN..... Transaktionsnummer
TLS ..... Transport Layer Security
U2F ..... Universal 2nd Factor
UAF ..... Universal Authentication Factor
URI ..... Uniform Resource Identifier
USB ..... Universal Serial Bus
W3C ..... World Wide Web Consortium

1 Problemstellung, Ziel und Aufbau

Web-Anwendungen sind Software-Systeme, die auf einem Web-Server betrieben werden und über das Internet per Web-Browser bedienbar sind. Web-Anwendungen haben eine Benutzeroberfläche, die mit der Logik auf dem Web-Server interagiert.[1] In der Regel erfordern Web-Anwendungen die Authentifizierung des Benutzers (User). Der User authentisiert sich, indem er seine Benutzer-Daten angibt. Als Authentifizierung wird der Vorgang bezeichnet, bei dem verifiziert wird, ob es sich bei einer Person oder einer Repräsentation einer Person (wie z. B. Benutzer-Daten) um eine bekannte, echte Identität handelt.[2] Eine solche Authentifizierung kann mittels verschiedener Methoden und jeweils zugehöriger Software-Instrumente umgesetzt werden.

Die Benutzer-Daten einer Person bestehen i. d. R. aus einem Benutzer-Namen (User-Name) und einem zugehörigen Benutzer-Passwort (User-Password). Diese Benutzer-Daten repräsentieren ein Benutzer-Konto (User-Account) für eine Web-Anwendung. Viele Web-Anwendungen nutzen die derzeit üblichen Authentifizierungsmethoden, bei denen die vorgenannten Benutzer-Daten als einziger „Faktor“ zur Authentifizierung eines Benutzers eingesetzt werden. Dafür gibt der Benutzer seine Benutzer-Daten in einen Login-Bildschirm im Web-Browser ein und der Server der Web-Anwendung prüft, ob für diese Benutzer-Daten der Zugriff auf die Web-Anwendung erlaubt ist oder nicht.[3]

Um die Zuverlässigkeit einer Benutzer-Authentifizierung zu erhöhen, setzen immer mehr Authentifizierungsmethoden einen zweiten „Faktor“ ein. Nachdem ein Benutzer seine Benutzer-Daten erfasst hat, verlangt der Web-Server vom Benutzer die Eingabe weiterer Identifikationsangaben. Dazu schickt der Web-Server z. B. eine TAN (Transaktionsnummer) per SMS-Nachricht (Short Message Service) an das Smartphone des Benutzers. Der Benutzer muss diese TAN dann ebenfalls im Login-Bildschirm eingeben, um Zugang zur Web-Anwendung zu erhalten.[4]

---

1 Vgl. Fischer, Manuel et al.: Entwicklung erfolgreicher Web-Anwendungen – Leitfaden Webentwicklung 2015, Bitkom e.V. (Hrsg.), 2015, https://www.bitkom.org/sites/default/files/file/import/LF-Web-Anwendungen-150910-1.pdf, abgerufen am 23.12.2020, S. 5f.

2 Vgl. Balzert, Helmut: Lehrbuch der Softwaretechnik: Entwurf, Implementierung, Installation und Betrieb, Spektrum Akademischer Verlag GmbH (Hrsg.), Heidelberg, 2011, S. 154.

3 Vgl. Thiel, Barbara (Landesbeauftragte für den Datenschutz Niedersachsen) (Hrsg.): Handlungsempfehlung sichere Authentifizierung, 05. Februar 2020, https://lfd.niedersachsen.de/download/61775, abgerufen am 28.12.2020, S. 2, 5.

4 Vgl. O. V.: Zwei-Faktor-Authentisierung für höhere Sicherheit, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/DigitaleGesellschaft/Online-Banking/Zwei_Faktor_Authentisierung/Zwei-Faktor-Authentisierung_node.html, 29.12.2020.

Die Zuverlässigkeit einer Benutzer-Authentifizierung für eine Web-Anwendung kann wesentlich erhöht werden, indem ein dritter Akteur die Authentifizierung eines Benutzers (1. Akteur) beim Web-Server (2. Akteur) bestätigt. Dazu überprüft der dritte Akteur als vertrauenswürdige Instanz die Gültigkeit der Benutzer-Daten (1. und/oder 2. Faktor) für den Zugriff auf eine Web-Anwendung. Aktuell bieten sich z. B. Apple („mit Apple anmelden“)[5] oder Google („weiter mit Google“)[6] als vertrauenswürdige Instanzen (im Folgenden „Auth Server“ genannt) an. Ein Auth-Server muss dazu vorab natürlich die gültigen Benutzer-Daten kennen, um seine Authentifizierungsaufgabe erfüllen zu können.[7]

Die vorgenannten, derzeit weit verbreiteten Authentifizierungsmethoden haben zwei wesentliche Eigenschaften gemein. Zum einen müssen die Benutzer-Daten bei einem Authentifizierungsvorgang zwischen dem Web-Browser des Users und dem Web-Server der Web-Anwendung über Netzwerkleitungen des offenen Internet übertragen werden. Zum anderen müssen die Benutzer-Daten auf Seiten des Benutzers und des Servers gespeichert werden.[8] Die Benutzer-Daten müssen also beim User, beim Server und auf dem Weg dazwischen geschützt werden.

Genau diese beiden Eigenschaften sind die Ursache für die allseits bekannten Sicherheitsprobleme von geschützten Web-Anwendungen im offenen Internet. Per Phishing und Malware werden Benutzer-Daten von User-Rechnern abgegriffen.[9] Durch gehackte oder kompromittierte Server können ganze Benutzer-Datenbanken in falsche Hände geraten.[10] Das Abhören von Telekommunikation durch Unbefugte ist heute gang und gäbe.[11]

5 Vgl. Apple Inc. (Hrsg.): Was ist "Mit Apple anmelden"?, https://support.apple.com/de-de/HT210318, abgerufen am 29.12.2020.

6 Vgl. Google LLC (Hrsg.): Mit Ihrem Google-Konto können Sie sich auch bei anderen Apps oder Diensten anmelden, https://support.google.com/accounts/answer/112802?co=GENIE.Platform%3DDesktop&hl=de&oco=0, abgerufen am 29.12.2020.

7 Vgl. Peyrott, Sebastian: Was ist Single Sign-on Authentication, und wie funktioniert sie?, auth0.com (Hrsg.), 05. März 2019, https://auth0.com/blog/de-what-is-and-how-does-single-sign-on-work/, abgerufen am 28.12.2020.

8 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, in: C’t 19/18, Heise Medien GmbH & Co. KG (Hrsg.), S. 30.

9 Vgl. O. V.: Phishing E-Mails – Passwortdiebstahl durch Phishing, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/Risiken/SpamPhishingCo/Phishing/phishing_node.html, abgerufen am 29.12.2020.

10 Vgl. Ries, Uli: Sie wurden gehackt! – Datenklau bei Yahoo, Dropbox & Co. betrifft Milliarden, in: C’t 23/16, Heise Medien GmbH & Co. KG (Hrsg.), S. 78.

11 Vgl. Holland, Martin: NSA-Überwachungsskandal: Von NSA, GCHQ, BND, PRISM, Tempora, XKeyScore und dem Supergrundrecht – was bisher geschah, Heise Medien GmbH & Co. KG (Hrsg.), 19. September 2013, https://www.heise.de/newsticker/meldung/NSA-Ueberwachungsskandal-Von-NSA-GCHQ-BND-PRISM-Tempora-XKeyScore-und-dem-Supergrundrecht-was-bisher-geschah-1958399.html, abgerufen am 30.12.2020.

Auch eine Verschlüsselung aller kritischen Daten kann nur bedingt Abhilfe schaffen.[12] FIDO2 (Fast Identity Online) ist eine neuartige Authentifizierungsmethode, die diese Sicherheitsprobleme wesentlich verringern kann. Mit FIDO2 müssen keine kritischen Benutzer-Daten auf der Server-Seite gespeichert oder über Netzwerkleitungen des offenen Internet übertragen werden. Die Sicherheit der Benutzer-Daten auf der User-Seite wird durch spezielle technische Vorrichtungen kategorial erhöht.[13]

Ziel der vorliegenden Arbeit ist es, die vorab geschilderten derzeit gängigen Authentifizierungsmethoden und FIDO2 vergleichend zu untersuchen. Als Vertreter der derzeit gängigen Authentifizierungsmethoden wird OpenID Connect (OIDC) herangezogen. OIDC funktioniert im Kern nach dem oben skizzierten Prinzip der Authentifizierung mit einem Faktor.

Die Eckpunkte der vergleichenden Untersuchung werden in Kapitel 2 dargelegt. Nach einer Abgrenzung des Untersuchungsbereichs (Kapitel 2.1) und der Untersuchungsobjekte (Kapitel 2.2) wird in Kapitel 2.3 der Kriterien-Katalog beschrieben, mit dem die beiden Authentifizierungsmethoden bewertet und verglichen werden. Die Eigenschaften werden in die Kategorien Sicherheitsniveau, Usability und Administrationsaufwand eingeordnet.[14] In Kapitel 2.4 werden die typischen Einsatzszenarien skizziert, in denen die beiden Authentifizierungsmethoden angewendet und verglichen werden.

In den Kapiteln 3 und 4 werden die Authentifizierungsmethoden OpenID Connect und FIDO2 im Detail beschrieben. Zu jeder Authentifizierungsmethode wird ein statisches Modell der Akteure und ein dynamisches Modell der Abläufe mit den jeweiligen spezifischen Eigenschaften dargestellt. Alle Eigenschaften einer Authentifizierungsmethode werden am Ende des jeweiligen Kapitels in einer Übersicht zusammengeführt. Kapitel 5 stellt die beiden Authentifizierungsmethoden mit allen Eigenschaften nebeneinander.

12 Vgl. O. V.: Was beim Einsatz von Verschlüsselung zu beachten ist, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/Empfehlungen/Verschluesselung/Datenverschluesselung/Grundlagen/Wissenswertes/wissenswertes_node.html, abgerufen am 29.12.2020.

13 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 30ff.

14 Im magischen Dreieck der BWL (Qualität, Zeit, Kosten) stehen Sicherheitsniveau und Usability für die Qualität. Der Administrationsaufwand steht für Zeit und Kosten. Siehe Gerboth, Thomas: Das Magische Dreieck, in: Controlling, Heft 7, Verlag C.H.BECK oHG (Hrsg.), Juli 2002, S. 417.

2 Eckpunkte der Untersuchung

2.1 Untersuchungsbereich

Der Untersuchungsbereich der vorliegenden Arbeit umfasst Web-Anwendungen als Ausprägung verteilter IT-Systeme. IT-Systeme können in „monolithische IT-Systeme“ und „verteilte IT-Systeme“ unterteilt werden. Die monolithischen IT-Systeme zentralisieren die Funktionalität des Systems in einem einzelnen Rechner. Im Gegensatz dazu bestehen verteilte IT-Systeme aus gekoppelten, (oft physisch) dezentralen und funktional eigenständigen Komponenten.[15]

Die Kopplung geschieht dabei durch eine sog. „Middleware“, die die eigenständigen Komponenten auch über Netzwerkgrenzen hinweg untereinander verbindet. Über standardisierte Funktionen und Datentypen der Middleware kann eine Systemkomponente die Ressourcen einer anderen Systemkomponente nutzen.[16] In Abbildung 1 ist der Aufbau von monolithischen und verteilten IT-Systemen dargestellt.

Die Kopplung verteilter Komponenten funktioniert auf Grundlage des in Abbildung 2 dargestellten „Client-Server-Konzepts“. Dieses Konzept setzt voraus, dass es mindestens zwei verteilte Komponenten in einem System gibt. Mindestens eine Komponente („Server“) bietet über Schnittstellen Dienste an und mindestens eine Komponente („Client“) fragt Dienste von Servern ab. Die Kommunikation basiert auf Nachrichten.[17] Der Client schickt einen „Request“ („Anfrage-Nachricht“) an den Server, der die geforderte Ressource oder den geforderten Dienst eindeutig beschreibt. Der Server verarbeitet den Request und antwortet mit einer passenden „Response“ („Antwort-Nachricht“), die die geforderte Ressource oder den geforderten Dienst liefert. Bei verteilten Web-Anwendungen läuft ein solcher Nachrichtenaustausch wie in Abbildung 2 visualisiert über das offene Internet ab.

15 Vgl. Fink, Andreas: Verteiltes IT-System, in: Enzyklopädie der Wirtschaftsinformatik, Universität Potsdam – Lehrstuhl für Wirtschaftsinformatik – Norbert Gronau (Hrsg.), 31. Oktober 2012, https://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/is-management/Systementwicklung/Softwarearchitektur/Middleware/index.html, abgerufen am 26.01.2021.

16 Vgl. Karcher, Andreas: Middleware, in: Enzyklopädie der Wirtschaftsinformatik, Universität Potsdam – Lehrstuhl für Wirtschaftsinformatik – Norbert Gronau (Hrsg.), 29. November 2016, https://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/is-management/Systementwicklung/Softwarearchitektur/Middleware/index.html, abgerufen am 26.01.2021.

17 Vgl. Fettke, Peter: Client-Server-Architektur, in: Enzyklopädie der Wirtschaftsinformatik, Universität Potsdam – Lehrstuhl für Wirtschaftsinformatik – Norbert Gronau (Hrsg.), 23. September 2016, https://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/is-management/Systementwicklung/Softwarearchitektur/Architekturparadigmen/Client-Server-Architektur/index.html, abgerufen am 26.01.2021.

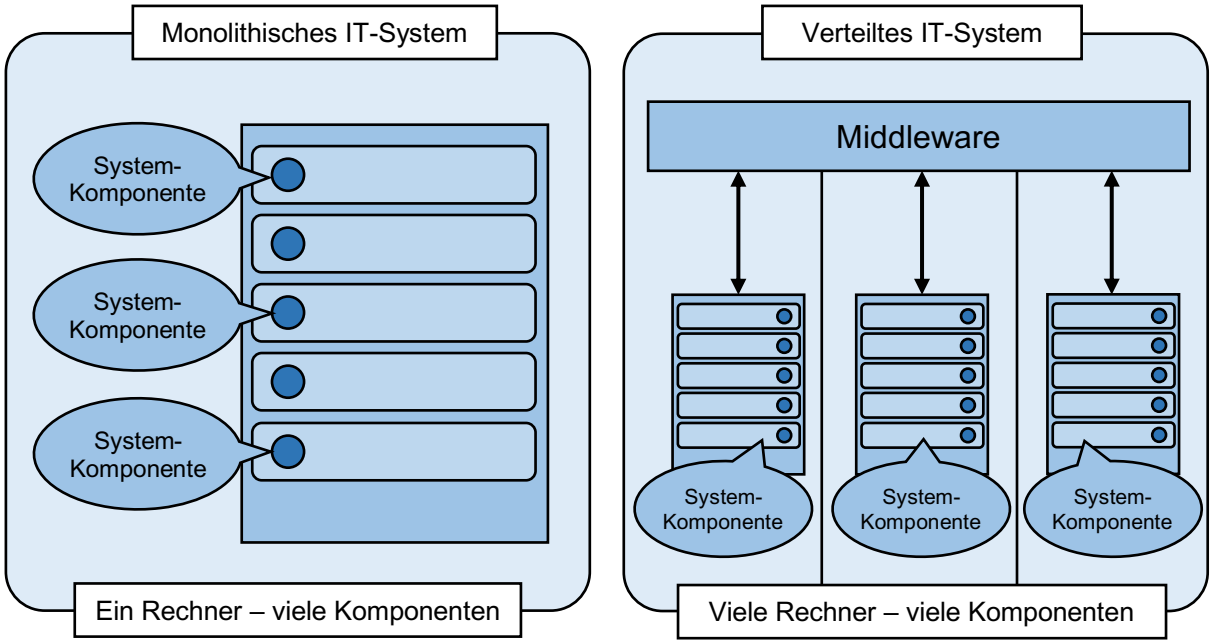

Abb. 1: Aufbau von monolithischen und verteilten IT-Systemen (eigene Abbildung)

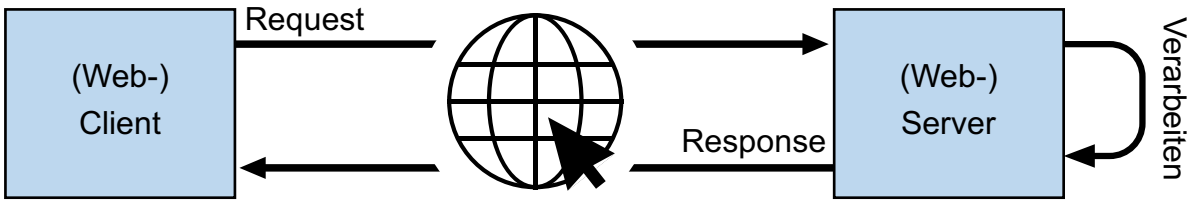

Abb. 2: Das Client-Server-Konzept (eigene Abbildung)

Ein Beispiel für den Einsatz des Client-Server-Konzepts ist das Aufrufen einer Web-Seite. Ein User gibt zuerst eine URI (Uniform Resource Identifier) in einem Web-Browser (Client) ein. Eine URI ist eine eindeutige Adresse für eine Ressource im Internet, die einen Server und den Endpunkt auf diesem Server spezifiziert. Der Endpunkt kann dabei als Server-interne Adresse verstanden werden. Über eine URI lässt sich ein bestimmter Dienst oder eine bestimmte Ressource abfragen. Eine Untergruppe der URIs sind die URLs (Uniform Resource Locator). Eine URL spezifiziert neben der Position der Ressource auch die Art, wie die Ressource zu erreichen ist (z. B. das Protokoll).[18] Wenn der User einen URI auf der Tastatur eingibt und „Enter“ drückt, schickt der Web-Browser (Client) einen Request an die angegebene URI über das offene Internet. Der Web-Server (Server), dem

18 Vgl. Berners-Lee, Tim et al.: Uniform Resource Identifier (URI): Generic Syntax, Internet Engineering Task Force (Hrsg.), Januar 2005, https://tools.ietf.org/pdf/rfc3986.pdf, abgerufen am 26.01.2021, S. 1, 7, 18-25.

die URI zugeordnet werden kann, verarbeitet nun den Request und sucht in seinem Dateisystem nach der Web-Seite (Ressource), die über den angegebenen Endpunkt erreichbar ist. Der Web-Server schickt die Web-Seite als Response über das offene Internet an den Web-Browser des Users (Client). Abbildung 3 visualisiert den beschriebenen Vorgang. Bei verteilten Web-Anwendungen wird für die Kommunikation zwischen Client und Server HTTP[19] (Hypertext Transfer Protocol) eingesetzt.

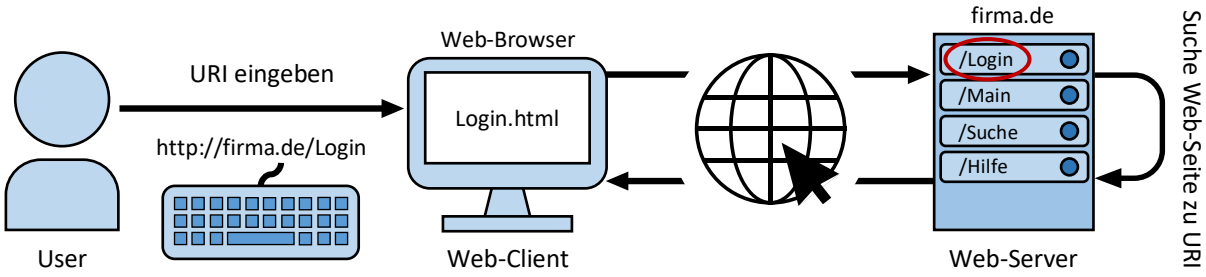

Abb. 3: Das Client-Server-Konzept am Beispiel einer Web-Seite (eigene Abbildung)

Der Untersuchungsbereich der vorliegenden Arbeit umfasst verteilte Web-Anwendungen für den Einsatz im industriellen Bereich. Der Begriff „Industrie“ umschreibt Unternehmen, die in Fertigungsanlagen maschinell und automatisiert Sachgüter herstellen. In solchen Fertigungsanlagen müssen alle Bestandteile des Fertigungsprozesses nahtlos ineinander übergehen. In Fertigungsprozessen werden Maschinen eingesetzt. Diese Maschinen werden von Software-Systemen gesteuert und tauschen Daten mit anderen Maschinen aus. Dadurch entstehen große Datenmengen, die für die Überwachung des Fertigungsprozesses aufbereitet werden müssen. Für diesen Zweck werden häufig verteilte Web-Anwendungen eingesetzt.

Die im Fertigungsprozess eines Industrie-Unternehmens anfallenden Daten sind vertraulich. Sollten solche Daten an die Öffentlichkeit gelangen oder durch Hacker ausgenutzt werden, kann dies schwerwiegende Folgen für das ganze Unternehmen haben. Deshalb müssen die Daten solcher Fertigungsanlagen sicher übertragen und gespeichert werden.

19 In Anhang A befindet sich eine kurze Einführung in die Themen „HTTP“ und „HTTPS“.

2.2 Untersuchungsobjekte

Authentifizierungsmethoden für Web-Anwendungen sind die Untersuchungsobjekte der vorliegenden Arbeit. Die Authentifizierung für eine Web-Anwendung kann mit verschiedenen Methoden umgesetzt werden. Eine Methode ist ein Bündel von aufeinander abgestimmten Maßnahmen, die einem Zweck dienen.[20] Authentifizierungsmethoden unterscheiden sich durch die verwendeten Maßnahmen sowie das Sicherheitsniveau, die Usability und den Administrationsaufwand. Die folgenden Authentifizierungsmethoden werden in der vorliegenden Arbeit untersucht:

1. OpenID Connect[21]: OpenID Connect (OIDC) ist eine Authentifizierungsmethode, die das Protokoll OAuth 2.0[22] erweitert. OIDC ist eine Authentifizierungsmethode, die für verschiedene Arten von Software-Systemen (z. B. Web-Anwendungen oder native Anwendungen) eingesetzt werden kann. OIDC wurde am 26. Februar 2014 als Standard der OpenID Foundation verabschiedet. Seitdem haben viele Unternehmen (z. B. Google und Microsoft)[23] OpenID Connect als Authentifizierungsmethode für ihre User implementiert.[24] OIDC liegt (Stand 14. Januar 2021) in Version 1.0 vor.[25]
2. FIDO2: Die FIDO Alliance ist ein offener Industrieverband, der sich zum Ziel gesetzt hat, Authentifizierungsstandards zu entwickeln, die dazu beitragen, die Abhängigkeit von Passwörtern zu reduzieren. Dazu entwickelt die FIDO Alliance technische Spezifikationen sowie Internet-Standards und betreibt Zertifizierungsprogramme für die Industrie. Die Geschichte der FIDO Alliance beginnt im Jahr 2009, als Vali-

20 Vgl. Digitales Wörterbuch der deutschen Sprache (Hrsg.): Methode, https://www.dwds.de/wb/Methode, abgerufen am 15.02.2021.

21 In Anhang B.1 sind weitere Information zur Historie von OIDC und OAuth zu finden.

22 OAuth 2.0 ist ein Internet-Protokoll, das zur Absicherung von APIs (Application Programming Interface) entwickelt wurde. OAuth 2.0 definiert auch einen Authentifizierungsvorgang. Dieser Authentifizierungsvorgang ist aber nicht der Zweck von OAuth 2.0 sondern nur eine Maßnahme des Protokolls. Siehe: Lodderstedt, Thorsten: OpenID Connect: Login mit OAuth Teil 1 – Grundlagen, Heise Medien GmbH & Co. KG (Hrsg.), 10. Juni 2014, https://www.heise.de/developer/artikel/OpenID-Connect-Login-mit-OAuth-Teil-1-Grundlagen-2218446.html?view= print, abgerufen am 25.01.2021, S. 2.

23 Vgl. OpenID Foundation (Hrsg.): Certified OpenID Connect Implementations, https://openid.net/developers/certified/, abgerufen am 15.02.2021.

24 Vgl. Lodderstedt, Thorsten: OpenID Connect: Login mit OAuth Teil 1 – Grundlagen, a. a. O., abgerufen am 25.01.2021, S. 1.

25 Vgl. Jones, Michael: Second Implementer's Draft of OpenID Connect User Questioning API Specification Approved, OpenID Foundation (Hrsg.), 14. Januar 2021, https://openid.net/2021/01/14/second-implementers-draft-of-openid-connect-user-questioning-api-specification-approved/, abgerufen am 15.02.2021.

dity Sensors and PayPal das Ersetzen von Passwörtern durch Biometrie diskutieren. Dabei kommt erstmals die Idee von einem Login-Verfahren auf, das durch Public-Key-Kryptographie und ausschließlich lokale Authentifizierung eine passwortlose Anmeldung ermöglicht. Im Juli 2012 wird die FIDO Alliance gegründet. Im Dezember 2014 wird die Version 1 des passwortlosen Authentifizierungsverfahrens FIDO UAF (Universal Authentication Factor) und des 2-Faktor-Authentifizierungsverfahrens FIDO U2F (Universal 2nd Factor) veröffentlicht. Über die Jahre entwickeln sich die Authentifizierungsmethoden weiter und ihr Funktionsumfang wächst. Im Februar 2016 wird FIDO2 erstmals vom W3C erwähnt. Im April 2018 wird FIDO2 dann veröffentlicht. Im September 2018 sind schon die ersten Produkte und Browser FIDO2-zertifiziert. Im März 2019 wird die Web-API „WebAuthn" vom W3C zum Internet-Standard erklärt.[26]

Die vorgenannten beiden Authentifizierungsmethoden werden in den Kapiteln 3 und 4 jeweils detailliert vorgestellt. In der vorliegenden Arbeit wird davon ausgegangen, dass Entwickler von Web-Anwendungen die jeweilige Authentifizierungsmethode korrekt implementiert haben. Denn durch eine falsche oder unangemessene Implementierung einer Authentifizierungsmethode kann sich deren Sicherheitsniveau, die Usability und/oder der Administrationsaufwand verschlechtern.

2.3 Vergleichskriterien

Das Forscherteam um Joseph Bonneau hat in seiner Veröffentlichung an der Universität Cambridge im März 2012 einen Kriterienkatalog für den Vergleich von 36 Web-Authentifikationsverfahren in 12 Kategorien aufgestellt.[27] Die 25 Vergleichskriterien sind in die drei Kategorien „Security benefits", „Usability benefits" und „Deployability benefits" eingeteilt.[28] Bonneau et al. untersuchen dabei jedoch nicht die Objekte der vorliegenden Arbeit. Die vorliegende Arbeit wendet somit die Vergleichskriterien von Bonneau et al. auf die Untersuchungsobjekte OIDC und FIDO2 an.

26 Vgl. FIDO Alliance (Hrsg.): History of FIDO Alliance, a. a. O., abgerufen am 01.03.2021.

27 Die Kategorien sind: „Klassisch", „Passwort-Manager", „Proxy", „Föderiert", „Graphisch", „Kognitiv", „Papier Tokens", „Visual Crypto", „Hardware Tokens", „Smartphone-basiert", „Biometrisch" und „Wiederherstellung". Die einzelnen Authentifizierungsmethoden sind einer Kategorie zugeordnet.

28 Vgl. Bonneau, Joseph et al.: The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes, University of Cambridge – Computer Laboratory (Hrsg.), März 2012, https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-817.pdf, abgerufen am 25.01.2021, S. 5.

Eine Authentifizierung wird durch einen Server durchgeführt. Dieser Server wird im Folgenden „Authentifizierungs-Server“ oder „Auth-Server“ genannt. Die Bezeichnung „Benutzer-Daten“ steht für Zugangsdaten wie Username und Passwort, die zur Authentifizierung benötigt werden. Die Bezeichnung Geheimnis wird nachfolgend für den Bestandteil der Benutzer-Daten benutzt, der geheim ist (z. B. ein Passwort oder ein Token).

Das Sicherheitsniveau einer Authentifizierungsmethode wird durch Kriterien beeinflusst, die die Privatsphäre und Resistenz gegen bestimmte Angriffsmuster betreffen. Die Kriterien S1 bis S10 repräsentieren die Sicherheits-Kriterien nach Bonneau et al.[29] Die Verfasser der vorliegenden Arbeit haben den Kriterienkatalog von Bonneau et al. um die Kriterien S11 und S12 ergänzt. Diese beiden Kriterien resultieren aus wesentlichen Eigenschaften des Untersuchungsobjekts FIDO2, das Bonneau et al. im Jahr 2012 noch nicht kennen konnten. Tabelle 1 zeigt die Sicherheits-Kriterien im Überblick.

S1 Resistent gegen physische Überwachung: Dritte können sich nicht als User ausgeben, indem sie die Benutzer-Daten des Users ausspähen. Ausspähen bezieht sich hier ausschließlich auf physische und nicht auf digitale Überwachung. Dieses Kriterium zeigt die Anfälligkeit einer Authentifizierungsmethode für Angriffe wie das „Shoulder surfing“ bei dem ein Dritter dem User bei der Eingabe seiner Benutzer-Daten zuschaut oder den Vorgang filmt. Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn solche Angriffe nicht möglich sind.

S2 Resistent gegen gezielte Nachahmung: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn das Wissen um personenbezogene Daten wie z. B. das Geburtsdatum des Users Dritten keine Authentifizierung ermöglicht. Sicherheitsfragen zur Wiederherstellung von Accounts erfüllen dieses Kriterium nicht.

S3 Resistent gegen Raten: Dieses Kriterium setzt sich aus den Kriterien „Resistent gegen nicht gedrosseltes Raten“ und „Resistent gegen gedrosseltes Raten“ nach Bonneau et al. zusammen. Eine Authentifizierungsmethode erfüllt dieses Kriterium vollständig, wenn Dritte die Benutzer-Daten eines Users nicht mit nicht gedrosseltem Raten erlangen können. Das Kriterium ist teilweise erfüllt, wenn Dritte die Benutzer-Daten eines Users nicht mit gedrosseltem Raten erlangen können. „Gedrosselt“ meint, dass nur wenige fehlgeschlagene Authentifizierungsversuche in kurzer

29 Vgl. Bonneau, Joseph et al.: The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes, a. a. O., abgerufen am 25.01.2021, S. 7f.

Zeit möglich sind und dann eine Zeitsperre einsetzt. Dieses Kriterium zeigt die Anfälligkeit einer Authentifizierungsmethode für Angriffe wie Rainbow-Table-Angriffe und Brute-Force-Angriffe[30].

S4 Resistent gegen internes Ausspähen: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn Dritte das Geheimnis des Users nicht digital ausspähen können. Mit „ausspähen" ist bei diesem Kriterium das Abgreifen des Geheimnisses innerhalb des Computers des Users gemeint. Ein Beispiel für die interne Überwachung ist eine Key-Logging-Software, die die Eingaben der Tastatur mitschneidet. 2-Faktor- (2FA) und Mehr-Faktor-Authentifizierungsmethoden (MFA) erfüllen dieses Kriterium ebenfalls. Authentifizierungsmethoden erfüllen dieses Kriterium nicht, wenn sie anfällig für „Replay-Angriffe"[31] sind.

S5 Resistent gegen Datenlecks bei anderen Authentifizierungs-Servern: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn Dritte sich nicht als User ausgeben können, indem sie Geheimnisse des gleichen Users von anderen Auth-Servern einsetzen. Dieses Kriterium zeigt die Anfälligkeit für Angriffe auf das Backend eines Software-Systems und wiederverwendete Geheimnisse.

S6 Resistent gegen Phishing: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn Dritte die Geheimnisse von Usern nicht mit Phishing abgreifen können. Beim Phishing wird der User z. B. über gefälschte E-Mails und gefälschte Web-Auftritte dazu verleitet, seine Benutzer-Daten von der Web-Anwendung anzugeben, die kopiert wird. Damit der User das tut, wird in der Phishing-Mail ein triftiger Grund genannt wie z. B., dass das Passwort ausläuft und erneuert werden muss.[32] Phishing ist eine Technik des „Social Engineering"[33].

30 „Rainbow-Tables" beinhalten die Ein- und Ausgabewerte von Hash-Funktionen von Passwörtern. Dadurch können in Datenbanken abgelegte Hash-Werte auf das zugehörige Passwort zurückgeführt werden. Brute-Force-Angriffen probieren Passwörter nach dem Trial-and-Error-Prinzip aus. Die zu Grunde liegende Methode, mögliche Passwörter zu finden, variiert. Siehe: Wintermeyer, Stefan: Nutzerpasswörter im Web intelligenter sichern, in: Linux-Magazin 08/19, https://www.linux-magazin.de/ausgaben/2019/08/passwoerter/, abgerufen am 03.03.2021, S. 2f.

31 „Replay-Angriffe" zeichnen die Kommunikation einer Session auf. Der Angreifer versucht zu einem späteren Zeitpunkt, die Kommunikation nachzuahmen, um eine valide Session zu erzeugen. Siehe: Adams, Carlisle: Replay Attack, in: Encyclopedia of cryptography and security, Henk C. A. van Tilborg – Eindhoven University of Technology (Hrsg.), abgerufen am 29.01.2021, S. 519.

32 Vgl. O. V.: Phishing E-Mails – Passwortdiebstahl durch Phishing, a. a. O., abgerufen am 29.12.2020.

33 Als „Social Engineering" bezeichnet man Techniken, bei denen Angreifer menschliche Eigenschaften, wie Hilfsbereitschaft, Vertrauen und Respekt vor Autorität ausnutzen, um die Geheimnisse von Usern zu erlangen. Siehe: O. V.: Social Engineering – der Mensch als Schwachstelle, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi.bund.de/DE/Themen/Verbraucher-

S7 Resistent gegen Diebstahl: Dieses Kriterium kann nur dann nicht erfüllt sein, wenn für die Authentifizierungsmethode ein physischer Authentifikator genutzt wird. Das Kriterium ist dann nicht erfüllt, wenn dieser physische Authentifikator gestohlen und von Dritten eingesetzt werden kann. Das Kriterium wird vollständig erfüllt, wenn der physische Authentifikator durch eine angemessene biometrische Maßnahme geschützt ist. Das Kriterium wird teilweise erfüllt, wenn der physische Authentifikator durch eine PIN (Personal Identification Number) geschützt ist.

S8 Keine dritten Auth-Server: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn sie keine Auth-Server benötigt, die von Dritten betrieben werden. Diese „dritten" Auth-Server können angegriffen werden, das Vertrauen des Web-Servers verlieren uvm.

S9 Zustimmung des Users: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn Authentifizierungsvorgänge nur mit explizitem Einverständnis des Users gestartet werden können, z. B. indem der User einen Knopf an einem externen Gerät drücken oder eine Smartcard präsentieren muss.

S10 Keine Verknüpfbarkeit von Accounts: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn Accounts des gleichen Users bei kollaborierenden Web-Anwendungen sich nicht durch die Authentifizierungsinformationen verknüpfen lassen. Andere Methoden zur Verknüpfung wie z. B. über die IP-Adresse eines Users werden hier außen vorgelassen.

S11 Keine Übertragungen von Geheimnissen: Die Übertragung von Geheimnissen über das Internet stellt ein Risiko da. Besonders „Man-in-the-middle" (MitM)-Angriffe[34] gefährden die Übertragung von Geheimnissen. Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn keine Geheimnisse übertragen werden müssen.

innen-und-Verbraucher/Cyber-Sicherheitslage/Methoden-der-Cyber-Kriminalitaet/Social-Engineering/ social-engineering_node.html, abgerufen am 02.02.2021.

34 Bei einem MitM-Angriff fängt der Angreifer die Nachrichten beider Kommunikationspartner von Beginn einer Session an ab und gibt sich gegenüber dem jeweils anderen als sein echter Kommunikationspartner aus. Der Angreifer kann die Nachrichten beider Kommunikationspartner lesen und verändern. Siehe: Grassi, Paul; Garcia, Michael; Fenton, James: NIST Special Publication 800-63-3 – Digital Identity Guidelines, U.S. Department of Commerce – National Institute of Standards and Technology (Hrsg.), Juni 2017, https://doi. org/10.6028/NIST.SP.800-63-3, abgerufen am 02.02.2021, S. 48.

Sicherheits-niveau	S1	Resistent gegen physische Überwachung
	S2	Resistent gegen gezielte Nachahmung
	S3	Resistent gegen Raten
	S4	Resistent gegen internes Ausspähen
	S5	Resistent gegen Datenlecks bei anderen Auth-Servern
	S6	Resistent gegen Phishing
	S7	Resistent gegen Diebstahl
	S8	Keine dritten Auth-Server
	S9	Zustimmung des Users
	S10	Keine Verknüpfbarkeit von Accounts
	S11	Keine Übertragungen von Geheimnissen
	S12	Geheimnisse nicht speichern

Tabelle 1: Die Liste der Sicherheits-Kriterien

S12 Geheimnisse nicht Speichern: Die Geheimnisse von Usern auf einem Server zu speichern ist ein Risiko; besonders, wenn sie nicht nach aktuellen Standards gehasht[35] abgelegt sind. Schwache Sicherheitsvorkehrungen verursachen immer wieder Datenlecks bei Servern.[36] Das Speichern von Geheimnissen auf dem Computer des Users ist ebenfalls problematisch. Denn Trojaner[37], Social Engineering und andere Angriffsarten ermöglichen es, Benutzer-Daten vom Computer des Users abzugreifen. Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn Geheimnisse auf Client- und Server-Seite nicht gespeichert werden. Das Kriterium wird auch noch erfüllt, wenn Geheimnisse auf der Client-Seite auf einem externen Gerät gespeichert werden, das konstruktionsbedingt nicht unbefugt auslesbar ist.

35 Beim „Hashen" von Passwörtern werden kryptographische „Einweg-Hash-Funktionen" eingesetzt, die aus Passwörtern eindeutige Hash-Werte errechnen, die heute nicht effizient auf das Passwort zurückgeführt werden können. Siehe: Wagner, Maik: Proseminar: "Electronic Commerce und Digitale Unterschriften" – Digitale Signaturen & Hashfunktionen, Technische Universität Chemnitz – Fakultät für Informatik – Professur Theoretische Informatik und Informationssicherheit (Hrsg.), https://tu-chemnitz.de/informatik/ThIS/downloads/courses/ss04/ecommerce/digsig.pdf, abgerufen am 03.02.2021, S. 1.

36 Vgl. Barth, Michael et al.: Spionage, Sabotage und Datendiebstahl – Wirtschaftsschutz in der vernetzten Welt – Studienbericht 2020, Bitkom e.V. (Hrsg.), https://www.bitkom.org/sites/default/files/2020-02/200211_bitkom_studie_wirtschaftsschutz_2020_final.pdf, abgerufen am 02.02.2021, S. 7, 10.

37 Ein Trojaner ist eine Schadsoftware, die sich in einem von außen sicher wirkenden IT-System verbirgt. Trojaner müssen von dem User installiert werden und können erst dann von sich aus Schaden anrichten. Trojaner werden häufig über Social Engineering beim User eingeschleust. Siehe: O. V.: Trojaner, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi.bund.de/DE/Themen/Verbraucherinnen-und-Verbraucher/Cyber-Sicherheitslage/Methoden-der-Cyber-Kriminalitaet/Schadprogramme/Trojaner/trojaner_node.html, abgerufen am 02.02.2021.

Das Wort „Usability“ bezeichnet die Nutzerfreundlichkeit von IT-Systemen. Die Betrachtung der Usability ist essentiell bei der Software-Entwicklung. Die Kriterien U1 bis U10 repräsentieren die Usability-Kriterien nach Bonneau et al.[38] Das Kriterium U11 haben die Verfasser der vorliegenden Arbeit durch die Recherche in neuerer Literatur beigefügt, die seit der Untersuchung durch Bonneau et al. im Jahr 2012 erschienen ist. Tabelle 2 zeigt die Usability-Kriterien im Überblick.

U1 Kein Erinnerungsvermögen: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn User sich keine Passwörter merken müssen. Das Kriterium wird teilweise erfüllt, wenn der User sich nur ein Passwort merken muss, um an die Passwörter für viele Web-Anwendungen heranzukommen.

U2 Skalierbar für User: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn diese Authentifizierungsmethode für mehrere Web-Anwendungen eingesetzt werden kann, ohne dass der Aufwand für den User dadurch größer wird (z. B. durch zusätzliche Passwörter).

U3 Keine zusätzliche Hardware: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn keine zusätzliche Hardware benötigt wird, um eine Authentifizierung durchzuführen. Das Kriterium wird teilweise erfüllt, wenn es sich bei der zusätzlichen Hardware um etwas handelt, das der User i. d. R. sowieso mit sich herumträgt wie z. B. ein Smartphone.

U4 Kein physischer Aufwand: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn ein User nicht physisch mit dem Computersystem interagieren muss, um sich zu authentifizieren. Wenn ein Knopf einmal gedrückt werden muss, wird das Kriterium immer noch erfüllt. Authentifizierungsmethoden, die Tastatureingaben bedürfen, erfüllen dieses Kriterium nicht.

U5 Zeit-effizient: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn ein Authentifizierungsvorgang wenig Zeit in Anspruch nimmt. Die Registrierung des Users für diese Authentifizierungsmethode muss ebenfalls in kurzer Zeit durchführbar sein.

U6 Niedrige False-Reject-Rate: Dieses Kriterium beinhaltet zusätzlich das Kriterium „Intuitiv nutzbar“ nach Bonneau et al. Eine Authentifizierungsmethode erfüllt die-

38 Vgl. Bonneau, Joseph et al.: The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes, a. a. O., abgerufen am 25.01.2021, S. 6.

ses Kriterium, wenn ihre „False-Reject-Rate“ (FRR) niedrig ist. Die False-Reject-Rate gibt den Anteil abgelehnter Authentifizierungsversuche eines legitimen Users an.[39] Eine Authentifizierungsmethode muss zuverlässig und einfach durchführbar sein, um dieses Kriterium zu erfüllen. Auch User, denen die Authentifizierungsmethode unbekannt ist, müssen diese intuitiv einsetzen können.

U7 Faktor wiederherstellbar: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn nach Verlust eines Authentifizierungsfaktors (z. B. Vergessen eines Passworts oder Verlieren zusätzlicher Hardware) die Möglichkeit zur Authentifizierung schnell und komfortabel vom User wiederhergestellt werden kann.

U8 Niedriger Migrationsaufwand für den User: Aktuell herrschen passwortbasierte Authentifizierungsmethoden vor.[40] Dazu gehören auch Authentifizierungsmethoden wie OIDC. Die Entscheidung für einen Wechsel der Authentifizierungsmethode wird durch den Migrationsaufwand für den User beeinflusst. Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn die Registrierung des Users und der Einsatz der Authentifizierungsmethode zusammengenommen weniger aufwendig ist als die Nutzung einer passwortbasierten Authentifizierung. Authentifizierungsmethoden, die Passwörter nutzen, erfüllen dieses Kriterium.

U9 Geheimnisse nicht erneuern: Das zyklische Erneuern von Passwörtern steigert das Sicherheitsniveau von Authentifizierungsverfahren. Für eine Erneuerung muss der User selbst tätig werden und sich ein neues Passwort ausdenken und merken. Da ein User i. d. R. mehrere Software-Systeme nutzt und der Erneuerungszyklus mehrmals im Jahr durchlaufen werden muss, steigt der Aufwand für den User durch zyklisches Erneuern von Passwörtern stark. Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn keine Erneuerung eines Geheimnisses nötig ist und dadurch keine Einbußen für das Sicherheitsniveau der Authentifizierungsmethode entstehen. Dieses Kriterium ist auch erfüllt, wenn eine Authentifizierungsmethode eine zyklische Erneuerung durchführt, ohne dass Aufwand für den User

39 Vgl. Günther, Oliver; Fabian, Benjamin; Ziekow, Holger: Biometrie, in: Enzyklopädie der Wirtschaftsinformatik, Universität Potsdam – Lehrstuhl für Wirtschaftsinformatik – Norbert Gronau (Hrsg.), 29. September 2020, https://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/technologien-methoden/Informatik--Grundlagen/automatische-identifikation/biometrie/index.html?searchterm=false+reject, abgerufen am 01.02.2020.

40 Vgl. Bonneau, Joseph et al.: The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes, a. a. O., abgerufen am 25.01.2021, S. 5.

entsteht. Das Kriterium ist teilweise erfüllt, wenn nur ein Geheimnis, das die Geheimnisse vieler Web-Anwendungen freischaltet, zyklisch erneuert werden muss.

U10 Barrierefrei: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn alle User mit geistigen oder körperlichen Einschränkungen, die passwortbasierte Authentifizierung einsetzen können, auch diese Authentifizierung einsetzen können.

U11 Single Sign-On: Im Durchschnitt hat jeder User ca. 90 Accounts im Internet.[41] Damit ein User sich nicht 90 Passwörter merken muss, wurde das Verfahren Single Sign-On (SSO) entwickelt. Mit SSO muss ein User sich nur einmal authentifizieren und kann damit auf alle Web-Anwendungen mit dem gleichen Auth-Server zugreifen.[42] Durch SSO müssen User sich weniger Passwörter merken und auch weniger Authentifizierungsvorgänge durchlaufen. Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn sie SSO ermöglicht.

Usability	U1	Kein Erinnerungsvermögen
	U2	Skalierbar für Nutzer
	U3	Keine zusätzliche Hardware
	U4	Kein physischer Aufwand
	U5	Zeit-effizient
	U6	Niedrige False-Reject-Rate
	U7	Faktor wiederherstellbar
	U8	Niedriger Migrationsaufwand für den User
	U9	Geheimnisse nicht erneuern
	U10	Barrierefrei
	U11	Single Sign-On

Tabelle 2: Die Liste der Usability-Kriterien

Der Administrationsaufwand einer Authentifizierungsmethode wird bei Bonneau et al. mit „Deployability“ bezeichnet. Die Kriterien A1 bis A5 bestimmen nach Bonneau et al. den Administrationsaufwand einer Authentifizierungsmethode.[43] Das Kriterium A6 ha-

41 Vgl. Shikiar, Andrew: FIDO2: A Web without passwords – W3C track at the web conference, fido alliance (Hrsg.), 15. Mai 2019, https://www.w3.org/2019/05/15-w3c-track/assets/andrew-fido.PDF, abgerufen am 30.12.2020.

42 Vgl. O. V.: Single Sign-On, Auth0.com (Hrsg.), https://auth0.com/docs/sso, abgerufen am 25.01.2021.

43 Vgl. Bonneau, Joseph et al.: The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes, a. a. O., abgerufen am 25.01.2021, S. 6f.

ben die Verfasser der vorliegenden Arbeit durch die Recherche in neuerer Literatur beigefügt, die seit der Untersuchung durch Bonneau et al. im Jahr 2012 erschienen ist. Tabelle 3 zeigt die Administrationsaufwand-Kriterien im Überblick.

A1 Vernachlässigbare Kosten pro User: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn die gesamten Kosten pro User auf Seiten des Users und auf Seiten des Betreibers der Web-Anwendung (z. B. zusätzliche Hardware) vernachlässigbar klein sind.

A2 Server-kompatibel: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn für die neue Authentifizierungsmethode die Server-Infrastruktur und Software auf der Server-Seite nicht ausgetauscht werden muss.

A3 Browser-kompatibel: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn ein User über einen Client, der aktuelle Standards erfüllt, diese Authentifizierungsmethode nutzen kann. Das Kriterium wird nicht erfüllt, wenn zusätzliche Software benötigt wird.

A4 Ausgereift: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn sie bereits von einer breiten Menge an Web-Anwendungen abseits der Forschung eingesetzt wird. Außerdem muss die Authentifizierungsmethode als Standard aufgenommen, durch User getestet und dokumentiert worden sein.

A5 Nicht proprietär: Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn jeder die Authentifizierungsmethode implementieren kann, ohne für eine Lizenz oder ähnliches zu bezahlen.

A6 Mobile Endgeräte: Mobile Endgeräte wie Smartphones oder Tablets sind ein wesentlicher Teil der Rechner, mit denen Menschen das Internet nutzen.[44] Deshalb sollten Web-Anwendungen und ihre Authentifizierungsmethoden über mobile Endgeräte nutzbar sein. Eine Authentifizierungsmethode erfüllt dieses Kriterium, wenn sie auf mobilen Endgeräten mit aktuellen Betriebssystemen verfügbar sind.

In Tabelle 4 werden alle Sicherheits-, Usability- und Administrationsaufwand-Kriterien in einer Übersicht zusammengefasst. Dieser Kriterienkatalog wird nachfolgend zum Vergleich der Authentifizierungsmethoden OIDC und FIDO2 herangezogen.

44 Vgl. Institut für Demoskopie Allensbach (Hrsg.): Auszug – Stationäre und mobile Internetnutzung, in: Allensbacher Computer und Technik-Analyse, 2016, https://www.ifd-allensbach.de/fileadmin/ACTA/ACTA2016/Codebuchausschnitte_ACTA_2016/ACTA2016_Stationaere_Mobile-Internetnutzung.pdf, abgerufen am 02.02.2021, S. 119.

Administrationsaufwand	A1	Vernachlässigbare Kosten pro User
	A2	Server-kompatibel
	A3	Browser-kompatibel
	A4	Ausgereift
	A5	Nicht proprietär
	A6	Mobile Endgeräte

Tabelle 3: Die Liste der Administrationsaufwand-Kriterien

Sicherheits-niveau	S1	Resistent gegen physische Überwachung
	S2	Resistent gegen gezielte Nachahmung
	S3	Resistent gegen Raten
	S4	Resistent gegen internes Ausspähen
	S5	Resistent gegen Datenlecks bei anderen Auth-Servern
	S6	Resistent gegen Phishing
	S7	Resistent gegen Diebstahl
	S8	Keine dritten Auth-Server
	S9	Zustimmung des Users
	S10	Keine Verknüpfbarkeit von Accounts
	S11	Keine Übertragungen von Geheimnissen
	S12	Geheimnisse nicht speichern
Usability	U1	Kein Erinnerungsvermögen
	U2	Skalierbar für Nutzer
	U3	Keine zusätzliche Hardware
	U4	Kein physischer Aufwand
	U5	Zeit-effizient
	U6	Niedrige False-Reject-Rate
	U7	Faktor wiederherstellbar
	U8	Niedriger Migrationsaufwand für den User
	U9	Geheimnisse nicht erneuern
	U10	Barrierefrei
	U11	Single Sign-On
Administrations-aufwand	A1	Vernachlässigbare Kosten pro User
	A2	Server-kompatibel
	A3	Browser-kompatibel
	A4	Ausgereift
	A5	Nicht proprietär
	A6	Mobile Endgeräte

Tabelle 4: Der Kriterienkatalog zum Vergleich von OIDC und FIDO2

2.4 Einsatz-Szenarien

Innerhalb des Untersuchungsbereichs gibt es mehrere Szenarien für den Einsatz der Untersuchungsobjekte. Diese Szenarien unterscheiden sich durch die Ausprägung der Eigenschaften des Untersuchungsbereichs. Es gibt viele Möglichkeiten, Szenarien zu spezifizieren. In der vorliegenden Arbeit werden die in Abbildung 4 dargestellten Szenarien durch die Lokalisierung von Authentifizierungs-Server und Web-Server unterschieden.

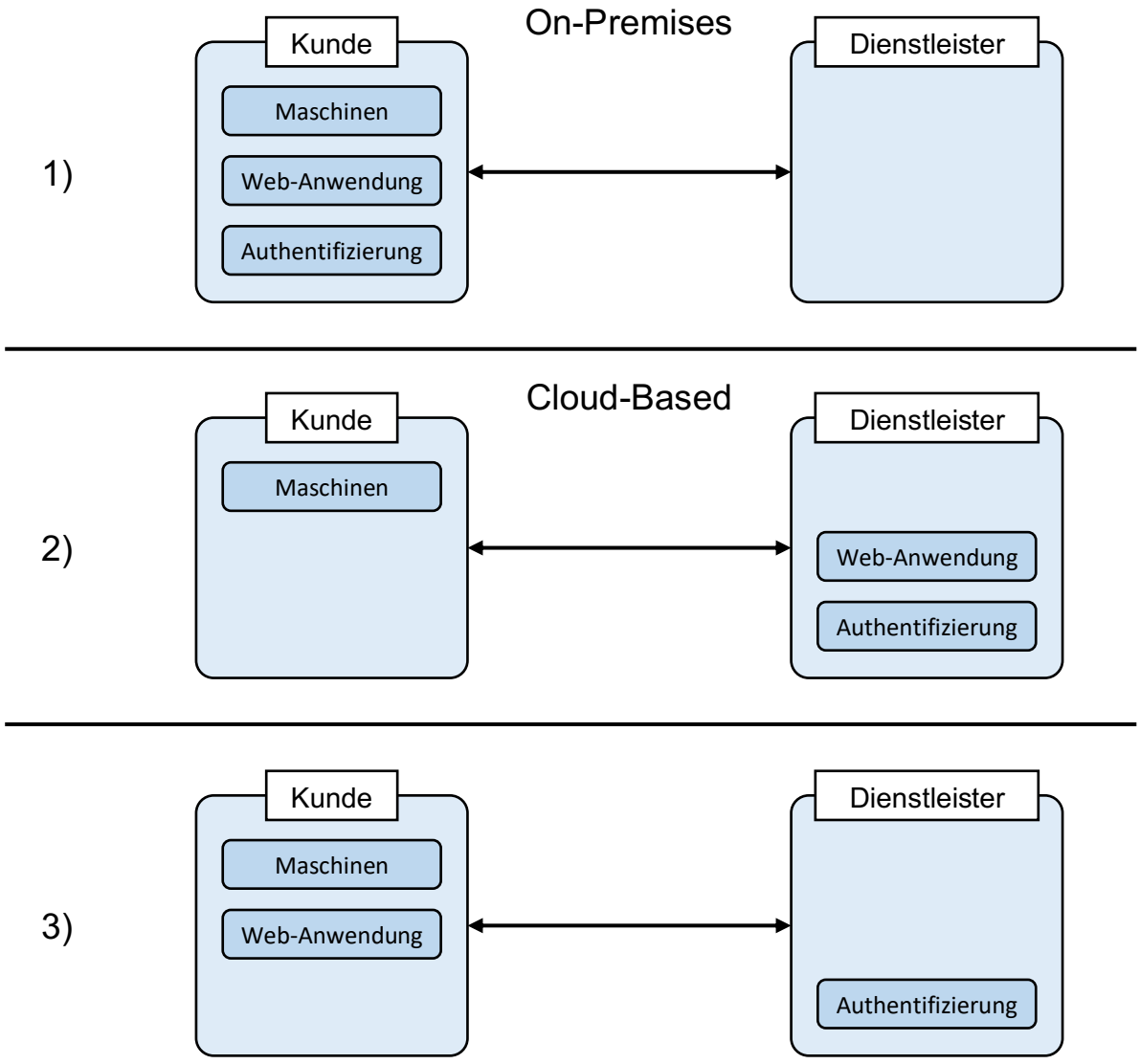

Abb. 4: Szenarien für den Einsatz verteilter Web-Anwendungen (eigene Abbildung)

Die Fertigungsmaschinen beim Kunden werden durch Web-Anwendungen überwacht, deren Nutzung ein Auth-Server („Authentifizierung“) für User erlauben muss. Unabhängig vom Szenario befinden sich die Maschinen einer Fertigungsanlage immer beim Kun-

den. Bei Szenario 1 befinden sich der Web-Server zum Hosten der Web-Anwendung und der Auth-Server beide vor Ort beim Kunden. Dieses Szenario wird „On-Premises“ („vor Ort“) genannt. Dabei wird die Web-Anwendung vollständig auf Kunden-eigenen Servern im Kunden-eigenen Intranet betrieben. Web-Anwendungen in einem Intranet sind nur aus diesem Intranet erreichbar. On-Premises-Web-Anwendungen haben den Vorteil, dass vertrauliche Daten nie den vom Kunden kontrollierten Bereich verlassen. Diesem Vorteil stehen u. a. die Nachteile der sprungfixen Kosten von On-Premises-Web-Anwendungen gegenüber. Die Anschaffungskosten für Hardware und Software sowie die Beschäftigung von Administrations-Personal schränken die Skalierbarkeit von On-Premises-Web-Anwendungen stark ein.[45]

Bei Szenario 2 befinden sich der Web-Server zum Hosten der Web-Anwendung und der Auth-Server beide nicht beim Kunden, sondern bei Dienstleistern. Die Dienstleister bieten dem Kunden die Web-Anwendung zur Nutzung über das offene Internet als Service an. Dieses Szenario wird „Cloud-Based“ genannt. Cloud-Based-Web-Anwendungen sind gut skalierbar, da der Kunde selbst nicht in Infrastruktur investieren muss. Stattdessen werden die Dienste von Anbietern wie z. B. Amazon Web Services oder Microsoft Azure nach Bedarf in Anspruch genommen. Der Großteil der Kosten ist damit variabilisiert. Allerdings laufen die Web-Anwendung und die Authentifizierung auf Kunden-externen Servern und somit liegen vertrauliche Daten auch auf Kunden-externen Servern.[46]

Szenario 3 ist eine Mischung aus Szenario 1 und Szenario 2. Dabei befindet sich der Web-Server zum Hosten der Web-Anwendung („Web-Anwendung“) beim Kunden und der Auth-Server („Authentifizierung“) bei einem Dienstleister. In diesem Szenario legen Web-Anwendungen keine Maschinendaten der kundenseitigen Fertigungsanlage auf externen Servern ab. Nur die Authentifizierungs-Informationen müssen auf Kunden-externen Servern vorgehalten werden. Die Auslagerung der Authentifizierung reduziert den Administrationsaufwand auf Seiten des Kunden.

45 Vgl. Steppan, Bernhard: Cloud statt on Premises: Java-Altanwendungen in AWS migrieren, in: iX 3/20, Heise Medien GmbH & Co. KG (Hrsg.), 20. Februar 2020, https://www.heise.de/ratgeber/Cloud-statt-on-Premises-Java-Altanwendungen-in-AWS-migrieren-4661891.html?view=print, abgerufen am 18.02.2021, S. 136.

46 Vgl. Mell, Peter; Grance, Tim: The NIST Definition of Cloud Computing, National Institute of Standards and Technology (Hrsg.), September 2011, https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf, abgerufen am 05.02.2021, S. 2f.

Szenario 1 wird in der vorliegenden Arbeit nicht betrachtet. Zum einen ist dieses Szenario begrenzt auf Unternehmen mit besonderem Sicherheitsbedarf wie z. B. in der Pharmabranche.

Szenario 2 und 3 unterscheiden sich durch die Lokalisierung des Web-Servers, der die Web-Anwendung („Web-Anwendung“) hostet. Der Vergleich der Authentifizierungsmethoden in Szenario 2 unterscheidet sich nur in einem Spezialfall bei der Umsetzung von OIDC von dem Vergleich der Authentifizierungsmethoden in Szenario 3 beim Sicherheits-Kriterium S11 (Keine Übertragungen von Geheimnissen). Da dieser Unterschied marginal ist, wird in der vorliegenden Arbeit nur Szenario 2 betrachtet. Das wird dadurch gestützt, dass aktuell die Cloud-Based-Lösung ein Trend zum Betrieb von Applikationen und User-Authentifizierung in der Industrie ist. Deshalb wird in der vorliegenden Arbeit der Vergleich der Authentifizierungsmethoden OIDC und FIDO2 nur in Szenario 2 durchgeführt.

3 Authentifizierungsmethode „OpenID Connect“

3.1 Grundlagen von „OpenID Connect“

OIDC basiert auf dem Autorisierungs-Framework „OAuth 2.0“ (Open Authorization).[47] OAuth wurde dafür konzipiert, dass eine Anwendung per M2M-Kommunikation (Machine-to-Machine) auf die Ressourcen und Dienste einer anderen Anwendung zugreifen kann.[48] OAuth wurde entwickelt, um sichere Kommunikation zwischen Anwendungen auf Basis von APIs zu ermöglichen. APIs (Application Programming Interfaces) sind die Schnittstellen von Anwendungen. Das Funktionsprinzip von OAuth wurde in OIDC übernommen, um die Kommunikation zwischen Mensch und Anwendung zu ermöglichen. OIDC ist eine Erweiterung von OAuth 2.0 und bietet zusätzliche User-Login-Funktionen. Aus OAuth 2.0 wird OIDC – eine Authentifizierungsmethode für H2M-Kommunikation (Human-to-Machine). Entwickler von Web-Anwendungen müssen keine proprietären User-APIs entwickeln, sondern können den OIDC-Standard verwenden.[49]

47 In Anhang B.1 ist eine kurze Einführung zur Historie von OAuth und OIDC zu finden.

48 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, Internet Engineering Task Force (Hrsg.), Oktober 2012, https://tools.ietf.org/pdf/rfc6749.pdf, abgerufen am 25.01.2021, S. 1.

49 Vgl. Lodderstedt, Thorsten: OpenID Connect: Login mit OAuth Teil 1 – Grundlagen, Heise Medien GmbH & Co. KG (Hrsg.), 10. Juni 2014, https://www.heise.de/developer/artikel/OpenID-Connect-Login-mit-OAuth-Teil-1-Grundlagen-2218446.html?view= print, abgerufen am 25.01.2021, S. 1.

OIDC setzt die gleichen Protokollabläufe und Kommunikations-Objekte (Objekte, die autorisierungsrelevante Informationen von einem Kommunikationspartner zum anderen überbringen) wie OAuth ein.[50] Die Kommunikations-Objekte sind „Authorization-Codes", „Access-Tokens" und „Refresh-Tokens".[51] Authorization-Codes werden eingesetzt, um die Autorisierung einer HTTP-Nachricht durch einen User nachzuwiesen.[52] Access-Tokens gewähren einer Anwendung Zugang zu einer Web-Anwendung.[53] Refresh-Tokens sind dazu da, um einen neuen Access-Token anzufordern, wenn der alte Access-Token abgelaufen ist.[54]

Um zu verstehen, wie OIDC funktioniert, ist es hilfreich, OAuth 2.0 zu kennen. Deshalb wird nachfolgend zunächst erläutert, wie OAuth 2.0[55] funktioniert. Für die M2M-Kommunikation mit OAuth 2.0 sind die folgenden vier Akteure relevant:[56]

Akteur 1: User mit Web-Browser

Der User einer Web-Anwendung sitzt vor seinem Computer. Auf diesem Computer läuft ein „Useragent". Dieser Useragent ist für die Kommunikation des Users mit den anderen Akteuren des OAuth-2.0-Protokolls verantwortlich. Für Web-Anwendungen ist der Useragent ein Web-Browser.[57]

Akteur 2: Client-Anwendung

Die Client-Anwendung ist bei OAuth 2.0 eine Anwendung, die auf eine Web-Anwendung zugreifen möchte.[58]

50 Vgl. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, OpenID Foundation (Hrsg.), https://openid.net/specs/openid-connect-core-1_0.html, abgerufen am 18.02.2021, S. 12, 17, 20. Vgl. auch Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 8.

51 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 8. In Anhang B.2 werden die Kommunikations-Objekte von OIDC und OAuth 2.0 detailliert beschrieben.

52 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 8.

53 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 10.

54 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 10.

55 OAuth 2.0 hat vier Protokollabläufe, die vorgeben, wann wer welche Daten übermittelt oder verarbeitet. OAuth 2.0 spezifiziert vier Authorization-Grants: „Authorization-Code-Grant", „Implicit-Grant", „Resource-Owner-Password-Credentials-Grant", „Client-Credentials-Grant". Für die vorliegende Arbeit ist nur der Authorization-Code-Grant relevant, da die anderen Protokollabläufe Spezialfälle abdecken. Siehe: Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 8f.

56 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 24.

57 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 6f.

58 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 6f.

Akteur 3: Auth-Server

Der Auth-Server ist eine vertrauenswürdige dritte Instanz, mit der ein User eine Client-Anwendung für den Zugriff auf die Web-Anwendung autorisiert.[59]

Akteur 4: Web-Anwendung

Die Web-Anwendung wird von einem Web-Server gehostet. Autorisierte Client-Anwendungen können auf die Web-Anwendung zugreifen.[60]

Damit eine Client-Anwendung auf die Web-Anwendung zugreifen kann, müssen die folgenden vier Phasen durchlaufen werden:

Phase 1: Registrierung des Users

Damit eine Client-Anwendung auf die Web-Anwendung zugreifen kann, muss der User einen Account bei der Web-Anwendung besitzen.[61] Durch eine Registrierung des Users wird ein User-Account bei der Web-Anwendung angelegt. Bei dieser Registrierung hinterlegt der User seine Authentifizierungsinformationen bei dem Auth-Server, den die Web-Anwendung einsetzt. Die Art der Authentifizierung des Users ist nicht durch OAuth 2.0 spezifiziert.[62] Deshalb können die Authentifizierungsinformationen unterschiedlich sein. In der vorliegenden Arbeit wird für OAuth 2.0 von einer Authentifizierung mit Username und Passwort ausgegangen. Für eine Registrierung muss der User i. d. R. ein HTML-Formular mit User-Daten wie Username und Passwort ausfüllen, das dann per HTTP-POST-Request an den mit der Web-Anwendung kooperierenden Auth-Server geschickt wird. Abschließend hinterlegt der Auth-Server die User-Daten in seiner Datenbank. Der User wurde bei der Web-Anwendung registriert.

Phase 2: Registrierung der Client-Anwendung

Damit ein Auth-Server Kommunikations-Objekte wie z. B. Access-Tokens einer Client-Anwendung zuordnen kann, muss die Client-Anwendung beim Auth-Server registriert sein. Die benötigten Abläufe finden bereits im Entwicklungsprozess einer Client-Anwendung statt. Der User ist daher hier nicht aktiv eingebunden. Für die Re-

59 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 6f.

60 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 6f.

61 Vgl. Lodderstedt, Thorsten; Hiller, Jochen: Flexible und sichere Internetdienste mit OAuth 2.0, Heise Medien GmbH & Co. KG (Hrsg.), 27. Dezember 2013, https://www. heise.de/developer/artikel/Flexible-und-sichere-Internetdienste-mit-OAuth-2-0-2068404.html?view=print, abgerufen am 25.01.2021, S. 1.

62 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 18.

gistrierung muss die Client-Anwendung bestimmte Informationen wie ihre URI hinterlegen und bekommt dafür vom Auth-Server Login-Informationen in Form von Username (Client-ID, Identifikationsnummer) und Passwort (Client-Secret). Wie dieser Austausch vonstatten geht, ist durch OAuth 2.0 nicht spezifiziert. Ein möglicher Weg ist der Austausch per HTTP oder auch über menschliche Kommunikation.[63]

Phase 3: Autorisierungsablauf

In dieser Phase wird ein bestimmter OAuth-2.0-Protokollablauf abgearbeitet und abschließend ein Access-Token für eine Client-Anwendung ausgestellt. Der User ist in dieser Phase aktiv eingebunden, da er seine Login-Informationen (Username und User-Passwort) eingeben muss.[64]

Phase 4: Zugriff der Client-Anwendung auf die Web-Anwendung

In dieser Phase nutzt die Client-Anwendung die Web-Anwendung, um auf die Fertigungsmaschinen zuzugreifen. Dafür setzt die Client-Anwendung den in Phase 3 erhaltenen Access-Token ein.

Nachfolgend werden die Ablaufschritte von Phase 3 und Phase 4 detailliert beschrieben (siehe Abbildung 5 und 6).[65]

Schritt 1: Anwendung aufrufen

Der User öffnet in seinem Web-Browser die Client-Anwendung.

Schritt 2: Autorisierung benötigt (schicke Client-ID)

Die Client-Anwendung soll auf die Web-Anwendung zugreifen. Dafür muss ein User die Client-Anwendung autorisieren. Damit diese Autorisierung stattfinden kann, muss der User sich bei der Web-Anwendung authentifizieren. Der Web-Browser erhält von der Client-Anwendung deren Client-ID.

Schritt 3: User authentifizieren und Client autorisieren (schicke Client-ID)

Der Web-Browser wird von der Client-Anwendung an den Auth-Server der Web-Anwendung weitergeleitet. Die Client-ID der Client-Anwendung wird an den Auth-Server geschickt, um mitzuteilen, welche Client-Anwendung auf die Web-Anwendung zugreifen will. Der User muss sich gleichzeitig beim Auth-Server der Web-Anwen-

63 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 13, 15f.

64 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 7.

65 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 24f.

dung mit Username und User-Passwort authentifizieren und den Zugriff der Client-Anwendung auf die Web-Anwendung autorisieren.

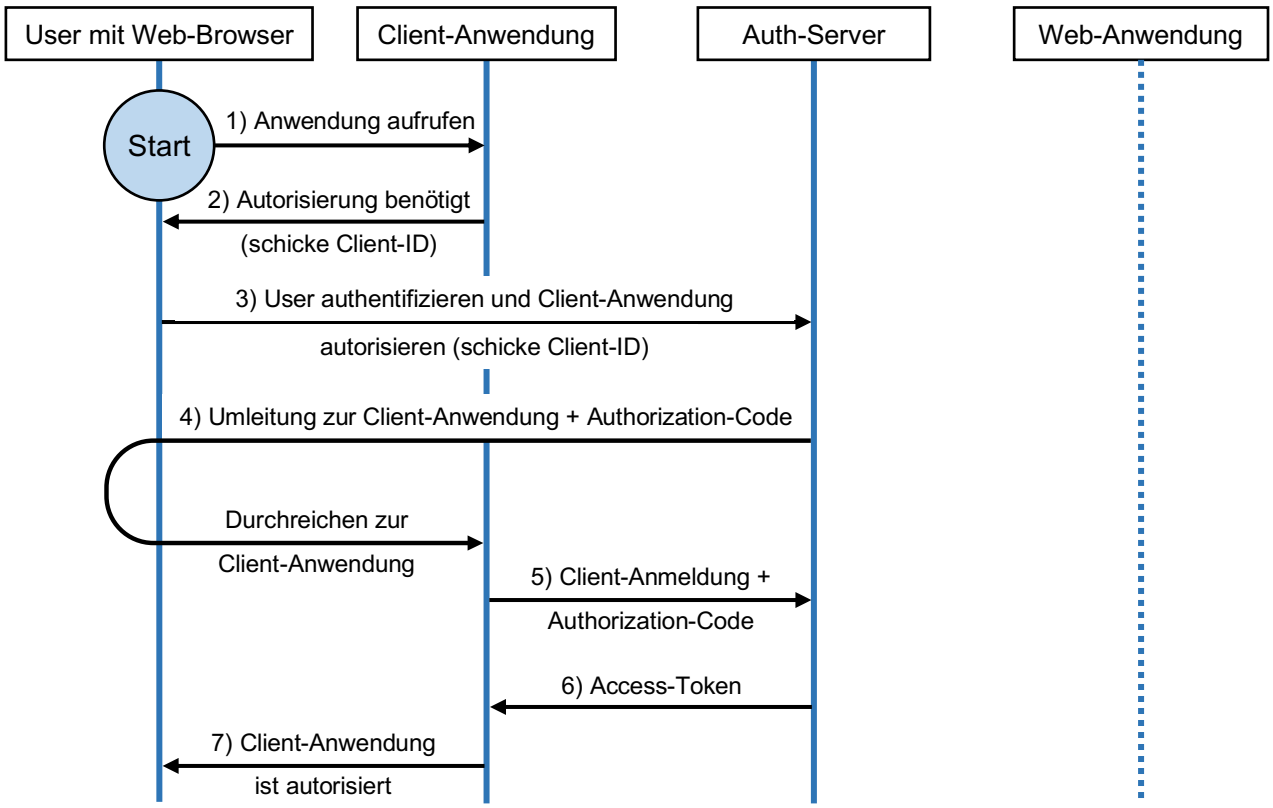

Abb. 5: Phase 3 – Der OAuth-2.0-Autorisierungsablauf (eigene Abbildung)

Schritt 4: Umleitung zur Client-Anwendung mit Authorization-Code

Der Auth-Server erstellt für die übergebene Client-ID einen Authorization-Code und der User wird zurück zur Client-Anwendung[66] umgeleitet. Der Authorization-Code wird durch den Web-Browser des Users an die Client-Anwendung weitergeleitet.

Schritt 5: Client-Anmeldung mit Authorization-Code

Die Client-Anwendung schickt den Authorization-Code zusammen mit ihrer Client-ID und ihrem Client-Secret an den Auth-Server. Client-ID und Client-Secret authentifizieren die Client-Anwendung beim Auth-Server. Der mitgelieferte Authorization-Code bescheinigt dem Auth-Server die Autorisierung der authentifizierten Client-Anwendung durch den User zum Zugriff auf die Web-Anwendung.

66 Bei der Umleitung zur Client-Anwendung sendet der Auth-Server den User an eine Umleitungs-Adresse – die „Redirect-URI“. Diese hat die Client-Anwendung bei ihrer Registrierung beim Auth-Server hinterlegt (siehe Phase 2). Siehe: Lodderstedt, Thorsten; Hiller, Jochen: Flexible und sichere Internetdienste mit OAuth 2.0, a. a. O., abgerufen am 25.01.2021, S. 7. Vgl. auch Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 13.

Schritt 6: Access-Token

Wenn die Client-ID des Authorization-Codes mit der Client-ID der authentifizierten Client-Anwendung übereinstimmt und der Authorization-Code valide[67] ist, stellt der Auth-Server der authentifizierten Client-Anwendung ein Access-Token aus und schickt dieses Access-Token an die Client-Anwendung.

Schritt 7: Client-Anwendung ist autorisiert

Die Autorisierung der Client-Anwendung ist nun abgeschlossen und die Client-Anwendung kann auf die Web-Anwendung zugreifen.

Der Zugriff auf die Web-Anwendung erfolgt in Phase 4 (siehe Abbildung 6).[68]

Schritt 7: Auf Client-Anwendung zugreifen

Wenn der Autorisierungsvorgang (siehe Phase 3) abgeschlossen ist, wird der User auf die Client-Anwendung umgeleitet.

Schritt 8: Zugriff auf die Web-Anwendung (schicke Access-Token)

Die Client-Anwendung fügt den Access-Token in den Authorization-Header[69] jeder HTTP-Nachricht an den Web-Server der Web-Anwendung ein.

Schritt 9: Validiere Access-Token

Der Web-Server der Web-Anwendung muss einen erhaltenen Access-Token validieren. Wie diese Validierung abläuft, ist nicht durch OAuth 2.0 spezifiziert. In der vorliegenden Arbeit schickt der Web-Server den Access-Token an den Auth-Server, damit dieser den Access-Token validiert. Der Auth-Server prüft in seiner Datenbank, ob es den übergegebenen Access-Token gibt und die mit dem Access-Token übergebenen Parameter valide Werte enthalten. [70] Dann schickt der Auth-Server der Web-Anwendung das Ergebnis der Validierung des Access-Tokens per HTTP zu.

67 Dies bedeutet u. a., dass der Authorization-Code nicht abgelaufen ist und alle zusätzlich übergebenen Attribute einen validen Wert haben. Siehe: Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 26.

68 Vgl. Grof, Thomas: OAuth 2.0: Anbietervergleich für selbst gehostete Autorisierungsserver, Johannes Kepler Universität Linz – Institut für Wirtschaftsinformatik – Software Engineering (Hrsg.), Januar 2020, https://epub.jku.at/obvulihs/download/pdf/47 77592?originalFilename=true, abgerufen am 25.01.2021, S. 39.

69 In Anhang A befindet sich eine kurze Einführung in die Themen „HTTP“ und „HTTPS“. Dort gibt es auch eine Erläuterung zum „Header“ einer HTTP-Nachricht.

70 In Anhang B.2 ist näher erläutert, was diese Parameter sind und welche Werte sie haben sollen. Siehe: Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 48.

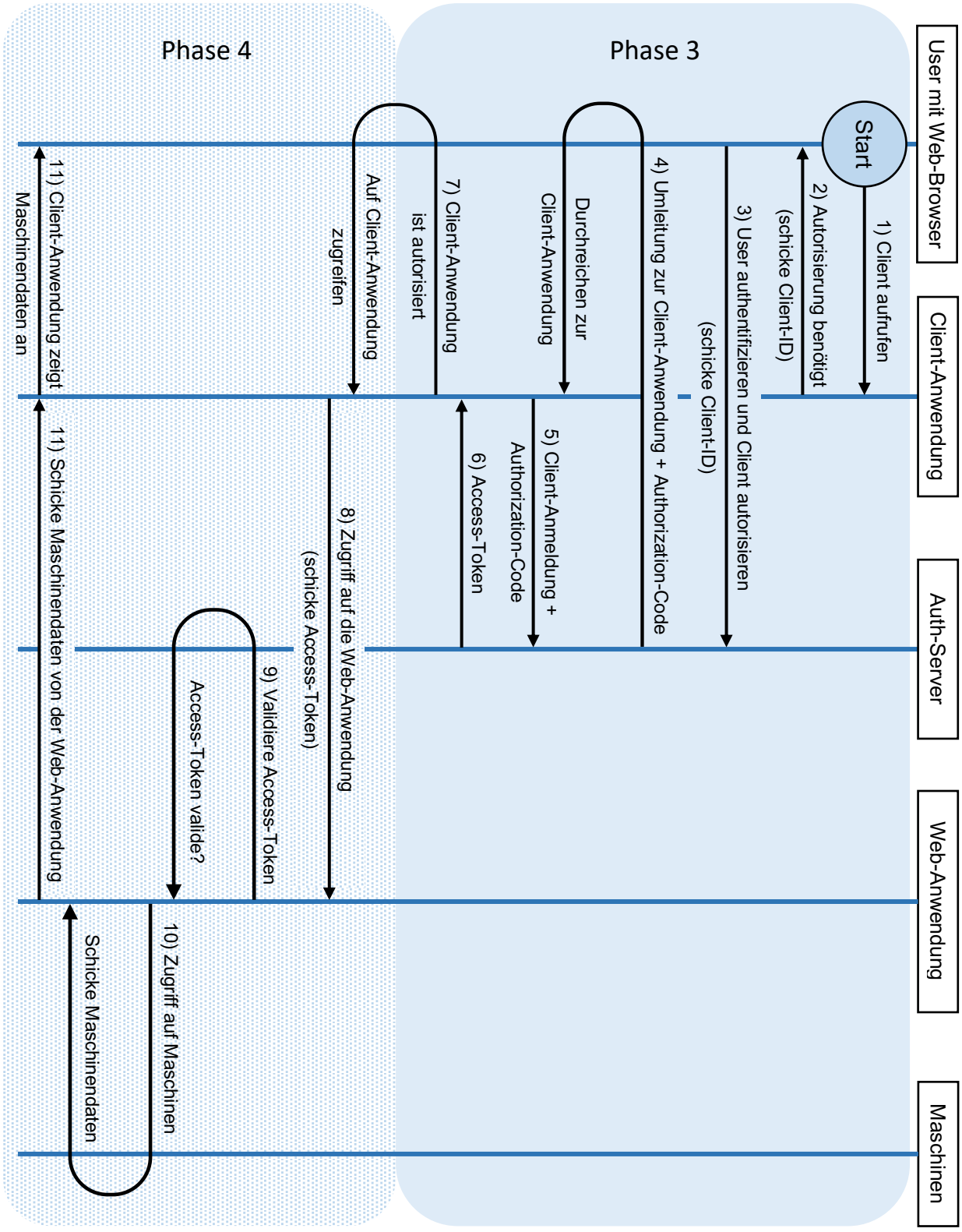

Abb. 6: Phase 4 – Der Zugriff auf die Web-Anwendung (eigene Abbildung)

Schritt 10: Web-Anwendung erhält Zugriff auf Maschinendaten

Die Web-Anwendung greift auf die durch sie überwachten Maschinen zu. Die Maschinen schicken der Web-Anwendung ihre Maschinendaten.

Schritt 11: Client zeigt Maschinendaten an

Die Web-Anwendung schickt der Client-Anwendung die geforderten Maschinendaten zu. Über die Client-Anwendung kann der User die Maschinendaten nutzen.

Solange der Access-Token nicht abgelaufen ist (valide ist), muss Phase 3 nicht wiederholt durchlaufen werden. Nur Phase 4 wird bei jeder Anfrage von neuen Maschinendaten wiederholt.[71]

OIDC läuft prinzipiell genauso ab wie OAuth 2.0.[72] Die wenigen Abweichungen im Ablauf von OIDC werden in Kapitel 3.3 beschrieben. Darüber hinaus unterscheidet sich OIDC von OAuth 2.0 auch dadurch, dass OIDC mit dem OAuth-2.0-Access-Token auch einen „ID-Token“ an die Client-Anwendung zurückschickt.[73] Der ID-Token macht aus der nicht User-bezogenen M2M-Kommunikation mit OAuth 2.0 eine User-bezogene H2M-Kommunikation mit OIDC.[74] Auch bei den am Ablauf beteiligten Akteuren unterscheidet sich OIDC von OAuth 2.0 (siehe das nachfolgende Kapitel 3.2).

3.2 Statisches Modell der Akteure von OIDC

OIDC nutzt weitgehend die gleichen Abläufe wie OAuth 2.0. In Abbildung 7 werden die Akteure von OAuth 2.0 und ihre Pendants bei OIDC dargestellt. Bei den Akteuren liegt ein Unterschied darin, dass bei OIDC die Web-Anwendung die beiden bei OAuth 2.0 getrennten Rollen „Client-Anwendung“ und „Web-Anwendung“ umfasst (siehe Abbildung 7).

Die Web-Anwendung besteht aus einem „Front-End“ und einem „Back-End“. Das Front-End (FE) ist die Benutzeroberfläche, die über einen Web-Browser abrufbar ist. Das Back-End (BE) ist die Komponente der Web-Anwendung, die das Front-End ausliefert und auf die Daten von den angebundenen Fertigungsmaschinen zugreift. Bei OIDC übernimmt das Front-End die Rolle der Client-Anwendung und das Back-End übernimmt die Rolle der Web-Anwendung. Das Front-End der Web-Anwendung ist also das Portal, über das der

71 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 48.

72 Vgl. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, a. a. O., abgerufen am 18.02.2021, S. 12, 17, 20. Vgl. auch Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 8.

73 Abbildung 18 und Abbildung 19 in Anhang B.2 zeigen den Unterschied zwischen einer HTTP-Nachricht mit ID-Token (Abbildung 19) und ohne ID-Token (Abbildung 18).

74 Vgl. Lodderstedt, Thorsten: OpenID Connect: Login mit OAuth Teil 1 – Grundlagen, a. a. O., abgerufen am 25.01.2021, S. 2.

User sich die Daten von angebundenen Fertigungsmaschinen ansehen kann und das Back-End der Web-Anwendung liefert diese Daten an das Front-End aus.

OIDC spezifiziert wie OAuth 2.0 auch die Rollen „User mit Web-Browser“ und „Auth-Server“ (siehe Abbildung 7). Der Auth-Server führt die Anmeldung des Users und alle damit verbundenen Aktionen für die Web-Anwendung durch. Der Auth-Server bei OIDC entspricht dem Auth-Server bei OAuth 2.0. Mit OIDC können Dienstleister eigene Auth-Server für bestimmte Web-Anwendungen eines bestimmten Kunden betreiben. Mit OIDC kann ein Dienstleister ebenso Auth-Server von dritten Dienstleistern wie z. B. Apple oder Google einsetzen. In der vorliegenden Arbeit wird angenommen, dass die Authentifizierungsmethode OIDC mit einem eigenen (nicht von Dritten) Auth-Server arbeitet.

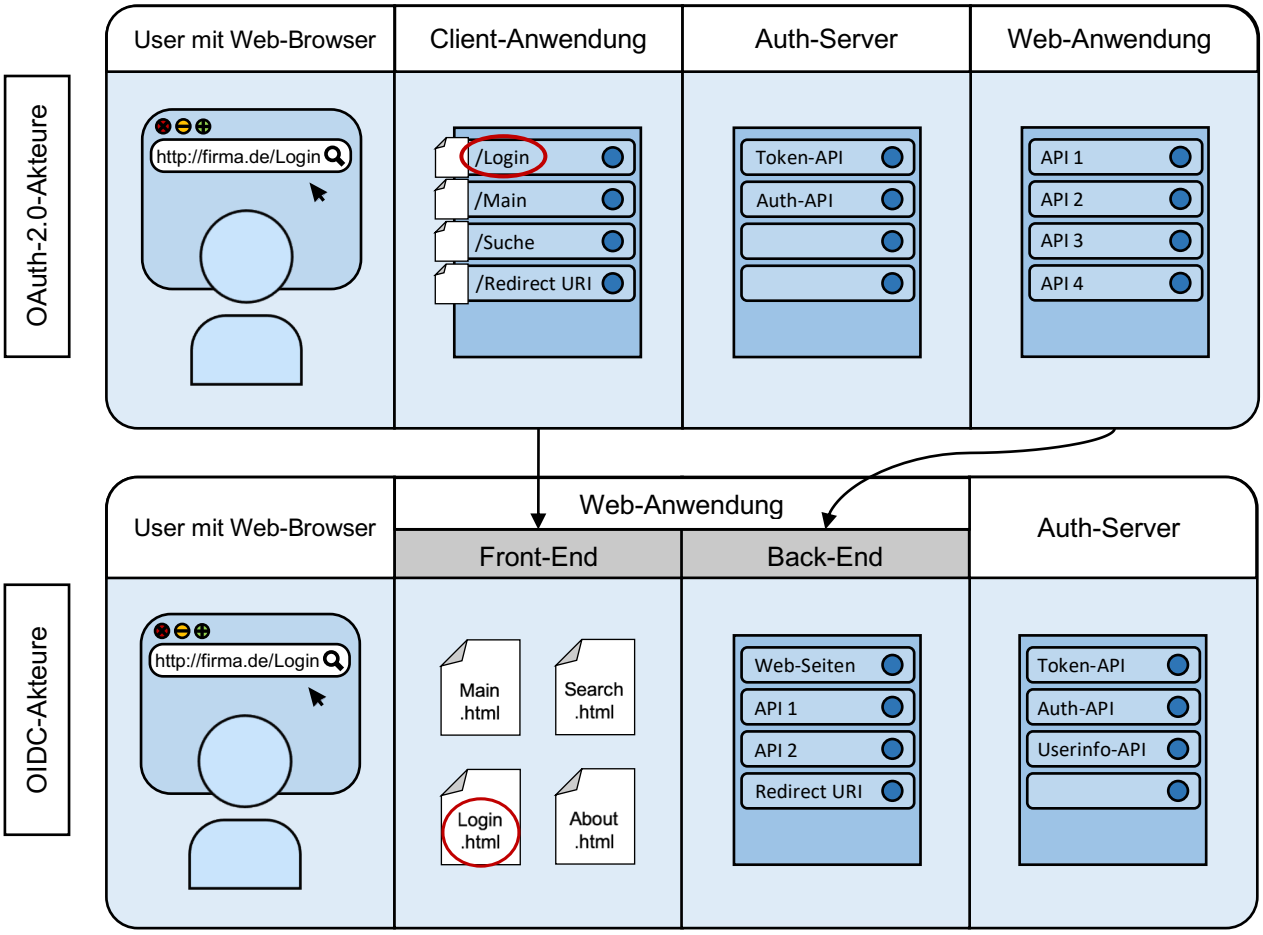

Abb. 7: Die Akteure von OAuth 2.0 und ihre Pendants in OIDC (eigene Abbildung)

Der User ist der Eigentümer seines Accounts und der damit verbundenen Daten. Er ist in der Lage, sich zu authentifizieren und ist autorisiert, auf die Maschinendaten zuzugreifen,

die die Web-Anwendung liefert.[75] In der vorliegenden Arbeit sind der „User“ und der „Useragent“, also der Web-Browser des Users, ein gemeinsamer Akteur. Dieser Akteur entspricht dem Akteur „User mit Web-Browser“ bei OAuth 2.0 (siehe Abbildung 7).

3.3 Dynamisches Modell der Abläufe von OIDC

Die Phasen, die für den Einsatz von OAuth 2.0 durchlaufen werden müssen, müssen auch für den Einsatz von OIDC durchlaufen werden. Die Phasen 3 und 4 unterscheiden sich bei OIDC von OAuth 2.0. In Phase 3 unterscheiden sich OAuth 2.0 und OIDC ausschließlich dadurch, dass in Schritt 6 ein Access-Token mit ID-Token und angepasstem scope-Parameter[76] (siehe Abbildung 19 in Anhang B.2) verschickt wird. In Abbildung 8 wird der Ablauf der Phase 3 von OIDC dargestellt.

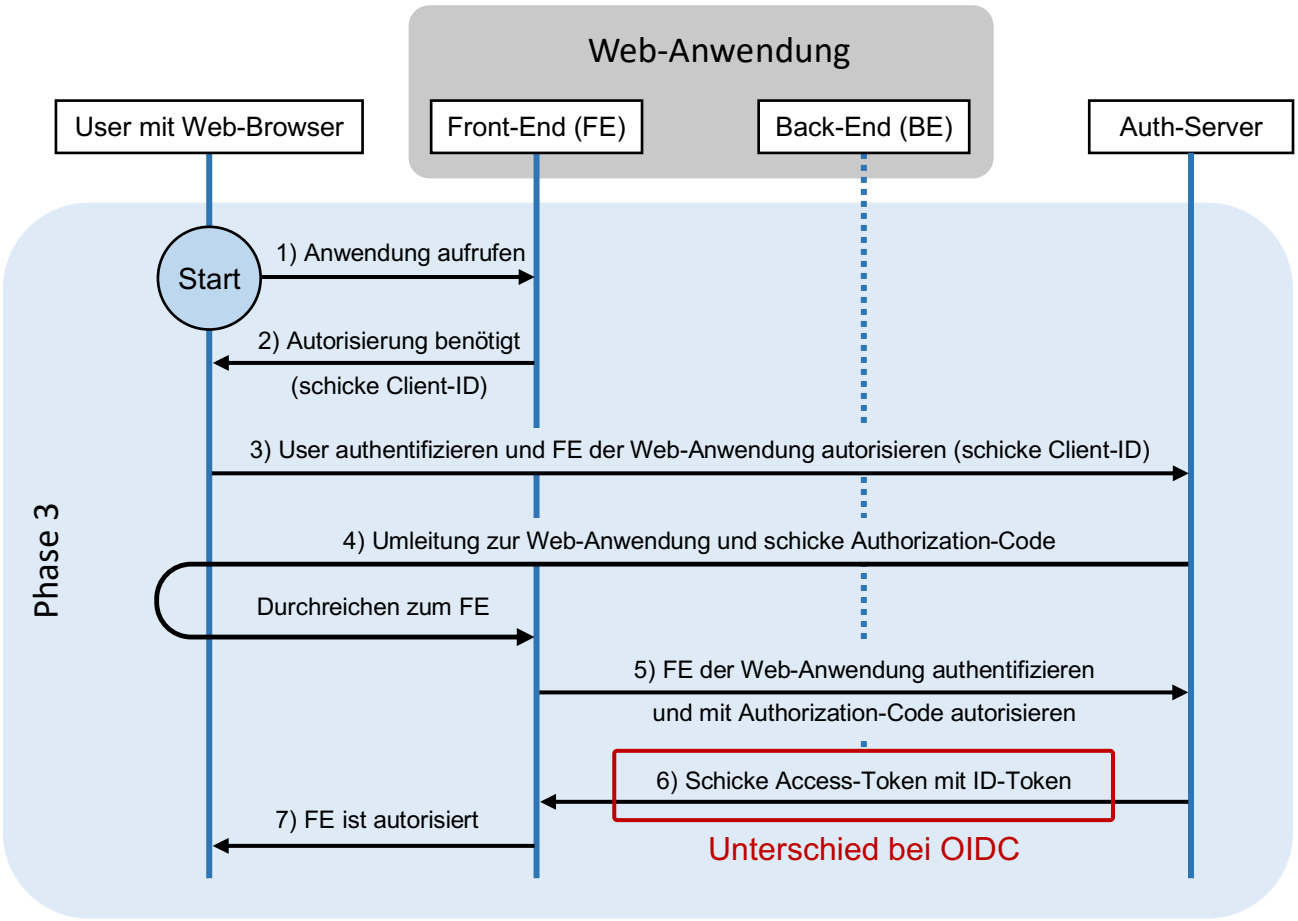

Abb. 8: Phase 3 – Der OIDC-Authentifizierungsablauf (eigene Abbildung)

75 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 6, 15.
76 In Anhang B.2 werden Access- und ID-Tokens sowie der scope-Parameter detailliert beschrieben.

Mit OIDC kann genau wie bei OAuth 2.0 in Phase 4 auf die Daten der angebundenen Fertigungsmaschinen zugegriffen werden. Im Unterschied zu OAuth 2.0 kann im OIDC-Ablauf von Phase 4 aber auch auf User-Daten wie Name und E-Mail-Adresse eines Users zugegriffen werden.[77] Abbildung 9 zeigt diesen Sachverhalt.

Die Phase 4a von OIDC entspricht der in Abbildung 6 dargestellten Phase 4 von OAuth. Die Phase 4b zeigt, wie mit OIDC auf User-Daten zugegriffen wird. Phase 4b kommt nur bei OIDC vor und nicht bei OAuth 2.0.

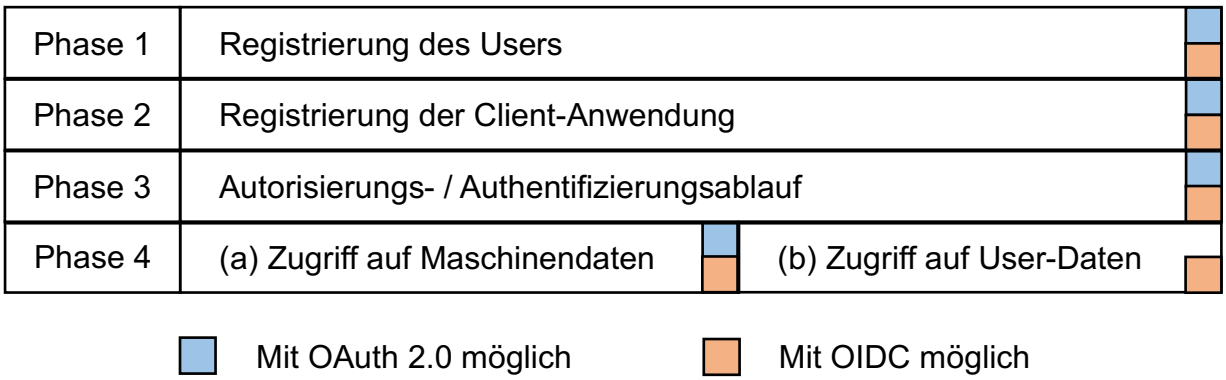

Abb. 9: Die Phasen zur Nutzung von OAuth 2.0 und OIDC (eigene Abbildung)

Nachfolgend werden die Schritte von Phase 4b von OIDC detailliert beschrieben.[78] In Abbildung 10 ist dieser Ablauf graphisch dargestellt.

Schritt 7: Front-End (FE) der Web-Anwendung aufrufen

Wenn Phase 3 abgeschlossen ist, wird der User auf das Front-End der Web-Anwendung umgeleitet.

Schritt 8: Zugriff auf die Userinfo-API des Auth-Servers mit Access-Token

Das Front-End der Web-Anwendung fügt den Access-Token in den Authorization-Header[79] der HTTP-Nachricht an Auth-Server ein.

77 Vgl. Lodderstedt, Thorsten: OpenID Connect: Login mit OAuth Teil 1 – Grundlagen, a. a. O., abgerufen am 25.01.2021, S. 2f.

78 Vgl. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, a. a. O., abgerufen am 18.02.2021, S. 38ff.

79 In Anhang A befindet sich eine kurze Einführung in die Themen „HTTP“ und „HTTPS“. Dort gibt es auch eine Erläuterung zum „Header“ einer HTTP-Nachricht.

Schritt 9: Validiere Access-Token

Der Auth-Server prüft in seiner Datenbank, ob es den übergegebenen Access-Token gibt und die mitgeschickten Parameter[80] valide Werte enthalten.

Schritt 10: Schicke User-Daten und zeige sie im Front-End der Web-Anwendung an

Der Auth-Server schickt der Web-Anwendung die geforderten User-Daten zu. Über das Front-End kann der User die User-Daten nutzen.

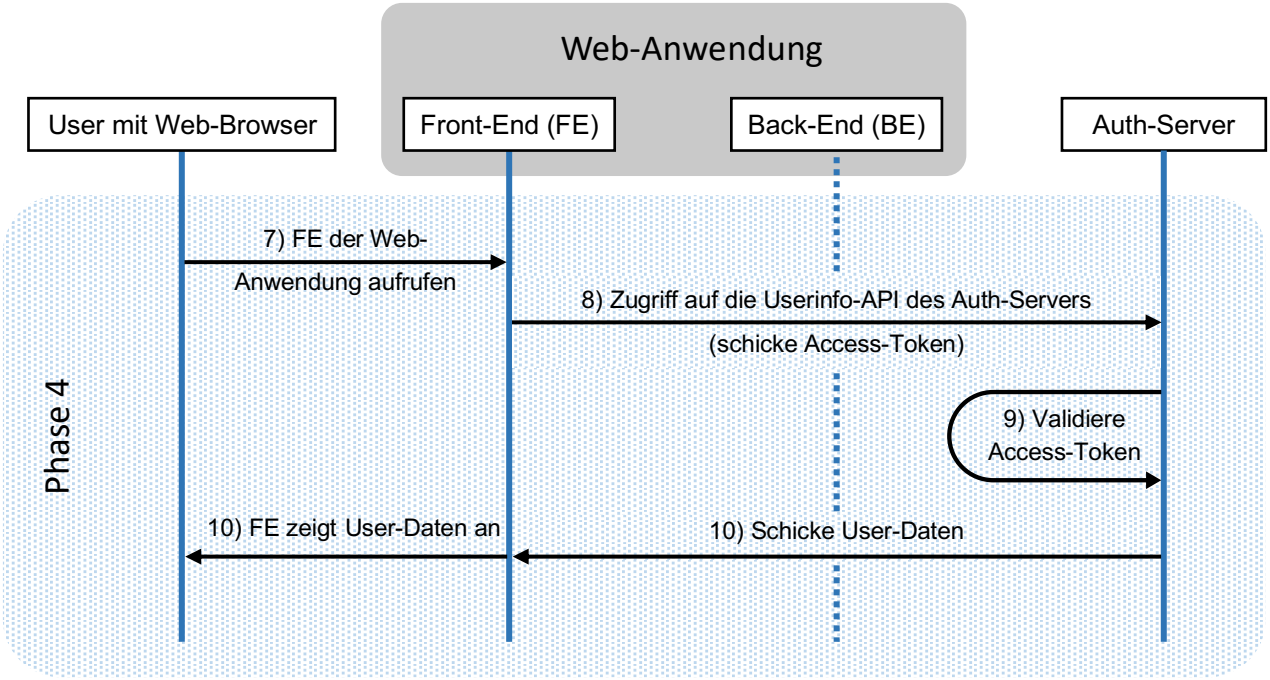

Abb. 10: Phase 4 – Zugriff auf User-Daten mit OIDC (eigene Abbildung)

Mit Phase 4a von OIDC kann eine Web-Anwendung Maschinendaten über das Back-End abrufen und im Front-End anzeigen. Die Phase 4a läuft genauso ab, wie die Phase 4 bei OAuth 2.0 (siehe Abbildung 6). OIDC erweitert OAuth 2.0 durch die Phase 4b, mit der eine Web-Anwendung User-Daten anzeigen kann. Die User-Daten werden vom Auth-Server bereitgestellt und vom Front-End der Web-Anwendung angezeigt.

80 In Anhang B.2 ist näher erläutert, was diese Parameter sind und welche Werte sie haben müssen.

3.4 Abgleich des Kriterienkatalogs mit OIDC

In diesem Kapitel wird gezeigt (siehe Tabelle 5), welche Kriterien des Kriterienkatalogs aus Kapitel 2.4 die Authentifizierungsmethode OIDC erfüllt (●), teilweise erfüllt (○) oder gar nicht erfüllt (leeres Feld). Bei den Kriterien, deren Grund für die Bewertung der Authentifizierungsmethode nicht selbsterklärend ist, werden textliche Erläuterungen gegeben. Eine Bewertung ist „selbsterklärend", wenn die Definition des Kriteriums in Kapitel 2.4, die Definition der Akteure einer Authentifizierungsmethode in Kapitel 3.2 und/oder die Funktionsweise der Authentifizierungsmethode in Kapitel 3.3 die Bewertung eindeutig belegen. Bei den anderen Kriterien sind in der Spalte „Begründung" die betreffenden Stichworte vermerkt, die auf den Grund der betreffenden Bewertung hinweisen.

S2 – Resistent gegen gezielte Nachahmung:

Das Kriterium S2 ist bei OIDC prinzipiell erfüllt. Denn bei OIDC werden Passwörter für die Authentifizierung des Users eingesetzt und keine personenbezogenen Daten abgefragt. In der Praxis enthalten „schlechte" Passwörter von Usern jedoch oft personenbezogene Daten wie z. B. das Geburtsdatum des Users. OIDC lässt ein solches „Fehlverhalten" zu; deshalb wird das Kriterium S2 durch OIDC nur bedingt erfüllt.

S5 – Resistent gegen Datenlecks bei anderen Auth-Servern:

Hinzu kommt, dass User häufig ihre Passwörter wiederverwenden.[81] Deshalb sind Datenlecks bei anderen Auth-Servern problematisch, bei denen der User auch einen User-Account besitzt. Denn dadurch könnte das Passwort des Users für die Web-Anwendungen bekannt werden. Das tritt nur auf, wenn der User seine Passwörter über verschiedene Auth-Server und Web-Anwendungen wiederverwendet. Deshalb wird das Kriterium S5 nur teilweise von OIDC erfüllt.

S10 – Keine Verknüpfbarkeit von Accounts:

In der vorliegenden Arbeit ist OIDC als Authentifizierungsmethode definiert, die mit einem eigenen Auth-Server arbeitet, der nur für diese Web-Anwendungen eingesetzt wird. Deshalb müssen User für jede Web-Anwendung mit OIDC einen Account mit neu-

81 Vgl. Bonneau, Joseph et al.: The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes, a. a. O., abgerufen am 25.01.2021, S. 8.

em Passwort erstellen. So können User-Accounts von zwei Web-Anwendungen nicht miteinander in Verbindung gebracht werden. Das Kriterium S10 wird von OIDC erfüllt.

U1, U2 und U11:

Aus dem gleichen Grund wie bei Kriterium S10, werden die Kriterien U1 (Kein Erinnerungsvermögen), U2 (Skalierbar für Nutzer) und U11 (Single Sign-On) von OIDC nicht erfüllt.

U5 – Zeit-effizient:

Passwörter müssen Standards wie z. B. ein Mindestmaß an Komplexität und Länge erfüllen, um als sicher zu gelten. In der Realität wählen User aber häufig Passwörter, die diese Standards nicht erfüllen. „Sichere" Passwörter sind aufwendiger für den User (Erzeugung und Eingabe) als einfache, personenbezogene, „unsichere" Passwörter. OIDC lässt ein solches Fehlverhalten zu; deshalb wird Kriterium U5 nur teilweise erfüllt.

A1 – Vernachlässigbare Kosten pro User:

OIDC benötigt auf der Seite des Users keine zusätzliche Hard- und Software. Der User muss keinen finanziellen Aufwand betreiben, um OIDC einzusetzen. Für den Auth-Server muss keine zusätzliche Software kostenpflichtig angeschafft werden. Als Software-System für den Auth-Server kann z. B. ein „Identity- & Access-Management-System" (IAM) wie z. B. „Keycloak" eingesetzt werden. Keycloak ist auch im kommerziellen Einsatz für den Betreiber kostenfrei.[82] Zudem ist Keycloak gut dokumentiert, ausgereift und hat einen großen Funktionsumfang.[83] Für den Auth-Server wird zusätzliche Hardware benötigt. Die Kosten dafür können kalkulatorisch auf die User umgelegt werden. In der Praxis werden Auth-Server-Systeme i. d. R. für große User-Zahlen betrieben (bis zu mehreren Tausend). Solange diese große Anzahl an Usern stabil bleibt, belaufen sich

82 Vgl. Koserwal, Abhishek: Keycloak: Core concepts of open source identity and access management, Red Hat Inc. (Hrsg.), 11. Dezember 2019, https://developers.redhat.com/blog/2019/12/11/keycloak-core-concepts-of-open-source-identity-and-access-management/, abgerufen am 01.03.2021. Vgl. auch Apache Software Foundation (Hrsg.): Apache License Version 2.0, Januar 2004, https://www.apache.org/licenses/LICENSE-2.0.txt, abgerufen am 01.03.2021, S. 2.

83 Vgl. Klassen, David: Eine Identität für alles mit Keycloak, Heise Medien GmbH & Co. KG (Hrsg.), 19. September 2017, https://m.heise.de/developer/artikel/Eine-Identitaet-fuer-alles-mit-Keycloak-3834525.html?seite=all, abgerufen am 01.03.2021, S. 2, 8. Vgl. auch Koserwal, Abhishek: Keycloak: Core concepts of open source identity and access management, a. a. O., abgerufen am 01.03.2021.

die Investitionen für Hardware und Software des Auth-Server-Systems nur auf sehr geringe Beträge pro User. Das Kriterium A1 gilt daher als erfüllt.[84]

A2 – Server-kompatibel und A4 – Ausgereift:

OIDC wurde bereits 2014 standardisiert und wird seither von vielen Auth-Servern als Authentifizierungsmethode bereitgestellt (ca. 56 zertifizierte Auth-Server und Services, Stand 2020) und von noch mehr Web-Anwendungen eingesetzt.[85] Die breite und langjährige Anwendung in der Praxis hat dazu geführt, dass OIDC für alle gängigen Server-Betriebssysteme und IAM-Systeme verfügbar ist.[86] Kriterium A2 ist daher erfüllt. OIDC ist die dritte Generation von Authentifizierungsmethoden der OpenID Foundation. Seit der ersten Generation aus dem Mai 2006 wurden permanent Verbesserungen und Anpassungen an die Erkenntnisse aus der Praxis an OIDC vorgenommen, sodass OIDC heute als ausgereift gilt.[87] OIDC erfüllt somit das Kriterium A4.

A3 – Browser-kompatibel und A6 – Mobile Endgeräte:

Da OIDC auf der Client-Seite keine spezifische Technologie erfordert, kann OIDC mit allen aktuellen Web-Browsern auf allen gängigen Endgeräten eingesetzt werden. Die Kriterien A4 und A6 gelten daher als erfüllt.

84 Nur wenn die Anzahl der User stark skaliert wird, können z. B. sprungfixe Investitionen die Kosten pro User in die Höhe treiben.

85 Vgl. Lodderstedt, Thorsten: OpenID Connect: Login mit OAuth Teil 1 – Grundlagen, a. a. O., abgerufen am 25.01.2021, S. 1. Vgl. auch OpenID Foundation (Hrsg.): Certified OpenID Connect Implementations, a. a. O., abgerufen am 15.02.2021.

86 Vgl. OpenID Foundation (Hrsg.): Certified OpenID Connect Implementations, a. a. O., abgerufen am 15.02.2021.

87 Vgl. OpenID Foundation (Hrsg.): OpenID Connect FAQ and Q&As, a. a. O., abgerufen am 01.03.2021. Für weitere Informationen zu der Geschichte von OIDC siehe Anhang B.1.

●: Kriterium ganz erfüllt ○: Kriterium teilweise erfüllt Leer: Kriterium nicht erfüllt			Begründung	OIDC
Sicherheitsniveau	S1	Resistent gegen physische Überwachung	Passwort	
	S2	Resistent gegen gezielte Nachahmung		○
	S3	Resistent gegen Raten	Passwort	
	S4	Resistent gegen internes Ausspähen	Passwort	
	S5	Resistent gegen Datenlecks		○
	S6	Resistent gegen Phishing	Passwort	
	S7	Resistent gegen Diebstahl	Keine Hardware	●
	S8	Keine dritten Auth-Server	Definition OIDC	●
	S9	Zustimmung des Users	Definition OIDC	●
	S10	Keine Verknüpfbarkeit von Accounts		●
	S11	Keine Übertragungen von Geheimnissen	Passwort	
	S12	Geheimnisse nicht speichern	Passwort	
Usability	U1	Kein Erinnerungsvermögen	Passwort	
	U2	Skalierbar für Nutzer	Passwort	
	U3	Keine zusätzliche Hardware	Keine Hardware	●
	U4	Kein physischer Aufwand	Passwort	
	U5	Zeit-effizient		○
	U6	Niedrige False-Reject-Rate	Passwort	●
	U7	Faktor wiederherstellbar	Passwort	●
	U8	Niedriger Migrationsaufwand für den User	Passwort	●
	U9	Geheimnisse nicht erneuern	Passwort	
	U10	Barrierefrei	Passwort	●
	U11	Single Sign-On		
Admin.-Aufwand	A1	Vernachlässigbare Kosten pro User		●
	A2	Server-kompatibel		●
	A3	Browser-kompatibel		●
	A4	Ausgereift		●
	A5	Nicht proprietär	Definition OIDC	●
	A6	Mobile Endgeräte		●

Tabelle 5: Abgleich des Kriterienkatalogs mit OIDC

4 Authentifizierungsmethode „FIDO2“

4.1 Grundlagen von FIDO2

FIDO2 ist eine Authentifizierungsmethode der zweiten Generation, entwickelt von der FIDO Alliance. Die erste Generation der Authentifizierungsmethoden der FIDO Alliance aus dem Dezember 2014 besteht aus der passwortlosen Authentifizierungsmethode FIDO UAF (Universal Authentication Factor) und der 2-Faktor-Authentifizierungsmethode FIDO U2F (Universal 2nd Factor). Die im April 2018 veröffentlichte Authentifizierungsmethode FIDO2 vereint die Funktionalität von FIDO UAF und FIDO U2F.[88] FIDO2 kann als zweiter Faktor für eine andere Authentifizierungsmethode oder als passwortlose 1-Faktor-Authentifizierungsmethode eingesetzt werden.[89] In der vorliegenden Arbeit wird die passwortlose 1-Faktor-Authentifizierung durch FIDO2 mit OIDC verglichen. Im Ausblick (Kapitel 5) wird auch auf FIDO2 in Kombination mit anderen Authentifizierungsmethoden eingegangen

Alle Authentifizierungsmethoden der FIDO Alliance verzichten auf den Einsatz von Passwörtern und setzen stattdessen asymmetrische Kryptographie ein.[90] Mit asymmetrischen Schlüsselpaaren sichert FIDO2 den Datenverkehr ab und authentifiziert den User. Ein asymmetrisches Schlüsselpaar ist eindeutig und nur auf eine bestimmte Web-Anwendung anwendbar.[91]

Damit eine Web-Anwendung FIDO2 einsetzen kann, müssen der Web-Server der Web-Anwendung und der Web-Browser des Users die „WebAuthn-API“ unterstützen. WebAuthn ist eine im März 2019 von der FIDO Alliance und dem W3C standardisierte Web-API für den Einsatz von FIDO2.[92]

Zudem benötigt ein User einen „Authentifikator“. Ein Authentifikator ist ein IT-System beim User, das die asymmetrischen Schlüsselpaare für FIDO2 generiert, aufbewahrt und

88 Vgl. FIDO Alliance (Hrsg.): History of FIDO Alliance, https://fidoalliance.org/overview/history/, abgerufen am 01.03.2021. Siehe Anhang D für weitere Informationen zur Geschichte der FIDO Alliance und von FIDO2.

89 Vgl. Eikenberg, Ronald: Schlüssel zum Glück – Was schon heute mit dem Passwort-Killer FIDO2 geht, in: C’t 19/18, Heise Medien GmbH & Co. KG (Hrsg.), S. 21.

90 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 30.

91 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 30f.

92 Vgl. FIDO Alliance (Hrsg.): History of FIDO Alliance, a. a. O., abgerufen am 01.03.2021.

auf Nachrichten anwendet.[93] Solche Authentifikatoren bestehen i. d. R. aus einem „Secure Element“ und einer Steuerungs-Software.

Ein „Secure Element“ ist ein speziell abgesichertes Stück Hardware, das technisch nicht unbefugt ausgelesen werden kann. Auf diesem Secure Element werden nicht nur die asymmetrischen Schlüsselpaare mit Zusatzinformationen gespeichert, sondern auch eine für den Authentifikator eindeutige, geheime Zeichenfolge.[94] Authentifikatoren können in Computern oder mobilen Endgeräten integriert sein oder als externer „Sicherheitsschlüssel“ vorliegen.[95]

Interne Authentifikatoren sind in die Hardware eines Computers oder mobilen Endgeräts integriert. Das Secure Element wird bei internen Authentifikatoren „Trusted Platform Module“ (TPM) genannt. Bei internen Authentifikatoren kommuniziert das TPM mit der WebAuthn-API im Web-Browser über das Betriebssystem (Operating System, OS) des Endgeräts beim User. Solche internen Authentifikatoren sind z. B. in neueren Windows- und Android-Endgeräten verbaut.[96]

Sicherheitsschlüssel sind externe Authentifikatoren, die über USB, NFC oder Bluetooth mit einem Endgerät verbunden sind. Sicherheitsschlüssel kommunizieren bei FIDO2 über das Client-to-Authenticator-Protocol-2 (CTAP2) mit der WebAuthn-API im Web-Browser des Users. So kann ein Sicherheitsschlüssel mit der WebAuthn-API auf einem Web-Server kommunizieren. Externe Authentifikatoren wie z. B. USB-Sticks oder Smartphones können mit mehreren Computern oder mobilen Endgeräten benutzt werden.[97] Yubico, Goldengate und Feitian stellen z. B. externe Sicherheitsschlüssel für FIDO2 her.[98]

93 Vgl. Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, Humboldt-Universität zu Berlin – Mathematisch-Naturwissenschaftliche Fakultät – Institut für Informatik (Hrsg.), https://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2020-04/SAR-PR-2020-04_.pdf, abgerufen am 30.12.2020, S. 11.

94 Vgl. Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 11. Vgl. auch Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 30.

95 Vgl. Eikenberg, Ronald: Schlüssel zum Glück – Was schon heute mit dem Passwort-Killer FIDO2 geht, a. a. O., S. 20f.

96 Vgl. Lapscheck, Niklas; Schick, Lukas; Schwickert, Axel: FIDO2 – Grundlagen, Prinzipien und praktische Anwendung, in: Arbeitspapiere WI, Nr. 2/2021, Hrsg.: Professur BWL – Wirtschaftsinformatik, Justus-Liebig-Universität Gießen 2021, S. 20.

97 Vgl. Lapscheck, Niklas; Schick, Lukas; Schwickert, Axel: FIDO2 – Grundlagen, Prinzipien und praktische Anwendung, a. a. O., S. 20.

98 Vgl. Eikenberg, Ronald: Login per Finger – Goldengate Security Keys für FIDO2, in: C’t 20/11, Heise Medien GmbH & Co. KG (Hrsg.), S. 85. Vgl. auch Yubico.com (Hrsg.): FIDO2 passwordless authentication, https://www.yubico.com/authentication-standards/fido2/, abgerufen am 03.03.2021.

Bei FIDO2 wird zur Registrierung eines Users bei einer Web-Anwendung ein asymmetrisches Schlüsselpaar aus der Domain der Web-Anwendung und dem Geheimnis des Authentifikators eines Users erzeugt.[99] Der öffentliche Schlüssel dieses asymmetrischen Schlüsselpaars wird an die Web-Anwendung geschickt und der private Schlüssel wird mit zusätzlichen Informationen wie z. B. einer User-ID nur auf dem Authentifikator des Users abgelegt.[100]

Die Authentifizierung bei der Anmeldung und bei der Registrierung eines Users mit FIDO2 läuft in einem asymmetrischen „Challenge-Response-Verfahren" ab.[101] Für das Verständnis von FIDO2 ist es hilfreich, die Funktionsweise von Challenge-Response-Verfahren zu kennen. Nachfolgend wird diese Funktionsweise daher erläutert.

Challenge-Response-Verfahren weisen die Kenntnis von einem Geheimnis nach, ohne dieses Geheimnis preiszugeben. Asymmetrische Challenge-Response-Verfahren basieren auf asymmetrischen Schlüsselpaaren. Die Geheimnisse sind daher die privaten Schlüssel der betreffenden asymmetrischen Schlüssel-Paare. Bei Challenge-Response Verfahren muss die Client-Anwendung ihren privaten Schlüssel und der authentifizierende Partner (Auth-Server) den öffentlichen Schlüssel der Client-Anwendung kennen.[102]

Bei asymmetrischen Challenge-Response-Verfahren werden die zwei kryptographischen Methoden „Verschlüsselung" und „Signatur" eingesetzt. Bei „Verschlüsselung" werden Nachrichten vom Sender mit dem öffentlichen Schlüssel des Empfängers verschlüsselt und können mit dem privaten Schlüssel des Empfängers entschlüsselt werden.[103] Eine Signatur hingegen ist eine Zeichenfolge, die entstehet, wenn eine Nachricht des Senders mit dem privaten Schlüssel des Senders verschlüsselt und an den Empfänger verschickt

99 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 32. Wie genau ein solches Schlüsselpaar erzeugt wird, ist für die vorliegende Arbeit nicht relevant

100 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 30. Vgl. auch Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 15f.

101 Vgl. Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 3.

102 Vgl. Hildebrandt, Knut: Konzeption von Authentifizierungsmechanismen für anonyme Dienste und Realisierung eines ausgewählten Verfahrens im Anonymisierungsnetzwerk Tor, Otto-Friedrich-Universität Bamberg – Fakultät Wirtschaftsinformatik und Angewandte Informatik – Lehrstuhl für Praktische Informatik (Hrsg.), 18. Januar 2007, https://www.uni-bamberg.de/fileadmin/uni/fakultaeten/wiai_lehrstuehle/praktische_informatik/Dateien/Forschung/Tor/hildebrandt-authentication.pdf, abgerufen am 05.03.2021, S. 40-43.

103 Vgl. Hildebrandt, Knut: Konzeption von Authentifizierungsmechanismen für anonyme Dienste und Realisierung eines ausgewählten Verfahrens im Anonymisierungsnetzwerk Tor, a. a. O., abgerufen am 05.03.2021, S. 42.

wird. Der Empfänger wendet den öffentlichen Schlüssel des Senders auf die Signatur an. Auf Grund des eindeutigen mathematischen Zusammenhangs zwischen privatem und öffentlichen Schlüssel kann der Empfänger prüfen, ob die Signatur von Sender stammt.[104] Die einzelnen Schritte eines asymmetrischen Challenge-Response-Verfahrens mit Signaturen werden anschließend textlich und mit Abbildung 11 detailliert beschrieben.[105]

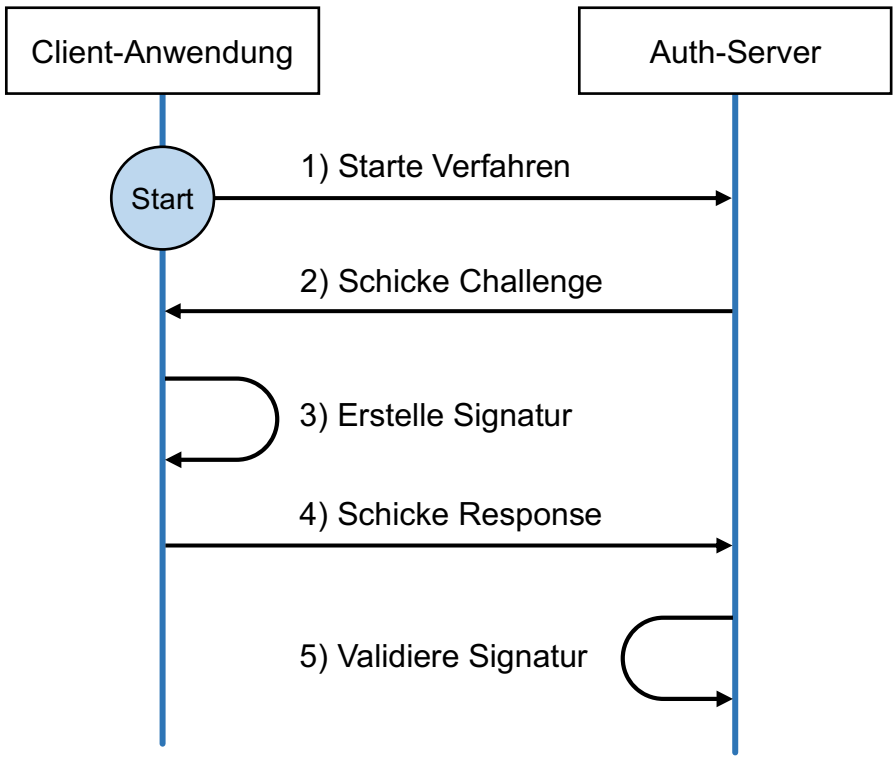

Abb. 11: Asym. Challenge-Response-Verfahren mit Signatur (eigene Abbildung)

Schritt 1: Starte Verfahren

Um ein asymmetrisches Challenge-Response-Verfahren mit Signaturen zu starten, schickt die Client-Anwendung dem Auth-Server eine Nachricht und signalisiert damit, dass die Client-Anwendung sich authentifizieren will.

Schritt 2: Schicke Challenge

Der Auth-Server generiert eine Zufallszahl als Challenge. Diese Challenge schickt der Auth Server an die Client-Anwendung.

104 Vgl. O. V.: Digitale Signatur, in: Elektronik-Kompendium.de, Patrick Schnabel (Hrsg.), https://www.elektronik-kompendium.de/sites/net/1910131.htm, abgerufen am 05.03.2021.

105 Vgl. Hildebrandt, Knut: Konzeption von Authentifizierungsmechanismen für anonyme Dienste und Realisierung eines ausgewählten Verfahrens im Anonymisierungsnetzwerk Tor, a. a. O., abgerufen am 05.03.2021, S. 42f.

Schritt 3: Erstelle Signatur

Die Client-Anwendung wendet ihren privaten Schlüssel auf die Challenge an. Die Client-Anwendung „signiert“ damit die Challenge.

Schritt 4: Schicke Response

Die Client-Anwendung schickt die signierte Challenge (die „Signatur“) als Response an der Auth-Server.

Schritt 5: Validiere Response

Der Auth-Server wendet den öffentlichen Schlüssel der Client-Anwendung auf die signierte Challenge (die „Signatur“) an. Auf Grund des eindeutigen mathematischen Zusammenhangs zwischen öffentlichem und privatem Schlüssel eines Schlüsselpaars kann der Auth-Server erkennen, dass die bei ihm eingegangene Signatur mit dem privaten Schlüssel der Client-Anwendung erstellt wurde.

Mit diesem Challenge-Response-Verfahren wird also sichergestellt, dass der öffentliche und der private Schlüssel ein asymmetrisches Schlüsselpaar bilden. Damit dies als „Authentifizierung“ einer Client-Anwendung dienen kann, muss das Schlüsselpaar von einer anerkannten Institution „zertifiziert“ sein. Um ein Schlüsselpaar zu zertifizieren, schickt die Client-Anwendung dem Auth-Server ein „Zertifikat“. Ein Auth-Server kann durch Nachfragen bei der Institution, die das Zertifikat ausgestellt hat, nachprüfen, wem das Schlüsselpaar gehört.

FIDO2 nutzt die geschilderte Challenge-Response-Mechanik zur Authentifizierung von Usern einer Web-Anwendung. Die beteiligten Akteure im Ablauf von FIDO2 werden in Kapitel 4.2 und die Abläufe von FIDO2 in Kapitel 4.3 genauer erläutert.

4.2 Statisches Modell der Akteure von FIDO2

In den Ablauf von FIDO2 sind vier Akteure eingebunden. Diese Akteure sind: „Auth-Server“, „Web-Anwendung im Web-Browser“, „Authentifikator“ und „User“.[106] In Abbildung 12 sind die Akteure von FIDO2 als Übersicht dargestellt.

106 Vgl. Lapscheck, Niklas; Schick, Lukas; Schwickert, Axel: FIDO2 – Grundlagen, Prinzipien und praktische Anwendung, a. a. O., S. 18.

FIDO2 kann mit einem Auth-Server betrieben werden, der genau wie bei OIDC unabhängig von dem Web-Server einer Web-Anwendung agiert. Die Funktionalität des Auth-Servers kann aber auch im Back-End der Web-Anwendung auf dem Web-Server integriert werden. In der vorliegenden Arbeit wird davon ausgegangen, dass ein Auth-Server eingesetzt wird, der unabhängig von dem Web-Server der Web-Anwendung läuft und nicht durch Dritte kontrolliert wird. Der Auth-Server muss die WebAuthn-API implementieren, um mit dem Browser des Users per FIDO2 zu kommunizieren.

Bei FIDO2 sind die Web-Anwendung und der Web-Browser ein Akteur (bei OIDC sind der Web-Browser und der User ein Akteur; siehe Kapitel 3.2). Dieser Akteur ist für die Kommunikation mit dem Auth-Server zuständig. Der Web-Browser muss die WebAuthn-API implementieren, um mit dem Auth-Server per FIDO2 zu kommunizieren.[107]

Der Authentifikator ist bei FIDO2 ein zentraler Akteur. Ein Authentifikator ist ein IT-System beim User, das die asymmetrischen Schlüsselpaare für FIDO2 generiert, aufbewahrt und auf Nachrichten anwendet.[108] Authentifikatoren können in Computern oder mobilen Endgeräten integriert sein oder als externe Sicherheitsschlüssel vorliegen.[109] Interne Authentifikatoren sind in die Hardware eines Computers oder mobilen Endgeräts integriert. Interne Authentifikatoren kommunizieren über das Betriebssystem des Endgeräts beim User mit der Web-Authn-API im Web-Browser. Externe Authentifikatoren, die über USB, NFC oder Bluetooth mit einem Endgerät verbunden sind, kommunizieren über das Client-to-Authenticator-Protocol-2 (CTAP2) mit der WebAuthn-API im Web-Browser.[110] In der vorliegenden Arbeit wird FIDO2 mit einem externen Authentifikator betrachtet.

Der User einer Web-Anwendung muss bei FIDO2 den Authentifikator „freischalten", damit über diesen Authentifikator die Authentifizierung des Users vorgenommen werden kann. Das Freischalten kann dabei auf verschiedene Weisen erfolgen:[111]

1. Authentifikator mit Knopf: Ein Authentifikator muss bei FIDO2 immer mit einer physischen Interaktion des Users mit dem Authentifikator freigeschaltet werden. Dazu

107 Vgl. Lapscheck, Niklas; Schick, Lukas; Schwickert, Axel: FIDO2 – Grundlagen, Prinzipien und praktische Anwendung, a. a. O., S. 19.

108 Vgl. Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 11.

109 Vgl. Eikenberg, Ronald: Schlüssel zum Glück – Was schon heute mit dem Passwort-Killer FIDO2 geht, a. a. O., S. 20f.

110 Vgl. Lapscheck, Niklas; Schick, Lukas; Schwickert, Axel: FIDO2 – Grundlagen, Prinzipien und praktische Anwendung, a. a. O., S. 20.

111 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 31.

haben Authentifikatoren einen Knopf, den der User drücken muss, um den Authentifikator freizuschalten. Ein solcher Authentifikator kann von jeder Person eingesetzt werden, die den Authentifikator in der Hand hat.

2. Authentifikator mit PIN: Ein Authentifikator mit PIN erweitert einen Authentifikator mit Knopf um eine lokale Authentifizierung des Users mit einer PIN. Ein User muss den Knopf auf dem Authentifikator drücken und seine PIN eingeben, um den Authentifikator freizuschalten.
3. Authentifikator mit Biometrie: Ein Authentifikator mit biometrischem Faktor erweitert einen Authentifikator mit Knopf um eine lokale Authentifizierung des Users mit einem biometrischen Faktor wie z. B. einem Fingerabdrucks- oder Netzhaut-Scan. Um den Authentifikator freizuschalten, muss ein User den Knopf auf dem Authentifikator drücken und sich mit dem biometrischen Faktor lokal authentifizieren.

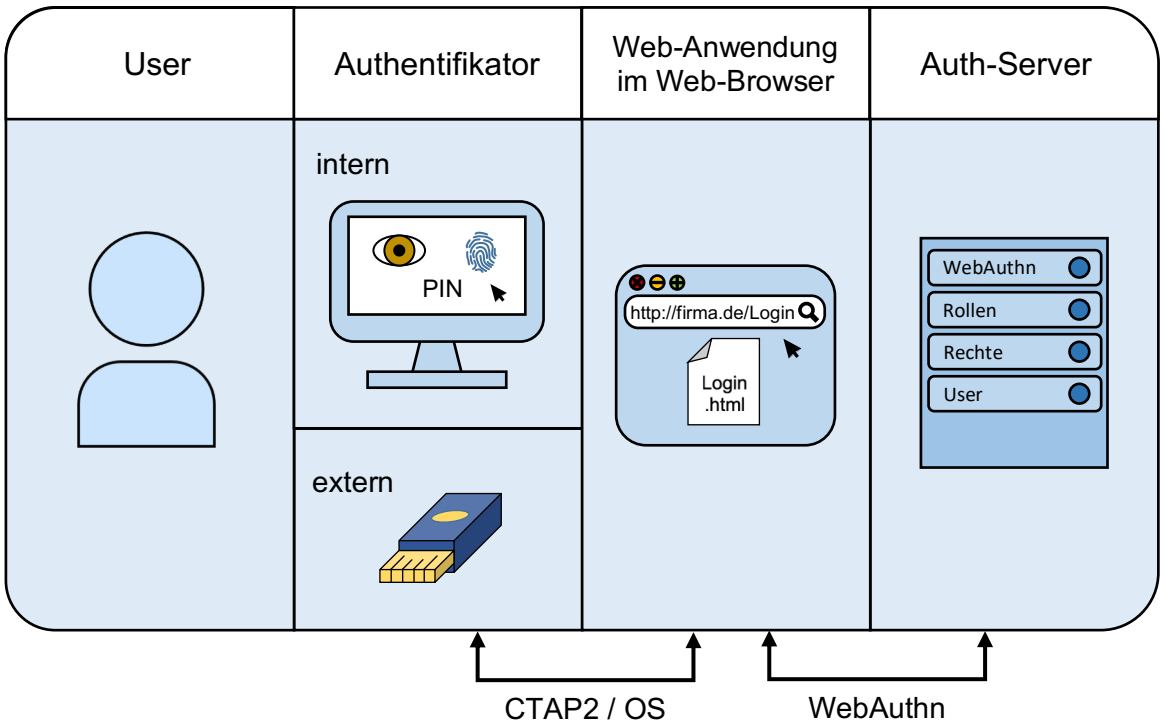

Abb. 12: Die Akteure von FIDO2 (eigene Abbildung)

Die Authentifizierung bei einem Authentifikator ist lokal, da die PIN oder der biometrische Faktor nur lokal vom Authentifikator beim User abgeglichen wird. Die geheimen Daten des Users werden nicht über das Internet übertragen.

Authentifikatoren mit lokaler Authentifizierung per PIN oder Biometrie teilen dem Auth-Server mit, dass ein authentifizierter User sich anmeldet („User Verification“, UV). Ein

Auth-Server kann eine UV verlangen (siehe Kap. 4.3). Authentifikatoren mit Knopf weisen lediglich nach, dass eine Person sich anmelden will („User Present“, UP).[112] Wie die Authentifizierung mit FIDO2 abläuft, zeigt das nachfolgende Kapitel 5.3 im Detail.

4.3 Dynamisches Modell der Abläufe von FIDO2

Damit ein User auf eine mit FIDO2 gesicherte Web-Anwendung zugreifen kann, müssen die folgenden vier Phasen durchlaufen werden:

Phase 1: Authentifikator einrichten

Für FIDO2 wird ein Authentifikator benötigt. Authentifikatoren, die den User lokal authentifizieren, müssen eingerichtet werden, bevor sie benutzt werden können. Die PIN, der biometrische Faktor o. ä. müssen zuerst festgelegt werden.

Phase 2: Registrierung des Users

Damit ein User sich bei einer Web-Anwendung anmelden kann, muss er sich bei der Web-Anwendung zunächst registrieren. Sollte der User schon einen Account bei der Web-Anwendung besitzen, muss der User FIDO2 als neue Authentifizierungsmethode hinzufügen. Dazu werden die gleichen Schritte durchlaufen, wie bei einer erstmaligen Registrierung des Users.

Phase 3: Authentifizierungsablauf

Damit ein User auf eine Web-Anwendung zugreifen kann, muss er sich bei der Web-Anwendung anmelden. Eine Anmeldung beinhaltet die Authentifizierung des Users. In dieser Phase findet die Authentifizierung des Users mit FIDO2 statt.

Phase 4: Zugriff auf die Web-Anwendung

Der User ist bei einer Web-Anwendung registriert und angemeldet. In dieser Phase greift ein User auf die Web-Anwendung zu.

Das Einrichten eines FIDO2-Authentifikators (Phase 1) unterscheidet sich je nach Authentifikator, Freischaltungs-Typ (Knopf, PIN oder Biometrie) und Betriebssystem vom Endgerät des Users. Wesentlich ist, dass der FIDO2-Authentifikator mit dem Endgerät ver-

112 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 31.

bunden wird (über USB, Bluetooth oder NFC) und die lokale Authentifizierung eingerichtet wird. Das heißt, dass der User eine PIN oder einen biometrischen Faktor auf dem Authentifikator hinterlegen muss. Wenn der Authentifikator keine lokale Authentifizierung unterstützt, muss der User den Authentifikator lediglich mit dem Endgerät verbinden.

Die Registrierung (Phase 2) und die Authentifizierung (Phase 3) eines Users mit FIDO2 läuft nach dem in Kapitel 4.1 geschilderten asymmetrischen Challenge-Response-Verfahren ab.[113] Nachfolgend werden alle Schritte der Registrierung eines Users (Phase 2) von FIDO2 detailliert beschrieben.[114] In Abbildung 13 sind die Schritte der Phase 2 in einem Ablaufdiagramm dargestellt.

Schritt 1: Web-Anwendung aufrufen und Registrierung anfordern

Der User ruft die Web-Anwendung auf. Die Web-Anwendung merkt, dass der User nicht angemeldet ist und leitet den User auf den Auth-Server weiter. Da der User noch keinen Account bei der Web-Anwendung besitzt, muss er einen neuen Account registrieren. Dazu wählt der User zuerst einen Usernamen und gibt weitere User-Daten wie z. B. die E-Mail-Adresse ein.

Schritt 2: Schicke Challenge, User-Daten und Domain zum Authentifikator

Der Auth-Server generiert eine Challenge und schickt diese mit User-Daten (User-ID, Username und Anzeigename) und der Domain der Web-Anwendung über die Web-Authn-API an die Web-Anwendung im Web-Browser. Von dort wird alles über CTAP2 an den Authentifikator weitergeleitet.

Schritt 3: User-Interaktion und optionale Authentifizierung

Der User muss bei FIDO2 sein Einverständnis mit einem Authentifizierungsvorgang physisch mitteilen. Wenn das Licht am Authentifikator blinkt, muss der User den Knopf auf seinem Authentifikator drücken. Bei Authentifikatoren mit UV-Funktion muss der User anschließend die lokale Authentifizierung mit PIN oder Biometrie durchführen.

Schritt 4: Generiere und speichere Schlüsselpaar

Auf dem Authentifikator ist ein eindeutiges Geheimnis gespeichert. Ein Authentifikator generiert in Schritt 4 ein asymmetrisches Schlüsselpaar des Users aus diesem Geheim-

113 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 31.

114 Vgl. Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 14ff.

nis und der Domain der Web-Anwendung.[115] Der private Schlüssel des Users wird mit einer „Credential-ID", der Domain der Web-Anwendung, der User-ID und weiteren Daten[116] auf dem Authentifikator abgelegt.[117] Die Credential-ID identifiziert das Schlüsselpaar eines bestimmten Accounts des Users bei einer bestimmten Web-Anwendung auf dem Authentifikator und beim Auth-Server dieser Web-Anwendung (zur Erinnerung: Ein User benötigt bei FIDO2 für jede Web-Anwendung ein eigenes Schlüsselpaar). Die Domain der Web-Anwendung gibt den Gültigkeitsbereich des Schlüsselpaars an und die User-ID kann bei Bedarf zur Identifizierung des Users vom Authentifikator eingesetzt werden.

Schritt 5: Challenge signieren und Response verschicken

Die Challenge (wurde in Schritt 2 vom Auth-Server an die Web-Anwendung geschickt und per CTAP2 an den Authentifikator weitergeleitet) wird nun vom Authentifikator mit dem privaten Schlüssel des Users signiert. Diese Signatur wird mit dem öffentlichen Schlüssel des Users inklusive zugehörigem Zertifikat und der Credential-ID als Response über CTAP2 an den Web-Browser übermittelt. Die gesamte Response wird von der Web-Anwendung über die WebAuthn-API an den Auth-Server geschickt.[118]

Schritt 6: Validiere Response und speichere Daten

Der Auth-Server validiert die Response, indem er alle übergebenen Parameter[119] und die Signatur überprüft. Um die Signatur zu prüfen, wendet der Auth-Server den öffentlichen Schlüssel des Users auf die Signatur an und überprüft die Zertifizierung. Der Auth-Server speichert den öffentlichen Schlüssel mit der Credential-ID ab und ordnet den öffentlichen Schlüssel einem User-Account zu.

115 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, a. a. O., S. 32. Wie genau der Authentifikator das Schlüsselpaar generiert, wird in der vorliegenden Arbeit nicht betrachtet.

116 Für nähere Informationen zu den gespeicherten Daten siehe: Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 14ff.

117 Accounts können auf dem Authentifikator optional vom User benannt werden, um die Accounts unterscheiden zu können. In der vorliegenden Arbeit ist diese Funktionalität aber nicht relevant.

118 Einige Authentifikatoren haben „Signaturzähler" für Schlüsselpaar-Einträge. Diese Zähler werden mit jeder Signatur um eins hochgezählt. Diese Signaturzähler sind eine Sicherheitsmaßnahme, durch die das Klonen von Authentifikatoren verhindert wird. Die Signaturzähler werden in der Response an den Auth-Server übermittelt. Der Auth-Server speichert den Wert des Signaturzählers ab.

119 Besonders wichtig ist, dass das Schlüsselpaar mit der eindeutigen Credential-ID nicht bereits einem anderen User zugewiesen wurde. Für nähere Informationen zur Validierung der Response siehe: Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 14ff.

Schritt 7: Erfolgreiche Authentifizierung und Registrierung

Die Registrierung wurde erfolgreich abgeschlossen und der User wird auf die Web-Anwendung geleitet, wo er sich mit seinem User-Account anmelden kann.

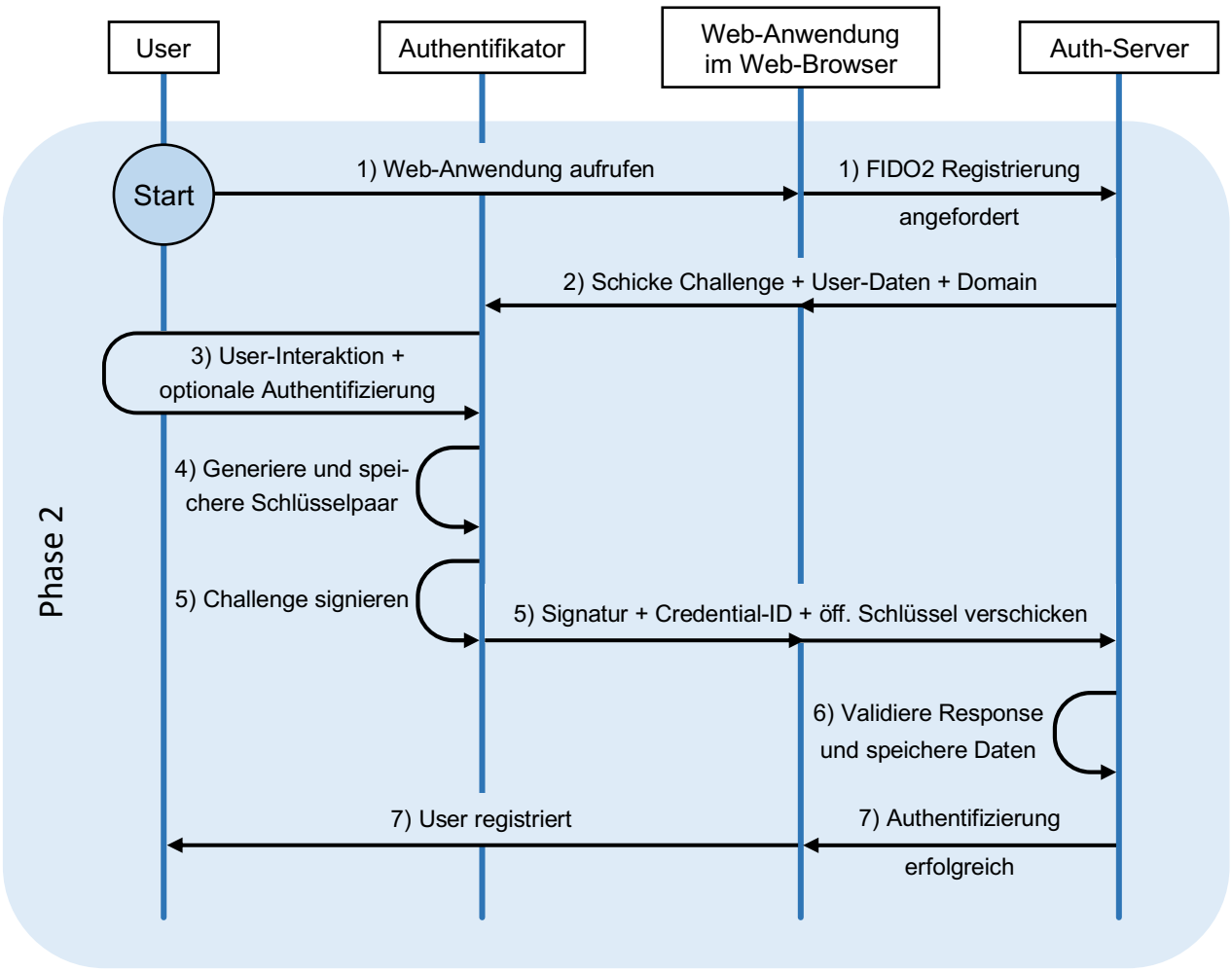

Abb. 13: Phase 2 – Die Registrierung des Users mit FIDO2 (eigene Abbildung)

Nachdem ein User sich registriert hat, kann er sich in Phase 3 mit seinem User-Account bei der Web-Anwendung anmelden, um auf diese zuzugreifen. Die einzelnen Schritte eines Anmeldevorgangs mit FIDO2 werden anschließend textlich und mit Abbildung 14 detailliert beschrieben.[120]

Schritt 1: Web-Anwendung aufrufen und Anmeldung anfordern

Der (registrierte) User ruft die Web-Anwendung auf und gibt seinen Username ein. Die Web-Anwendung leitet den (registrierten) User zum Auth-Server.

120 Vgl. Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 17-20.

Schritt 2: Schicke Challenge und weitere Daten zum Authentifikator

Der Auth-Server generiert eine Challenge. Die Challenge wird zusammen mit der Domain der Web-Anwendung, einer Liste erlaubter Credential-IDs für den User und der Nutzer-Verifikations-Anforderung (UV oder UP) über die WebAuthn-API an die Web-Anwendung im Web-Browser geschickt.[121] Über CTAP2 werden die Challenge und die Daten an den Authentifikator weitergeleitet.

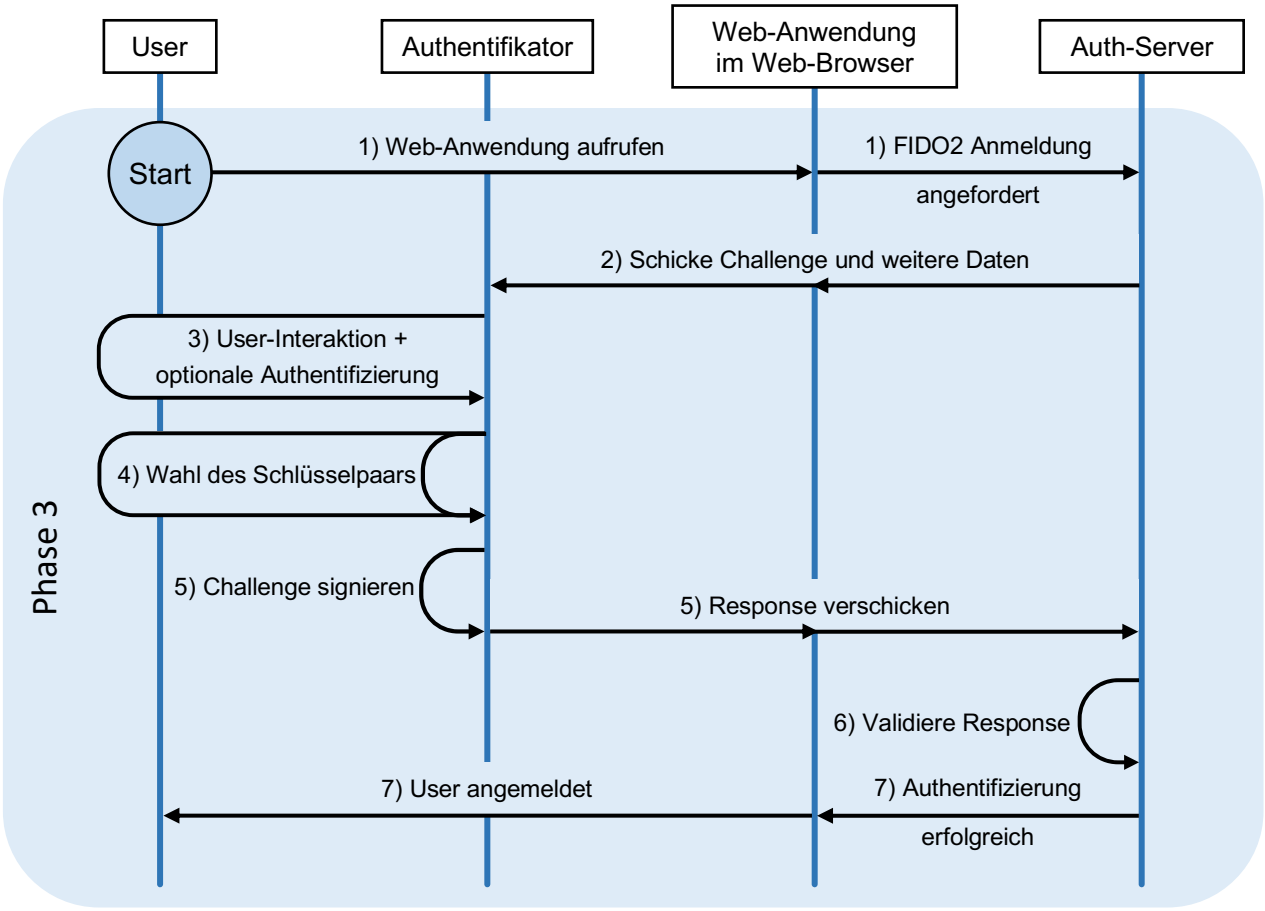

Abb. 14: Phase 3 – Die Anmeldung des Users mit FIDO2 (eigene Abbildung)

Schritt 3: User-Interaktion und optionale lokale Authentifizierung

Wie bei der Registrierung muss der User bei der Anmeldung mit FIDO2 sein Einverständnis mit einem Authentifizierungsvorgang physisch mitteilen. Wenn das Licht am Authentifikator blinkt, muss der User den Knopf auf seinem Authentifikator drücken. Die übergebene Nutzer-Verifikations-Anforderung des Auth-Servers muss erfüllt werden. Bei geforderter „User Verification“ (UV) muss der User sich lokal per PIN oder Biometrie authentifizieren.

121 Für nähere Informationen zu Parametern, die mit der Challenge verschickt werden siehe Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 17f.

Schritt 4: Wahl des Schlüsselpaars

Da der User in Schritt 1 seinen Usernamen eingegeben hat, kann der Authentifikator ohne Zutun des Users das Schlüsselpaar auswählen, das zu der Domain der Web-Anwendung und dem Usernamen gehört. Wenn der Username vorher nicht abfragt wird, muss der User einen der Accounts für die Domain der Web-Anwendung von dem Authentifikator auswählen. In der vorliegenden Arbeit muss der User den Account nicht manuell auf dem Authentifikator auswählen. Der Authentifikator kann den passenden Account selbst auswählen, da der Username vorher angegeben wurde. Der Authentifikator kennt damit die Credential-ID des User-Accounts, den der User nutzen will.

Schritt 5: Challenge signieren und Response verschicken

Der Authentifikator signiert die Challenge mit dem privaten Schlüssel des Users. Diese Signatur wird zusammen mit der Domain der Web-Anwendung und der Nutzer-Verifikations-Anforderung (UV oder UP) als Response über CTAP2 an die Web-Anwendung im Web-Browser übergeben.[122] Über die WebAuthn-API wird die Response an den Auth-Server weitergeschickt.

Schritt 6: Validiere Response

Der Auth-Server validiert die Response, indem er alle übergebenen Parameter[123] und die Signatur überprüft. Um die Signatur zu prüfen, wendet der Auth-Server den öffentlichen Schlüssel des Users auf die Signatur an.[124]

Schritt 7: Authentifizierung und Anmeldung erfolgreich

Der User wurde erfolgreich authentifiziert und angemeldet. Jetzt kann der User auf die Web-Anwendung zugreifen (Phase 4).

FIDO2 gibt nicht vor, wie der Zugriff auf die Web-Anwendung (Phase 4) erfolgt. FIDO2 stellt die Identität eines Users fest und ob dieser User überhaupt auf eine Web-Anwen-

122 Der öffentliche Schlüssel des Users liegt bereits durch die Registrierung beim Auth-Server vor.

123 Besonders wichtig ist, dass das Schlüsselpaar mit der eindeutigen Credential-ID beim Auth-Server vorhanden und dem User-Account zugeordnet ist. Zudem muss die in der Response übergebene Domain mit der Domain der Web-Anwendung übereinstimmen, die für das Schlüsselpaar bei der Registrierung festgelegt wurde. Für nähere Informationen zur Validierung der Response siehe Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 19f.

124 Bei diesem Anmeldevorgang (Phase 3) muss die Zertifizierung des Schlüsselpaars nicht mehr überprüft werden. Diese Überprüfung erfolgt bereits bei der Registrierung des Users (siehe Phase 2). Ein übergebener Signaturzähler muss immer größer als der bei dem Auth-Server gespeichert Signaturzähler sein. Anderenfalls wurde der Authentifikator möglicherweise dupliziert. Wenn der Auth-Server einen validen Signaturzähler übermittelt bekommt, speichert er diesen neuen Wert ab.

dung zugreifen darf („Authentifizierung“). FIDO2 lässt jedoch die Autorisierung des Users für eine Web-Anwendung offen. „Autorisierung“ beschreibt, welche Berechtigung der User für die Nutzung der Web-Anwendung hat. Für eine solche Autorisierung (nach der Authentifizierung des Users) muss neben FIDO2 ein weiteres Protokoll wie z. B. OIDC eingesetzt werden. Zur Erinnerung: Bei OIDC sind die Berechtigungen eines Users zur Nutzung einer Web-Anwendung an ein Access-Token geknüpft. OIDC lässt jedoch offen, wie die Authentifizierung eines Users erfolgt. In Kapitel 3 wurde dargelegt, wie bei OIDC die Authentifizierung bspw. mit Username und Passwort erfolgen kann, die durch den User selbst gewählt und über das offene Internet übertragen werden.

FIDO2 übernimmt also die Authentifizierung des Users, während OIDC die Autorisierung eines Users umsetzt. Die Kombination dieser beiden Verfahren bietet sich also an; mehr dazu in Kapitel 5.

4.4 Abgleich des Kriterienkatalogs mit FIDO2

In diesem Kapitel wird gezeigt (siehe Tabelle 6), welche Kriterien des Kriterienkatalogs aus Kapitel 2.4 die Authentifizierungsmethode FIDO2 erfüllt (●), teilweise erfüllt (○) oder nicht erfüllt (leeres Feld). Dabei wird FIDO2 in drei Varianten miteinander verglichen. Diese Varianten sind die drei in Kapitel 4.2 genannten Möglichkeiten, wie ein User einen FIDO2-Authentifikator freischalten kann: „Authentifikator mit Knopf“, „Authentifikator mit PIN“ und „Authentifikator mit Biometrie“.

Bei den Kriterien, deren Grund für die Bewertung der Authentifizierungsmethode nicht selbsterklärend ist, werden textliche Erläuterungen gegeben. Eine Bewertung ist „selbsterklärend“, wenn die Definition des Kriteriums in Kapitel 2.4, die Definition der Akteure einer Authentifizierungsmethode in Kapitel 4.2 und/oder die Funktionsweise der Authentifizierungsmethode in Kapitel 4.3 die Bewertung eindeutig belegen. Bei den anderen Kriterien sind in der Spalte „Begründung“ die betreffenden Stichworte vermerkt.

S1, S2 und S3:

Die Kriterien S1 (Resistent gegen physische Überwachung), S2 (Resistent gegen gezielte Nachahmung) und S3 (Resistent gegen Raten) sind für die FIDO2-Varianten mit Knopf und Biometrie alle erfüllt. Das liegt daran, dass ein User keine Passwörter oder PINs über eine Tastatur eingeben muss. Die zweite FIDO2-Variante hat eine PIN, die

den Authentifikator freischaltet. Unbefugte können Kenntnis von dieser PIN durch physische Überwachung (Kriterium S1) oder Raten (Kriterium S3) erhalten. Eine gezielte Nachahmung der Person durch die Kenntnis von persönlichen Daten des Users ist nur dann möglich, wenn die PIN des Users persönliche Daten enthält (siehe Kapitel 3.4). Deshalb ist für die zweite FIDO2-Variante („Authentifikator mit PIN“) das Kriterium S2 nur teilweise erfüllt.

S7 – Resistent gegen Diebstahl:

Da FIDO2 einen physischen Authentifikator benötigt, ist FIDO2 offensichtlich anfällig für den Diebstahl dieses Geräts. Die erste FIDO2-Variante erfüllt Kriterium S7 nicht, da jede Person, die sich den Authentifikator unrechtmäßig angeeignet hat, diesen einsetzen kann. Die beiden anderen FIDO2-Varianten erfüllen das Kriterium S7 teilweise, da die lokale Authentifizierung des Users mit PIN oder Biometrie vor Missbrauch durch Unbefugte schützt.

S10 – Keine Verknüpfbarkeit von Accounts:

User-Accounts bei kooperierenden Web-Anwendungen miteinander in Verbindung zu bringen, ist mit FIDO2 nur möglich, wenn es explizit durch den Auth-Server gefordert wird. Diese Forderung ist dann erfüllt, wenn für die Identifizierung des Users eine personenbezogene Information wie z. B. eine E-Mail-Adresse genutzt wird.[125] Bei Auth-Servern ist i. d. R. jedoch voreingestellt, dass die Identifizierung des Users mit der pro Web-Anwendung eindeutigen und unpersönlichen User-ID erfolgt. Die Verknüpfbarkeit von User-IDs ist daher nicht gegeben.[126] Deshalb erfüllen alle FIDO2-Varianten das Kriterium S10.

U1, U4 und U5:

FIDO2 setzt keine Passwörter ein, die sich ein User merken und in einem (bei sicheren Passwörtern) aufwendigen Prozess per Tastatur eingeben muss. Deshalb erfüllen die erste und die letzte FIDO2-Variante die Kriterien U1 (Kein Erinnerungsvermögen), U4 (Kein physischer Aufwand) und U5 (Zeit-effizient). Die zweite FIDO2-Variante setzt jedoch eine PIN ein, die sich ein User merken und mit der Tastatur erfassen muss. Dadurch erfüllt die zweite FIDO2-Variante die Kriterien U1 und U4 nicht. Das Kriterium

125 Auch der Username in Schritt 1 von Phase 3 kann eine solche personenbezogene Information sein.

126 Vgl. Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, a. a. O., abgerufen am 30.12.2020, S. 20.

U5 wird von der zweiten FIDO2-Variante aber erfüllt, da eine PIN aus meist vier Zeichen besteht, die in kurzer Zeit eingegeben werden kann.

U6 – Niedrige False-Reject-Rate:

FIDO2 ist eine Authentifizierungsmethode, die einfach einzusetzen ist, obwohl der Ablauf ungewohnt ist. Das liegt daran, dass der User im einfachsten Fall nur einen Knopf drücken muss (FIDO2-Varianten 1 und 3). Bei der zweiten FIDO2-Variante muss der User direkt nach dem Drücken des Knopfs eine PIN eingeben. Deshalb ist der User keine Quelle für viele falsche Zurückweisungen bei der Authentifizierung. Neben dem User kann jedoch auch der Authentifikator für falsche Zurückweisungen bei der Authentifizierung verantwortlich sein, wenn ein biometrischer Faktor nicht erkannt wird. Authentifikatoren von renommierten Unternehmen wie Yubico, Goldengate und Feitian setzen jedoch meist hochwertige und gut funktionierende Biometrie-Sensoren ein. Deshalb kann für alle FIDO2-Varianten das Kriterium U6 erfüllt werden.

U8 – Niedriger Migrationsaufwand für den User:

FIDO2 ist ein passwortloser Ansatz für die User-Authentifizierung. Der Aufwand für eine Migration zu FIDO2 wird dadurch erhöht, dass der User seinen Authentifikator bei jeder Web-Anwendung einrichten muss, die der User nutzen will. Der Aufwand für die Migration zu FIDO2 wird ganz wesentlich erhöht, wenn der User dem Rat von Experten folgt, und zwei oder mehr Authentifikatoren vorhält. Wenn ein Authentifikator verloren geht, kann er nicht ersetzt oder reaktiviert werden, wie es z. B. bei einem vergessenen Passwort der Fall ist (siehe Kriterium U7).[127]

U10 – Barrierefrei:

FIDO2 ist genauso barrierefrei, wie eine Passwort-basierte Authentifizierungsmethode. In beiden Fällen muss der User lediglich eine manuelle Eingabe tätigen. Deshalb wird Kriterium U10 von allen FIDO2-Varianten erfüllt.

A1 – Vernachlässigbare Kosten pro User:

Um FIDO2 einsetzen zu können, muss ein User einen Authentifikator besitzen. Authentifikatoren mit Knopf und Authentifikatoren mit PIN haben ein verhältnismäßig gerin-

127 Um eine Ausfallsicherheit zu haben, ist es ratsam, mindestens 2 Authentifikatoren bei einer Web-Anwendung zu registrieren (siehe Kriterium U7). Siehe: Schmidt, Jürgen; Eikenberg, Ronald: Passwort-Nachfolger FIDO2, in: C't 19/22, Heise Medien GmbH & Co. KG (Hrsg.), S. 169.

gen Preis (ca. 25-50 Euro pro Stück; Stand März 2021).[128] Authentifikatoren mit Biometrie-Sensor können deutlich mehr kosten (ca. 50-140 Euro pro Stück; Stand März 2021).[129] Ein User muss keine zusätzliche Software kostenpflichtig anschaffen. Die Kosten auf Seiten des Auth-Servers sind vergleichbar mit den Kosten für den Auth-Server bei OIDC (siehe Kapitel 3.4). Diese Kosten können kalkulatorisch auf die User umgelegt werden. In der Praxis werden Auth-Server-Systeme i. d. R. für große User-Zahlen betrieben (mehrere Tausend). Solange diese große Anzahl an Usern stabil bleibt, belaufen sich die Investitionen für Hardware und Software des Auth-Server-Systems nur auf sehr geringe Beträge pro User.[130] Die FIDO2-Varianten 1 und 2 können das Kriterium A1 wegen der verhältnismäßig geringen Kosten für den Authentifikator noch teilweise erfüllen. Die dritte FIDO2-Variante erfüllt wegen des hohen Preises des Authentifikators mit Biometrie-Sensor das Kriterium A1 nicht.

A4 – Ausgereift:

FIDO2 ist eine verhältnismäßig junge Authentifizierungsmethode. FIDO2 wurde jedoch schon von großen Unternehmen wie z. B. Microsoft und Google implementiert. FIDO2 geht aus den zwei ausgereiften Authentifizierungsmethoden U2F und UAF hervor. Die zentrale WebAuthn-API wurde durch das W3C standardisiert. In den Standard-Dokumenten zu der WebAuthn-API sind die Funktionen ausführlich dokumentiert. User können auf Web-Seiten wie „WebAuthn.io“ die Authentifizierungsmethode FIDO2 jederzeit selbst testen und Entwickler können mit Open-Source-Bibliotheken in verschiedenen Programmiersprachen FIDO2 selbst implementieren. Deshalb wird das Kriterium A4 von allen FIDO2-Varianten erfüllt.

128 Vgl. Yubico.com (Hrsg.): Yubikey 5 Series – For professionals, https://www.yubico.com/de/store/#for-professionals, abgerufen am 07.03.2021.

129 Vgl. Feitian Technologies Co., Ltd. (Hrsg.): FIDO-Security-Key, https://www.ftsafe.com/store/product-category/fido-security-key/#productCategory, abgerufen am 07.03.2021.

130 Nur wenn die Anzahl der User stark skaliert wird, können z. B. sprungfixe Investitionen die Kosten pro User in die Höhe treiben.

●: Kriterium ganz erfüllt O: Kriterium teilweise erfüllt Leer: Kriterium nicht erfüllt			Begründung	FIDO2 (Knopf)	FIDO2 (PIN)	FIDO2 (Biometrie)
Sicherheitsniveau	S1	Resistent gegen physische Überwachung		●		●
	S2	Resistent gegen gezielte Nachahmung		●	O	●
	S3	Resistent gegen Raten		●		●
	S4	Resistent gegen internes Ausspähen	Authentifikator	●	●	●
	S5	Resistent gegen Datenlecks	Definition FIDO2	●	●	●
	S6	Resistent gegen Phishing	Domain-spezifisch	●	●	●
	S7	Resistent gegen Diebstahl			O	O
	S8	Keine dritten Auth-Server	Definition FIDO2	●	●	●
	S9	Zustimmung des Users	Definition FIDO2	●	●	●
	S10	Keine Verknüpfbarkeit von Accounts		●	●	●
	S11	Keine Übertragungen von Geheimnissen	Definition FIDO2	●	●	●
	S12	Geheimnisse nicht speichern	Definition FIDO2	●	●	●
Usability	U1	Kein Erinnerungsvermögen		●	O	●
	U2	Skalierbar für Nutzer	Keine Passwörter	●	●	●
	U3	Keine zusätzliche Hardware	Authentifikator			
	U4	Kein physischer Aufwand		●		●
	U5	Zeit-effizient		●	●	●
	U6	Niedrige False-Reject-Rate		●	●	●
	U7	Faktor wiederherstellbar	Authentifikator			
	U8	Niedriger Migrationsaufwand für den User				
	U9	Geheimnisse nicht erneuern	Keine Passwörter	●	●	●
	U10	Barrierefrei		●	●	●
	U11	Single Sign-On	Domain-spezifisch			
Admin.-Aufwand	A1	Vernachlässigbare Kosten pro User		O	O	
	A2	Server-kompatibel	Erst 3 Jahre alt	O	O	O
	A3	Browser-kompatibel	Implementiert	●	●	●
	A4	Ausgereift		●	●	●
	A5	Nicht proprietär	Internet-Standard	●	●	●
	A6	Mobile Endgeräte	Implementiert	●	●	●

Tabelle 6: Abgleich des Kriterienkatalogs mit FIDO2

5 Vergleich der Authentifizierungsmethoden

In diesem Kapitel werden die Erkenntnisse aus den Kapiteln 3 und 4 zusammengefasst. Dazu werden die Bewertungen der Authentifizierungsmethoden aus den Tabellen 5 und 6 in eine Vergleichsmatrix (Tabelle 7) überführt. Die Authentifizierungsmethoden können so anhand der Kriterien verglichen werden. Eine Authentifizierungsmethode kann ein Kriterium vollständig erfüllen (●), teilweise erfüllen (○) und nicht erfüllen (leeres Feld).

Nachfolgend wird nicht mehr auf die Begründung für die Bewertung einer Authentifizierungsmethode eingegangen, sondern nur noch auf die Unterschiede der Authentifizierungsmethoden. Dabei wird nicht jedes Kriterium einzeln betrachtet. Stattdessen werden die Kriterien-Kategorien „Sicherheitsniveau“, „Usability“ und „Administrationsaufwand“ und Kriterien-übergreifende Begründungen dargelegt.

Sicherheitsniveau:

In Bezug auf das Sicherheitsniveau ist jede der drei FIDO2-Varianten im Vorteil gegenüber OIDC. Das liegt besonders an dem passwortlosen Ansatz von FIDO2 und dem Einsatz von Passwörtern bei OIDC. Passwörter werden kritisiert, da sie entweder unsicher oder wegen ihrer Komplexität unbenutzbar sind.[131] Der Einsatz von Passwörtern wirkt sich besonders auf die Kriterien S1 bis S6 sowie S11 (Keine Übertragungen von Geheimnissen) und S12 (Geheimnisse nicht speichern) aus. Hier zeigt sich eindeutig, dass Authentifizierungsmethoden, die Passwörter oder PINs (FIDO2-Variante 2) nutzen, im Nachteil gegenüber passwortlosen Authentifizierungsmethoden sind.

Das Kriterium S7 (Resistent gegen Diebstahl) zeigt einen der häufigsten von Usern vorgebrachten Grund für Bedenken bei FIDO2[132]: „Was passiert, wenn mein FIDO2-Authentifikator verloren geht oder er gestohlen wird?“ Authentifizierungsmethoden mit zusätzlicher Hardware wie z. B. einem FIDO2-Authentifikator sind eben nicht resistent gegen Diebstahl. Hier ist OIDC gegenüber jeder FIDO2-Variante im Vorteil.

131 Vgl. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, in: C’t 19/18, Heise Medien GmbH & Co. KG (Hrsg.), S. 30.

132 Vgl. Schmidt, Jürgen; Eikenberg, Ronald: Passwort-Nachfolger FIDO2, in: C’t 19/22, Heise Medien GmbH & Co. KG (Hrsg.), S. 169. Vgl. auch Lyastani, Sanam et al.: Is FIDO2 the Kingslayer of User Authentication? A Comparative Usability Study of FIDO2 Passwordless Authentication, CISPA Helmholtz Center for Information Security, Saarbrücken, https://publications.cispa.saarland/3146/1/oakland20.pdf, abgerufen am 30.12.2020, S. 9.

Die Kriterien S8 (Keine dritten Auth-Server) und S10 (Keine Verknüpfbarkeit von Accounts) hängen vorwiegend von der Implementierung der jeweilige Authentifizierungsmethode ab. Die Definitionen der Authentifizierungsmethoden in der vorliegenden Arbeit führen dazu, dass FIDO2 in jeder Variante und OIDC vorteilhaft sind.

Bei der Betrachtung der Kriterien-Kategorie „Sicherheitsniveau“ als Ganzes fällt auf, dass FIDO2 in jeder Variante bei jedem Kriterium außer S7 (Resistent gegen Diebstahl) genauso gut wie oder besser als OIDC abschneidet. Die FIDO2-Varianten mit Knopf und mit Biometrie weisen zudem eine bessere Bilanz auf als die FIDO2-Variante mit PIN.

Usability:

Die Usability-Kriterien U1, U2, U4, U5, U8 und U9 hängen in der vorliegenden Arbeit stark davon ab, ob eine Authentifizierungsmethode passwortbasiert oder passwortlos funktioniert. Denn ein User muss sich mehrere (U2 – Skalierbar für Nutzer) hinreichend komplexe Passwörter merken (U1 – Kein Erinnerungsvermögen), manuell erfassen (U4 – Kein physischer Aufwand und U5 – Zeit-effizient) und zyklisch erneuern (U9 – Geheimnisse nicht erneuern), um die Sicherheit des User-Accounts zu erhalten. Bei diesen Kriterien liegt FIDO2 in jeder Variante wieder klar vorne. Ein User hat keinen Aufwand für die Migration von einer passwortbasierten Authentifizierungsmethode (Kriterium U8 – Niedriger Migrationsaufwand für den User) zu einer anderen passwortbasierten Authentifizierungsmethode. Die Migration zu einer passwortlosen Authentifizierungsmethode wie FIDO2 bringt für den User jedoch einen wesentlichen Migrationsaufwand mit sich. Deshalb ist in der vorliegenden Arbeit bezüglich Kriterium U8 OIDC im Vorteil gegenüber FIDO2. Mit weiterer Bekanntheit von FIDO2 nutzen mehr User diese Authentifizierungsmethode. Dadurch registrieren sich User seltener mit Passwörtern, die nachträglich auf FIDO2 migriert werden müssen; User registrieren sich gleich mit FIDO2.

Da FIDO2 von sich aus kein SSO (Kriterium U11 – Single Sign-On) ermöglicht[133] und OIDC in der vorliegenden Arbeit so definiert ist, dass SSO nicht möglich ist, erfüllen beide Authentifizierungsmethoden das Kriterium U11 nicht.

Die Kriterien U3 (Keine zusätzliche Hardware) und U7 (Faktor wiederherstellbar) hängen in der vorliegenden Arbeit von dem Einsatz zusätzlicher Hardware (U3) wie z. B.

133 FIDO2 kann mit einem Protokoll wie OIDC kombiniert werden. Diese Kombination kann SSO ermöglichen.

einem FIDO2-Authentifikator ab. Denn Authentifikatoren wie bei FIDO2 lassen sich nicht wiederherstellen oder kopieren (U7).[134] Passwörter oder andere Geheimnisse, die nicht an solitäre Hardware gebunden sind, können vergleichsweise einfach z. B. über eine E-Mail zurückgesetzt werden. OIDC als Authentifizierungsmethode ohne Authentifikatoren erfüllt somit die Kriterien U3 und U7, während FIDO2 sie nicht erfüllt.

Bei der Betrachtung der Kriterien-Kategorie „Usability" als Ganzes fällt auf, dass die Authentifizierungsmethoden FIDO2-Variante 1 und FIDO2-Variante 3 mehr Usability-Kriterien erfüllen oder zumindest teilweise erfüllen als OIDC und die FIDO2-Variante 2. Die FIDO2-Varianten 1 und 3 liegen bei der Kategorie „Usability" ungefähr gleich auf.

Administrationsaufwand:

Die Bewertung des Kriteriums A1 (Vernachlässigbare Kosten pro User) wird dadurch beeinflusst, ob ein User sich für den Einsatz einer Authentifizierungsmethode zusätzliche Hardware wie z. B. einen Authentifikator anschaffen muss. Denn die Kosten pro User beim Einsatz der in der vorliegenden Arbeit betrachteten Authentifizierungsmethoden unterscheiden sich lediglich durch die Kosten für einen FIDO2-Authentifikator. Alle FIDO2-Varianten sind hier mehr (FIDO2-Variant 3) oder weniger (FIDO2-Varianten 1 und 2) im Nachteil gegenüber OIDC. Dieser Nachteil von FIDO2 wird jedoch mit weiteren Entwicklungen (z. B. Smartphone als Authentifikator) zukünftig wegfallen.

Das Kriterium A2 (Server-kompatibel) wird von allen FIDO2-Varianten nur teilweise erfüllt, da FIDO2 noch verhältnismäßig jung ist. Es kann davon ausgegangen werden, dass FIDO2 mit zunehmender Bekanntheit zukünftig das Kriterium A2 erfüllen wird.

Bei der Betrachtung der Kriterien-Kategorie „Administrationsaufwand" als Ganzes zeigt sich, dass OIDC besser bewertet wird als alle FIDO2-Varianten. Zukünftig dürfte FIDO2 jedoch auch alle Administrationsaufwand-Kriterien erfüllen können und in der Kategorie „Administrationsaufwand" genauso gut bewertet werden wie OIDC.

Zusammengenommen zeigt sich, dass die FIDO2-Varianten mit Knopf und mit Biometrie besser bewertet werden als die Authentifizierungsmethoden OIDC und die FIDO2-Variante 2.

134 Vgl. Schmidt, Jürgen; Eikenberg, Ronald: Passwort-Nachfolger FIDO2, in: C't 19/22, Heise Medien GmbH & Co. KG (Hrsg.), S. 169. Vgl. auch Kruse, Malte: Biometriebasierte Authentifizierung mit WebAuthn, Humboldt-Universität zu Berlin – Mathematisch-Naturwissenschaftliche Fakultät – Institut für Informatik (Hrsg.), https://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2020-02/SAR-PR-2020-02_.pdf, abgerufen am 30.12.2020, S. 23.

●: Kriterium ganz erfüllt ○: Kriterium teilweise erfüllt Leer: Kriterium nicht erfüllt		OIDC	FIDO2-Var. 1	FIDO2-Var. 2	FIDO2-Var. 3
Sicherheitsniveau	S1 Resistent gegen physische Überwachung		●		●
	S2 Resistent gegen gezielte Nachahmung	○	●	○	●
	S3 Resistent gegen Raten		●		●
	S4 Resistent gegen internes Ausspähen		●	●	●
	S5 Resistent gegen Datenlecks	○	●	●	●
	S6 Resistent gegen Phishing		●	●	●
	S7 Resistent gegen Diebstahl	●		○	○
	S8 Keine dritten Auth-Server	●	●	●	●
	S9 Zustimmung des Users	●	●	●	●
	S10 Keine Verknüpfbarkeit von Accounts	●	●	●	●
	S11 Keine Übertragungen von Geheimnissen		●	●	●
	S12 Geheimnisse nicht speichern		●	●	●
Usability	U1 Kein Erinnerungsvermögen		●	○	●
	U2 Skalierbar für Nutzer		●	●	●
	U3 Keine zusätzliche Hardware	●			
	U4 Kein physischer Aufwand		●		●
	U5 Zeit-effizient	○	●	●	●
	U6 Niedrige False-Reject-Rate	●	●	●	●
	U7 Faktor wiederherstellbar	●			
	U8 Niedriger Migrationsaufwand für den User	●			
	U9 Geheimnisse nicht erneuern		●	●	●
	U10 Barrierefrei	●	●	●	●
	U11 Single Sign-On				
Admin.-Aufwand	A1 Vernachlässigbare Kosten pro User	●	○	○	
	A2 Server-kompatibel	●	○	○	○
	A3 Browser-kompatibel	●	●	●	●
	A4 Ausgereift	●	●	●	●
	A5 Nicht proprietär	●	●	●	●
	A6 Mobile Endgeräte	●	●	●	●

Tabelle 7: Vergleichsmatrix – OIDC und FIDO2

Da FIDO2 eine reine Authentifizierungsmethode ist und nicht die Autorisierung zum Zugriff auf eine Web-Anwendung regelt (siehe Kapitel 4.3), muss ein weiteres Protokoll diese Autorisierung umsetzen. Dazu bietet sich z. B. OIDC an. Denn OIDC spezifiziert die Authentifizierungsmethode für den User in Schritt 3 des Authentifizierungsablaufs (siehe Abbildung 8) nicht. Die Autorisierung des Users zum Zugriff auf eine Web-Anwendung wird bei OIDC mit Access-Tokens umgesetzt.

OIDC und FIDO2 ergänzen sich somit. Wenn FIDO2 mit OIDC kombiniert wird, kann auch das Kriterium U11 (Single Sign On) von FIDO2 erfüllt werden. Das kann jedoch dazu führen, dass die Kriterien S8 (Keine dritten Auth-Server) und S10 (Keine Verknüpfbarkeit von Accounts) nicht mehr erfüllt sind.

Anhang

A. HTTP und HTTPS

Die Kommunikation von Client und Server bei Web-Anwendungen funktioniert mit HTTP (Hypertext Transfer Protocol). HTTP ist ein zustandsloses Protokoll für die Übertragung von Daten über das Internet. „Zustandslos“ bedeutet, dass HTTP-Nachrichten nicht ohne Modifikationen in einen gemeinsamen Kontext gebracht werden können. Das ist für Web-Anwendungen mit User-Accounts aber notwendig, da eine Interaktion des Browsers mit dem Web-Server über HTTP sonst nicht mit dem Account eines Users in Verbindung gebracht werden kann.[135]

Dieses Problem wird durch Session-Tracking behoben. „Session-Tracking“ („Sitzungsverfolgung“) bezeichnet das Zuordnen von HTTP-Nachrichten zu einem Kontext. Dieser Kontext wird „Session“ („Sitzung“) genannt. Eine solche Session hat einen Zustand, der sich mit jeder HTTP-Nachricht ändern kann.[136]

HTTP-Nachrichten bestehen i. d. R. aus den Bestandteilen General-Header, Request-Line, Request-Header, Entity-Header und Entity-Body. Die Header-Elemente enthalten Meta-Informationen wie den verwendeten Zeichensatz, aber auch „Cookies“ (siehe unten), welche z. B. für das Session-Tracking genutzt werden können[137]. Die „Request-Line“ enthält die Informationen über die „HTTP-Methode“ (siehe unten), die URI einer Ressource und die verwendete Version von HTTP. Der „Entity-Body“ beinhaltet Parameter, die übergeben werden sollen. HTTP bietet die Möglichkeit, Anfragen mit fünf verschiedenen Methoden zu verschicken: GET, HEAD, POST, PUT und DELETE.[138] Für die vorliegende Arbeit sind nur GET und POST relevant. Diese werden nachfolgend näher beschrieben.

GET-Requests nutzen nur die Header-Elemente und die Request-Line einer HTTP-Nachricht. GET-Requests schicken eine Request an einen URI und erhalten die Ressource oder den Dienst mit dieser URI zurück. Dabei können über die URI Parameter als Parameter-Wert-Paare übergeben werden.[139]

135 Vgl. Balzert, Helmut: Lehrbuch der Software-Technik – Softwareentwicklung, a. a. O., S. 959.
136 Vgl. Balzert, Helmut: Lehrbuch der Software-Technik – Softwareentwicklung, a. a. O., S. 959.
137 Vgl. Balzert, Helmut: Lehrbuch der Software-Technik – Softwareentwicklung, a. a. O., S. 959.
138 Vgl. Balzert, Helmut: Lehrbuch der Software-Technik – Softwareentwicklung, a. a. O., S. 956.
139 Vgl. Balzert, Helmut: Lehrbuch der Software-Technik – Softwareentwicklung, a. a. O., S. 956.

Die Größe der URI ist begrenzt. Außerdem sollten vertrauliche Informationen nicht in die URI geschrieben werden, wo sie leicht einsehbar sind. Die POST Methode löst beide Probleme. POST-Requests nutzen alle Bestandteile einer HTTP-Nachricht. Anstatt in der URI werden in dem Entity-Body eines POST-Requests Informationen als Parameter-Wert-Paare übergeben. Der Entity-Body ist größer als die URI und ist nicht so leicht einsehbar wie die URI.[140]

Es gibt drei Arten, mit denen Session-Tracking betrieben werden kann: „URL-Rewriting", „versteckte Felder" und „HTTP-Cookies".

1. URL-Rewriting: Das URL-Rewriting übergibt bei jeder HTTP-Nachricht die Informationen, die die Session identifizieren über die URL. Dabei werden die Parameter-Wert-Paare hinten an die ursprüngliche URL angehängt. Dadurch wird z. B. aus der URL „http://www.firma.de" die URL „http://www.firma.de?user=admin&pw=1234".[141] Die Parameter-Wert-Paare werden durch das „?" von der ursprünglichen URL und durch das „&" untereinander getrennt.
2. Versteckte Felder: Diese Art des Session-Trackings übergibt Username und Passwort über den Entity-Body jeder HTTP-Nachricht. Die Web-Anwendung hält Username und Passwort in versteckten HTML-Formularfeldern vor und fügt bei jeder HTTP-Nachricht die Benutzer-Daten in den Entity-Body ein. Im HTML-Quellcode lässt sich der Inhalt dieser Felder jedoch sehr einfach auslesen.[142]
3. HTTP-Cookies: HTTP-Cookies sind kleine Textdokumente, die Parameter-Wert-Paare für die Übermittlung mit HTTP speichern. HTTP-Cookies werden im Header einer HTTP-Nachricht übergeben und sind deshalb nur schwer für Dritte auslesbar. HTTP-Cookies beinhalten neben den Benutzer-Daten auch Metadaten wie z. B. einen Auslauf-Zeitstempel des HTTP-Cookies und Domains für die ein HTTP-Cookie gilt.[143]

HTTP übermittelt Informationen unverschlüsselt über das Internet. Dadurch können Informationen wie Account- oder Transaktions-Daten von Dritten einfach abgegriffen werden. Um den Datenverkehr abzusichern, wird HTTPS (Hypertext Transfer Protocol Se-

140 Vgl. Balzert, Helmut: Lehrbuch der Software-Technik – Softwareentwicklung, a. a. O., S. 956, 959.
141 Vgl. Balzert, Helmut: Lehrbuch der Software-Technik – Softwareentwicklung, a. a. O., S. 959.
142 Vgl. Balzert, Helmut: Lehrbuch der Software-Technik – Softwareentwicklung, a. a. O., S. 959.
143 Vgl. Barth, Adam: HTTP State Management Mechanism, Internet Engineering Task Force (Hrsg.), April 2011, https://tools.ietf.org/pdf/rfc6265.pdf, abgerufen am 29.01.2021, S. 10.

cure) statt HTTP eingesetzt. HTTPS verschlüsselt den Datenverkehr mit dem TLS-Protokoll (Transport Layer Security).[144]

Das TLS-Protokoll führt eine Ende-zu-Ende Verschlüsselung des Datenverkehrs durch.[145] Eine Ende-zu-Ende Verschlüsselung nutzt „asymmetrische Verschlüsselungsverfahren", die eine Nachricht über den ganzen Weg von Sender zu Empfänger verschlüsselt. Erst lokal beim Empfänger wird die Nachricht entschlüsselt.[146]

Asymmetrische Verschlüsselungsverfahren nutzen ein „asymmetrisches Schlüsselpaar". Ein asymmetrisches Schlüsselpaar besteht aus einem öffentlichen und einem privaten Schlüssel. Der öffentliche Schlüssel wird bekannt gegeben und kann von jedem Kommunikationspartner eingesetzt werden, um Nachrichten an den Eigentümer des öffentlichen Schlüssels zu verschlüsseln. Der private Schlüssel ist nur dem User/Computer bekannt, dem das Schlüsselpaar gehört. Dieser User/Computer entschlüsselt mit dem privaten Schlüssel die Nachrichten, die mit seinem öffentlichen Schlüssel verschlüsselt wurden. Durch kryptographische Verfahren ist ein Schlüsselpaar eindeutig und ein Schlüssel lässt sich nicht aus dem anderen berechnen.[147]

B. OAuth und OpenID Connect

B.1 Geschichtlicher Hintergrund von OAuth und OIDC

Im November 2006 kam eine Gruppe von Entwicklern um Blaine Cook erstmals auf die Idee, die Authentifizierung für den Zugriff auf APIs auszulagern. Im April 2007 wurde eine Gruppe von Entwicklern bei Google gegründet, um einen Vorschlag für ein offenes Protokoll zu verfassen. Im Juli 2007 entwarf das Team von Google eine erste Spezifikation und

144 Vgl. Hansen, Hans; Mendling, Jan; Neumann, Gustaf: Wirtschaftsinformatik – Grundlagen und Anwendung, Walter de Gruyter GmbH (Hrsg.), Berlin/Boston, 2019, 12. Auflage, S. 401.

145 Vgl. Rescorla, Eric: The Transport Layer Security (TLS) Protocol Version 1.3, Internet Engineering Task Force (Hrsg.), August 2018, https://www.rfc-editor.org/rfc/pdfrfc/rfc8446.txt.pdf, abgerufen am 29.01.2021, S. 6.

146 Vgl. O. V.: Verschlüsselt kommunizieren im Internet, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/Empfehlungen/Verschluesselung/Verschluesselt kommunizieren/verschluesselt_kommunizieren_node.html;jsessionid=2FEEAF0B27614EF2A 50F0FB85DF5BF28.2_cid500, abgerufen am 28.12.2020.

147 Vgl. O. V.: Asymmetrische Verschlüsselung, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/Empfehlungen/Verschluesselung/Verschluesselt kommunizieren/Grundlagenwissen/AsymmetrischeVerschluesselung/asymmetrische_verschluesselung_node.htm, abgerufen am 28.12.2020.

die Gruppe wurde für jeden geöffnet, der daran interessiert war, einen Beitrag zu leisten. Am 3. Oktober 2007 wurde der endgültige Entwurf von OAuth Core 1.0 veröffentlicht.[148]

Im Okober 2012 wurde dann die zweite Version des OAuth-Frameworks durch die Internet Engineering Task Force (IETF) standardisiert.[149] OAuth 1.0 basierte auf zwei proprietären Protokollen: Flickr's Autorisierungs API und Google's AuthSub. Dadurch waren Implementierungen von anderen Unternehmen in der Funktionalität eingeschränkt. Je mehr OAuth 1.0 erweitert werden sollte, desto schwieriger und aufwändiger wurde eine Umsetzung. Nach langen Diskussionen mit großen Unternehmen wie z. B. Microsoft und Facebook wurde dann OAuth 2.0 geschaffen. OAuth 2.0 ist eine komplette Überarbeitung von OAuth 1.0.[150]

OAuth 2.0 ist noch immer der aktuelle Standard von OAuth. Doch in naher Zukunft soll die Version 2.1 von OAuth eingeführt werden. Mit OAuth 2.1 sollen veraltete und unsichere Funktionalitäten von OAuth 2.0 wegfallen und alle über die Jahre neu hinzugekommenen Funktionalitäten mit in den Standard übernommen werden. OAuth 2.1 soll außerdem die umfangreiche Sammlung an Standard-Dokumenten zusammenfassen und übersichtlicher machen.[151]

OpenID Connect ist die dritte Generation von OpenID-Technologie. Das erste OpenID-Protokoll wurde noch vor OAuth im Mai 2006 standardisiert und basiert dementsprechend auch nicht auf OAuth. Die Idee hinter OpenID 1 war schon damals eine dezentrale Authentifizierung durch Auth-Server und der Einsatz von Tokens, damit Passwörter nicht mehrfach übertragen werden müssen.[152] OpenID Version 1 wurde aber nur selten implementiert.[153]

148 Vgl. Hammer-Lahav, Eran: Introduction, oauth.net (Hrsg.), 05. September 2007, https://oauth.net/about/introduction/, abgerufen am 01.03.2021.

149 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021.

150 Vgl. Parecki, Aaron: Differences Between OAuth 1 and 2, in: OAuth 2.0 Simplified, oauth.com (Hrsg.), https://www.oauth.com/oauth2-servers/differences-between-oauth-1-2/, abgerufen am 01.03.2021.

151 Vgl. Parecki, Aaron (Hrsg.): It's Time for OAuth 2.1, 12. Dezember 2019, https://aaronparecki.com/2019/12/12/21/its-time-for-oauth-2-dot-1, abgerufen am 01.03.2021.

152 Vgl. Recordon, D.; Fitzpatrick, B.: OpenID Authentication 1.1, OpenID Foundation (Hrsg.), Mai 2006, https://openid.net/specs/openid-authentication-1_1.html, abgerufen am 01.03.2021, S. 1.

153 Vgl. OpenID Foundation (Hrsg.): OpenID Connect FAQ and Q&As, https://openid.net/connect/faq/, abgerufen am 01.03.2021.

Mit OpenID 2.0 wurde dann am 05. Dezember 2007[154] eine verbesserte, konzeptionell vollständig durchdachte Version herausgebracht. OpenID 2.0 war sicher und funktionierte gut. Doch OpenID 2.0 funktionierte nur mit Web-Client-Anwendungen und konnte nicht mit nativen Client-Anwendungen umgesetzt werden.[155]

Am 26 Februar 2014 wurde dann OpenID Connect veröffentlicht. OIDC ist das erste OpenID-Protokoll, das auf OAuth basiert. Dadurch konnten viele Probleme von OpenID 2.0 behoben werden.[156] OIDC ist bis heute der aktuelle Standard für Authentifizierung mit OpenID-Technologie.

B.2 Kommunikations-Objekte von OAuth und OIDC

OIDC und OAuth 2.0 setzen in ihren Protokollabläufen Kommunikations-Objekte ein. Kommunikations-Objekte enthalten autorisierungsrelevante Informationen, die in Form von Parameter-Wert-Paaren im Header, in der URI oder dem Entity-Body einer HTTP-Nachricht übermittelt werden. OAuth 2.0 und OIDC setzen die gleichen Kommunikations-Objekte ein.[157] Die Kommunikations-Objekte sind „Authorization-Codes", „Access-Tokens" und „Refresh-Tokens".[158]

Access-Tokens sind die wichtigsten Kommunikations-Objekte von OAuth 2.0 und OIDC. Denn Access-Tokens autorisieren eine Client-Anwendung zum Zugriff auf Web-Anwendungen.[159] Access-Tokens enthalten keine User-Daten.[160] OIDC spezifiziert den Einsatz von „ID-Tokens", die mit einem OAuth-2.0-Access-Token mitgeschickt werden. Diese ID-Tokens enthalten User-Daten und authentifizierungs- und autorisierungsrelevante Informationen (siehe unten).

154 Vgl. OpenID Foundation (Hrsg.): OpenID Authentication 2.0 - Final, 05. Dezember 2007, https://openid.net/specs/openid-authentication-2_0.html, abgerufen am 01.03.2021.

155 Vgl. OpenID Foundation (Hrsg.): OpenID Connect FAQ and Q&As, a. a. O., abgerufen am 01.03.2021.

156 Vgl. OpenID Foundation (Hrsg.): OpenID Connect FAQ and Q&As, a. a. O., abgerufen am 01.03.2021.

157 Vgl. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, a. a. O., abgerufen am 18.02.2021, S. 12. Vgl. auch Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 8.

158 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 8.

159 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 10.

160 Vgl. Krebs, Bruno: The OpenID Connect Handbook, Auth0 Inc. (Hrsg.), https://assets. Ctfassets.net/2ntc334xpx65/7pXoFWwEciDBmxEklweBKL/2621fe2b79c932fcddecb711bd6679fd/the-openid-connect-handbook-1_1.pdf, abgerufen am 25.01.2021, S. 16.

Ein User kann einen Access-Token nur mit einem „Authorization-Code“ beim Auth-Server anfordern (siehe Kapitel 3). Mit einem Authorization-Code darf nur genau ein Access-Token angefordert werden.[161] Ein Authorization-Code muss über ein „Authorization-Request“ (Schritt 3 in Abbildung 5) angefordert werden. Ein Authorization-Request ist eine HTTP-Nachricht von der Client-Anwendung an den Auth-Server. In Abbildung 15 ist ein solcher Authorization-Request dargestellt. Ein Authorization-Request mit OAuth 2.0 hat die folgenden Parameter:[162]

- scope (erforderlich): Dieser Parameter listet die Rechte auf, die mit einem Access-Token gewährt werden sollen.[163] Bei OIDC muss der Parameter „scope“ den Wert „openid“ haben, da sonst OIDC nicht eingesetzt wird.
- response type (erforderlich): Über diesen Parameter wird festgelegt, welcher OAuth-2.0-Protokollablauf eingesetzt werden soll. Der in der vorliegenden Arbeit betrachtete Authorization-Code-Grant hat den Wert „code“.
- client id (erforderlich): Dieser Parameter dient zur Identifizierung der Client-Anwendung, für die der Access-Token ausgestellt werden soll. Eine client_id ist für einen Auth-Server eindeutig.
- redircet uri (erforderlich): Dieser Parameter gibt an, an welche URI die Authentication-Response geschickt und wohin der User nach der Ausstellung des Access-Tokens umgeleitet werden soll.
- state (empfohlen): Dieser Parameter enthält einen zufällig generierten Wert, der unverändert in der Authorization-Response eingetragen wird. Mit diesem Attribut sollen „Cross-Site-Request-Forgery-Angriffe“[164] verhindert werden.

Ein Authorization-Request mit OIDC kann zusätzlich die folgenden Parameter haben:[165]

161 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 56.

162 Vgl. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, a. a. O., abgerufen am 18.02.2021, S. 13.

163 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 23.

164 Bei „Cross-Site-Request-Forgery“ (CSRF) manipulieren Angreifer HTTP-Nachrichten von authentifizierten Usern oder schieben authentifizierten Usern manipulierte HTTP-Nachrichten unter. Diese HTTP-Nachrichten sollen unbemerkt z. B. Username und User-Passwort oder Transaktionsdaten (z. B. bei Überweisungen) ändern. Siehe: S., Kirsten: Cross Site Request Forgery (CSRF), OWASP Foundation Inc. (Hrsg.), https://owasp. org/www-community/attacks/csrf#, abgerufen am 23.02.2021.

165 Vgl. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, a. a. O., abgerufen am 18.02.2021, S. 13-16.

- response_mode (optional): Über diesen Parameter kann festgelegt werden, in welchem Format die Authorization-Response zurückgegeben wird. Die Parameter-Wert-Paare der Authorization-Response werden in jedem Fall an eine URI angehängt. Dieser Parameter wird in der vorliegenden Arbeit nicht beachtet, da sein Wert nicht vom Standardwert abweicht und der Parameter deswegen nicht gesetzt werden muss.[166]
- display (optional): Dieser Parameter gibt an, wie Schritt 3 (siehe Abbildung 5) der Abläufe von OAuth 2.0 und OIDC technisch dargestellt wird. In der vorliegenden Arbeit hat dieser Parameter den Wert „popup". Dadurch werden die Abläufe aus Schritt 3 (siehe Abbildung 5) in einem Pop-Up Fenster dargestellt.
- prompt (optional): Dieser Parameter gibt an, was ein User tun muss, damit eine Session nicht ausläuft. In Form einer Liste wird angegeben, welche Aktionen benötigt werden. Es kann entweder nichts, eine Authentifizierung, eine Autorisierung und/oder eine Auswahl eines User-Accounts gefordert werden. In der vorliegenden Arbeit wird für diesen Parameter der Wert „login" angenommen, der eine Authentifizierung fordert.
- max_age (optional): Dieser Parameter gibt an, wie lange eine Authentifizierung eines Users aktiv ist. Ist diese Zeit verstrichen, muss der User sich neu authentifizieren. Wenn dieser Parameter gesetzt ist, muss ein ID-Token das Attribut „auth_time" enthalten.
- ui_locals (optional): Dieser Parameter enthält eine Liste der Kürzel der bevorzugten Sprachen der Web-Anwendung.
- id_token_hint (optional): Dieser Parameter enthält den ID-Token einer vorangegangenen, abgelaufenen Session der gleichen Client-Anwendung und des gleichen Users. Dieser Parameter soll eingesetzt werden, wenn der Wert des „prompt" Attributs „none" ist. Da in der vorliegenden Arbeit der Parameter „prompt" den Wert „login" hat, wird dieser Parameter nicht weiter betrachtet.
- login_hint (optional): Dieser Parameter übergibt dem Auth-Server vorab die User-ID des Users , der sich anmelden will. Dieser Parameter wird in der vorliegenden Arbeit nicht beachtet.
- acr_values (optional): Dieser Parameter gibt in einer Liste an, welche Werte das „acr" Attribut eines ID-Tokens haben darf.

166 Vgl. De Medeiros, Breno et al.: OAuth 2.0 Multiple Response Type Encoding Practices, a. a. O., abgerufen am 22.02.2021, S. 3.

```
HTTP/1.1 302 Found
Location: https://firma.de/auth?
  response_type=code
  &scope=openid
  &client_id=Djnf4UG2
  &state=hawJweg
  &redirect_uri=https%3A%2F%2Fservice.firma.de%2Flogin_cb
  &display=popup
  &prompt=login
  &max_age=3600
  &ui_locals=de
  &acr_values=urn:firma:de:level-1%20urn:firma:de:level-2
```

Abb. 15: Ein OIDC-Authorization-Request (eigene Abbildung)

Auf einen Authorization-Request folgt eine „Authorization-Response“ (Schritt 4 in Abbildung 5). Diese Authorization-Response ist eine HTTP-Nachricht vom Auth-Server an die Client-Anwendung, die den Authorization-Code übergibt. In Abbildung 16 ist eine solche Authorization-Response dargestellt. Eine Authorization-Response hat bei OAuth 2.0 und OIDC die folgenden Parameter-Wert-Paare:[167]

- code (erforderlich): Dieser Parameter beinhaltet einen vom Auth-Server generierten Wert, der die Autorisierung einer Client-Anwendung durch einen authentifizierten User nachweist. Der Wert dieses Parameters ist das Kommunikations-Objekt „Authorization-Code“.
- state (erforderlich): Dieser Parameter enthält einen zufällig generierten Wert, der bei der Anforderung eines Authorization-Codes an den Auth-Server übergeben und unverändert hier eingetragen wird.

```
https://firma.de/auth?code=HlejfIHWIFAWifhAwu&state=hawJweg
```

Abb. 16: Eine Authorization-Response nach OAuth 2.0 und OIDC (eigene Abb.)

167 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 26f.

Access-Tokens autorisieren eine Client-Anwendung zum Zugriff auf Web-Anwendungen.[168] Um Access-Tokens einzusetzen, werden diese als HTTP-Cookie im Authorization-Header[169] einer HTTP-Nachricht an den Web-Server mitgeschickt (siehe Kapitel 3).[170] Access-Tokens müssen mit einem „Token-Request" angefordert werden (Schritt 5 in Abbildung 5). Ein „Token-Request" ist eine HTTP-Nachricht, die von der Client-Anwendung an den Auth-Server geschickt wird, um einen Authorization-Code in einen Access-Token umzutauschen. Abbildung 17 zeigt einen solchen Token-Request. Die folgenden Parameter hat ein Token-Request bei OAuth 2.0 und OIDC:[171]

```
POST /token HTTP/1.1
Host: auth.firma.de
Content-Type: application/x-www-form-urlencoded
Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW

 grant_type=authorization_code
 &code=HlejfIHWIFAWifhAwu
 &redirect_uri=https%3A%2F%2Fservice.firma.de%2Flogin_cb
 &client_id= Djnf4UG2
```

Abb. 17: Ein Token-Request nach OAuth 2.0 und OIDC (eigene Abbildung)

- grant_type (erforderlich): Über diesen Parameter wird festgelegt, welcher OAuth-2.0-Ablauf[172] eingesetzt werden soll. Der in der vorliegenden Arbeit betrachtete Authorization-Code-Grant hat den Wert „authorization_code".

168 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 10.

169 In Anhang A befindet sich eine kurze Einführung in die Themen „HTTP" und „HTTPS". Dort gibt es auch einen Abschnitt zu „HTTP-Cookies" und eine Erläuterung zum „Header" einer HTTP-Nachricht.

170 Vgl. Grof, Thomas: OAuth 2.0: Anbietervergleich für selbst gehostete Autorisierungsserver, a. a. O., abgerufen am 25.01.2021, S. 39.

171 Vgl. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, a. a. O., abgerufen am 18.02.2021, S. 19f. Vgl. auch Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 29.

172 OAuth 2.0 hat vier Protokollabläufe, die vorgeben, wann wer welche Daten übermittelt oder verarbeitet. OAuth 2.0 spezifiziert vier Authorization-Grants: „Authorization-Code-Grant", „Implicit-Grant", „Resource-Owner-Password-Credentials-Grant", „Client-Credentials-Grant". Für die vorliegende Arbeit ist nur der Authorization-Code-Grant relevant, da die anderen Protokollabläufe Spezialfälle abdecken. Siehe: Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 8f.

- code (erforderlich): Dieser Parameter ist für den in den Authorization-Code-Grant verpflichtend zu setzen. Es beinhaltet den „Authorization-Code“.
- redirect_uri (erforderlich): Dieser Paramter enthält die gleiche URI wie der Parameter „redirect_uri“ des oben beschriebenen Authorization-Requests.
- client_id (erforderlich): Dieser Parameter dient zur Identifizierung des Clients, für den der Access-Token ausgestellt werden soll.

Auf einen Token-Request folgt eine „Token-Response“. Eine Token-Response ist eine HTTP-Nachricht vom Auth-Server an die Client-Anwendung (Schritt 6 in Abbildung 5). Diese Token-Response übergibt der Client-Anwendung den Access-Token und weitere Informationen in Form von Parameter-Wert-Paaren. In Abbildung 18 ist eine OAuth-2.0-Token-Response dargestellt. Die folgenden Parameter-Wert-Paare enthält eine OAuth-2.0-Token-Response:[173]

- access_token (erforderlich): Dieser Parameter enthält das Kommunikations-Objekt „Access-Token“. Dieser Token ist ein vom Auth-Server zufällig generierter Wert. Stimmt ein übergebener Access-Token mit dem beim Auth-Server gespeicherten Access-Token überein, stammt eine HTTP-Nachricht von einem autorisierten Absender.
- token_type (erforderlich): Dieser Parameter definiert, wie ein Access-Token eingesetzt werden und welche Parameter ein Token-Request zusätzlich enthalten muss.[174]
- expires_in (empfohlen): Dieser Parameter gibt an, wie lange ein Access-Token gültig ist. Wenn dieser Parameter nicht gesetzt ist, wird ein Standard-Wert eingefügt.
- refresh_token (optional): Der in diesem Parameter enthaltene Refresh-Token wird dafür genutzt, einen neuen Access-Token vom Auth-Server abzufragen, ohne dass der User sich erneut authentifizieren muss. Der Wert dieses Parameters ist das Kommunikations-Objekt „Refresh-Token“.
- scope (erforderlich): Dieser Parameter listet die Rechte auf, die mit dem Access-Token gewährt werden.[175] Je nach Scope-Wert kann eine Token-Response weitere Attribute enthalten (z. B. „scope=openid“, siehe Abbildung 19).

173 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 43f.
174 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 49.
175 Vgl. Hardt, Dick: The OAuth 2.0 Authorization Framework, a. a. O., abgerufen am 25.01.2021, S. 23.

```
{
      "access_token" : "RpNGfti3y772iWVW7GpK",
      "token_type" : "bearer",
      "expires_in" : 3600,
      "refresh_token" : "MqrFeEc7osT7Bvf9CgqV",
      "scope" : "read"
}
```

Abb. 18: Ein OAuth-2.0-Token-Request (eigene Abbildung)

Eine Token-Response bei OIDC muss den Wert „openid“ für den Parameter „scope“ haben.[176] Zudem enthält die Token-Response ein „ID-Token“. ID-Tokens bestehen aus Attributen und werden in Form eines verschlüsselten[177] „JSON Web Tokens“ (JWT) als Parameter („id_token“) einer Token-Response angefügt (siehe Abbildung 19).[178] Abbildung 19 zeigt wie die OAuth-2.0-Token-Response aus Abbildung 18 zu einer OIDC-Token-Response erweitert wurde. Die folgenden Attribute kann ein ID-Token enthalten:[179]

- iss (erforderlich): Dieses Attribut enthält die URL des Auth-Servers. So kann sichergestellt werden, dass ein ID-Token vom richtigen Auth-Server stammt.
- sub (erforderlich): Dieses Attribut enthält die für einen Auth-Server eindeutige ID des Users, dem der Token gehört.
- aud (erforderlich): Dieses Attribut enthält eine Liste von für einen Auth-Server eindeutigen Client-IDs. ID-Tokens können nur bei Web-Anwendung eingesetzt werden, die in diesem Attribut mit ihrer Client-ID eingetragen sind.
- exp (erforderlich): Dieses Attribut gibt an, wann ein ID-Token ausläuft.
- iat (erforderlich): Dieses Attribut gibt an, wann der ID-Token erzeugt wurde.
- auth_time (erforderlich): Dieses Attribut gibt an, wann die Authentifizierung des Users stattgefunden hat. Das Attribut ist erforderlich, wenn in der Anfrage des ID-Tokens das

176 Vgl. Grof, Thomas: OAuth 2.0: Anbietervergleich für selbst gehostete Autorisierungsserver, a. a. O., abgerufen am 25.01.2021, S. 41f.

177 Vgl. Jones, Michael; Bradley, J.; Sakimura, N.: JSON Web Token (JWT) – draft-ietf-oauth-json-web-token-32, Internet Engineering Task Force (Hrsg.), 09. Dezember 2014, https:// tools.ietf.org/pdf/draft-ietf-oauth-json-web-token-32.pdf, abgerufen am 25.01.2021, S. 1.

178 Vgl. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, a. a. O., abgerufen am 18.02.2021, S. 21.

179 Vgl. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, OpenID Foundation (Hrsg.), https://openid.net/specs/openid-connect-core-1_0.html, abgerufen am 18.02.2021, S. 9f.

„max_age“-Attribut (siehe oben) gesetzt ist und dadurch die Authentifizierung des Users nach einer bestimmten Zeit ausläuft. In der vorliegenden Arbeit wird dieses Attribut als erforderlich angesehen.

- nonce (erforderlich): Dieses Attribut beinhaltet einen Wert, der ein ID-Token mit einer Session[180] verbindet. Der Wert wird bei der Anforderung eines ID-Tokens angegeben und ohne Veränderung im ID-Token eingetragen. Dieser Wert unterscheidet sich bei jedem ID-Token. Das verringert das Risiko von Replay-Angriffen.
- acr (optional): Dieses Attribut gibt in Form eines abstrakten Werts Aufschluss über die Qualität der User-Authentifizierung.
- amr (optional): Dieses Attribut gibt an, welche Maßnahmen für eine User-Authentifizierung eingesetzt wurden.
- azp (optional): Dieses Attribut beinhaltet die ID des Clients, für den der Token ausgestellt wurde. Dieses Attribut ist nur für Spezialfälle nötig und wird in der vorliegenden Arbeit nicht beachtet.

Abb. 19: Eine OIDC-Token-Response (eigene Abbildung)

180 In Anhang A befindet sich eine kurze Einführung in die Themen „HTTP“ und „HTTPS“. Dort wird auch erläutert, was eine „Session“ und was „Session-Tracking“ ist.

Literaturverzeichnis

1. Adams, Carlisle: Replay Attack, in: Encyclopedia of cryptography and security, Henk C. A. van Tilborg – Eindhoven University of Technology (Hrsg.), abgerufen am 29.01.2021, S. 519.
2. Apache Software Foundation (Hrsg.): Apache License Version 2.0, Januar 2004, https://www.apache.org/licenses/LICENSE-2.0.txt, abgerufen am 01.03.2021.
3. Apple Inc. (Hrsg.): Was ist "Mit Apple anmelden"?, https://support.apple.com/de-de/HT210318, abgerufen am 29.12.2020.
4. Balzert, Helmut: Lehrbuch der Softwaretechnik: Entwurf, Implementierung, Installation und Betrieb, Spektrum Akademischer Verlag GmbH (Hrsg.), Heidelberg, 2011.
5. Balzert, Helmut: Lehrbuch der Software-Technik – Softwareentwicklung, Spektrum Akademischer Verlag GmbH (Hrsg.), Heidelberg; Berlin, 2001, 2. Auflage.
6. Barth, Adam: HTTP State Management Mechanism, Internet Engineering Task Force (Hrsg.), April 2011, https://tools.ietf.org/pdf/rfc6265.pdf, abgerufen am 29.01.2021.
7. Barth, Michael et al.: Spionage, Sabotage und Datendiebstahl – Wirtschaftsschutz in der vernetzten Welt – Studienbericht 2020, Bitkom e.V. (Hrsg.), https://www.bitkom.org/sites/default/files/2020-02/200211_bitkom_studie_wirtschaftsschutz_2020_final.pdf, abgerufen am 02.02.2021.
8. Berners-Lee, Tim et al.: Uniform Resource Identifier (URI): Generic Syntax, Internet Engineering Task Force (Hrsg.), Januar 2005, https://tools.ietf.org/pdf/rfc3986.pdf, abgerufen am 26.01.2021.
9. Bonneau, Joseph et al.: The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes, University of Cambridge – Computer Laboratory (Hrsg.), März 2012, https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-817.pdf, abgerufen am 25.01.2021.

10. Breitkopf, Tom-Lukas: FIDO2 als TLS-1.3-Erweiterung, Humboldt-Universität zu Berlin – Mathematisch-Naturwissenschaftliche Fakultät – Institut für Informatik (Hrsg.), https://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2020-04/SAR-PR-2020-04_.pdf, abgerufen am 30.12.2020.

11. De Medeiros, Breno et al.: OAuth 2.0 Multiple Response Type Encoding Practices, OpenID Foundation (Hrsg.), 25. Februar 2014, https://openid.net/specs/oauth-v2-multiple-response-types-1_0.html#id_token, abgerufen am 22.02.2021.

12. Digitales Wörterbuch der deutschen Sprache (Hrsg.): Methode, https://www.dwds.de/wb/Methode, abgerufen am 15.02.2021.

13. Eikenberg, Ronald: Login per Finger – Goldengate Security Keys für FIDO2, in: C't 20/11, Heise Medien GmbH & Co. KG (Hrsg.), S. 85.

14. Eikenberg, Ronald: Schlüssel zum Glück – Was schon heute mit dem Passwort-Killer FIDO2 geht, in: C't 19/18, Heise Medien GmbH & Co. KG (Hrsg.), S. 20-24.

15. Feitian Technologies Co., Ltd. (Hrsg.): FIDO-Security-Key, https://www.ftsafe.com/store/product-category/fido-security-key/#productCategory, abgerufen am 07.03.2021.

16. Fettke, Peter: Client-Server-Architektur, in: Enzyklopädie der Wirtschaftsinformatik, Universität Potsdam – Lehrstuhl für Wirtschaftsinformatik – Norbert Gronau (Hrsg.), 23. September 2016, https://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/is-management/Systementwicklung/Softwarearchitektur/Architekturparadigmen/Client-Server-Architektur/index.html, abgerufen am 26.01.2021.

17. FIDO Alliance (Hrsg.): History of FIDO Alliance, https://fidoalliance.org/overview/history/, abgerufen am 01.03.2021.

18. Fink, Andreas: Verteiltes IT-System, in: Enzyklopädie der Wirtschaftsinformatik, Universität Potsdam – Lehrstuhl für Wirtschaftsinformatik – Norbert Gronau (Hrsg.), 31. Oktober 2012, https://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/is-management/Systementwicklung/Softwarearchitektur/Middleware/index.html, abgerufen am 26.01.2021.

19. Fischer, Manuel et al.: Entwicklung erfolgreicher Web-Anwendungen – Leitfaden Webentwicklung 2015, Bitkom e.V. (Hrsg.), 2015, https://www.bitkom.org/sites/default/files/file/import/LF-Web-Anwendungen-150910-1.pdf, abgerufen am 23.12.2020.

20. Gerboth, Thomas: Das Magische Dreieck, in: Controlling, Heft 7, Verlag C.H.Beck oHG (Hrsg.), Juli 2002, S. 417.

21. Google LLC (Hrsg.): Mit Ihrem Google-Konto können Sie sich auch bei anderen Apps oder Diensten anmelden, https://support.google.com/accounts/answer/112802?co=GENIE.Platform%3DDesktop&hl=de&oco=0, abgerufen am 29.12.2020.

22. Grassi, Paul; Garcia, Michael; Fenton, James: NIST Special Publication 800-63-3 – Digital Identity Guidelines, U.S. Department of Commerce – National Institute of Standards and Technology (Hrsg.), Juni 2017, https://doi.org/10.6028/NIST.SP.800-63-3, abgerufen am 02.02.2021.

23. Grof, Thomas: OAuth 2.0: Anbietervergleich für selbst gehostete Autorisierungsserver, Johannes Kepler Universität Linz – Institut für Wirtschaftsinformatik – Software Engineering (Hrsg.), Januar 2020, https://epub.jku.at/obvulihs/download/pdf/4777592?originalFilename=true, abgerufen am 25.01.2021.

24. Günther, Oliver; Fabian, Benjamin; Ziekow, Holger: Biometrie, in: Enzyklopädie der Wirtschaftsinformatik, Universität Potsdam – Lehrstuhl für Wirtschaftsinformatik – Norbert Gronau (Hrsg.), 29. September 2020, https://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/technologien-methoden/Informatik--Grundlagen/automatische-identifikation/biometrie/index.html?searchterm=false+reject, abgerufen am 01.02.2020.

25. Hammer-Lahav, Eran: Introduction, oauth.net (Hrsg.), 05. September 2007, https://oauth.net/ about/introduction/, abgerufen am 01.03.2021.

26. Hansen, Hans; Mendling, Jan; Neumann, Gustaf: Wirtschaftsinformatik – Grundlagen und Anwendung, Walter de Gruyter GmbH (Hrsg.), Berlin/Boston, 2019, 12. Auflage.

27. Hardt, Dick: The OAuth 2.0 Authorization Framework, Internet Engineering Task Force (Hrsg.), Oktober 2012, https://tools.ietf.org/pdf/rfc6749.pdf, abgerufen am 25.01.2021.

28. Hildebrandt, Knut: Konzeption von Authentifizierungsmechanismen für anonyme Dienste und Realisierung eines ausgewählten Verfahrens im Anonymisierungsnetzwerk Tor, Otto-Friedrich-Universität Bamberg – Fakultät Wirtschaftsinformatik und Angewandte Informatik – Lehrstuhl für Praktische Informatik (Hrsg.), 18. Januar 2007, https://www.uni-bamberg.de/fileadmin/uni/fakultaeten/wiai_lehrstuehle/praktische_informatik/Dateien/Forschung/Tor/hildebrandt-authentication.pdf, abgerufen am 05.03.2021.

29. Holland, Martin: NSA-Überwachungsskandal: Von NSA, GCHQ, BND, PRISM, Tempora, XKeyScore und dem Supergrundrecht – was bisher geschah, Heise Medien GmbH & Co. KG (Hrsg.), 19. September 2013, https://www.heise.de/newsticker/meldung/NSA-Ueberwachungsskandal-Von-NSA-GCHQ-BND-PRISM-Tempora-XKeyScore-und-dem-Supergrundrecht-was-bisher-geschah-1958399.html, abgerufen am 30.12.2020.

30. Institut für Demoskopie Allensbach (Hrsg.): Auszug – Stationäre und mobile Internetnutzung, in: Allensbacher Computer und Technik-Analyse, 2016, https://www.ifd-allensbach.de/fileadmin/ACTA/ACTA2016/Codebuchausschnitte_ACTA_2016/ACTA2016_Stationaere_Mobile-Internetnutzung.pdf, abgerufen am 02.02.2021, S. 119f.

31. Jones, Michael: Second Implementer's Draft of OpenID Connect User Questioning API Specification Approved, OpenID Foundation (Hrsg.), 14. Januar 2021, https://openid.net/2021/01/14/second-implementers-draft-of-openid-connect-user-questioning-api-specification-approved/, abgerufen am 15.02.2021.

32. Jones, Michael; Bradley, J.; Sakimura, N.: JSON Web Token (JWT) – draft-ietf-oauth-json-web-token-32, Internet Engineering Task Force (Hrsg.), 09. Dezember 2014, https://tools.ietf.org/pdf/draft-ietf-oauth-json-web-token-32.pdf, abgerufen am 25.01.2021.

33. Karcher, Andreas: Middleware, in: Enzyklopädie der Wirtschaftsinformatik, Universität Potsdam – Lehrstuhl für Wirtschaftsinformatik – Norbert Gronau (Hrsg.), 29. November 2016, https://www. enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/is-management/Systementwicklung/Softwarearchitektur/Middleware/index.html, abgerufen am 26.01.2021.

34. Keycloak.org (Hrsg.): Server Administration Guide, https://www.keycloak.org/docs/latest/server_admin/, abgerufen am 30.12.2020.

35. KirstenS: Cross Site Request Forgery (CSRF), OWASP Foundation Inc. (Hrsg.), https://owasp.org/www-community/attacks/csrf#, abgerufen am 23.02.2021.

36. Klassen, David: Eine Identität für alles mit Keycloak, Heise Medien GmbH & Co. KG (Hrsg.), 19. September 2017, https://m.heise.de/developer/artikel/Eine-Identitaet-fuer-alles-mit-Keycloak-38 34525.html?seite=all, abgerufen am 01.03.2021.

37. Koserwal, Abhishek: Keycloak: Core concepts of open source identity and access management, Red Hat Inc. (Hrsg.), 11. Dezember 2019, https://developers.redhat.com/blog/2019/12/11/keycloak-core-concepts-of-open-source-identity-and-access-management/, abgerufen am 01.03.2021.

38. Krebs, Bruno: The OpenID Connect Handbook, Auth0 Inc. (Hrsg.), https://assets. Ctfassets.net/2ntc334xpx65/7pXoFWwEciDBmxEklweBKL/2621fe2b79c932fcddecb711bd6679fd/the-openid-connect-handbook-1_1.pdf, abgerufen am 25.01.2021.

39. Krebs, Rouven; Momm, Christof; Kounev, Samuel: Architectural Concerns in Multi-Tenant SaaS Applications, Julius-Maximilians-Universität Würzburg – Fakultät für Informatik, https://se2.informatik.uni-wuerzburg.de/pa/uploads/papers/paper-371.pdf, abgerufen am 02.03.2021.

40. Kruse, Malte: Biometriebasierte Authentifizierung mit WebAuthn, Humboldt-Universität zu Berlin – Mathematisch-Naturwissenschaftliche Fakultät – Institut für Informatik (Hrsg.), https://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2020-02/SAR-PR-2020-02_.pdf, abgerufen am 30.12.2020.

41. Lapscheck, Niklas; Schick, Lukas; Schwickert, Axel: FIDO2 – Grundlagen, Prinzipien und praktische Anwendung, in: Arbeitspapiere WI, Nr. 2/2021, Hrsg.: Professur BWL – Wirtschaftsinformatik, Justus-Liebig-Universität Gießen 2021.

42. Lodderstedt, Thorsten: OpenID Connect: Login mit OAuth Teil 1 – Grundlagen, Heise Medien GmbH & Co. KG (Hrsg.), 10. Juni 2014, https://www.heise.de/developer/artikel/OpenID-Connect-Login-mit-OAuth-Teil-1-Grundlagen-2218446.html?view=print, abgerufen am 25.01.2021.

43. Lodderstedt, Thorsten; Hiller, Jochen: Flexible und sichere Internetdienste mit OAuth 2.0, Heise Medien GmbH & Co. KG (Hrsg.), 27. Dezember 2013, https://www.heise.de/developer/artikel/Flexible-und-sichere-Internetdienste-mit-OAuth-2-0-2068404.html?view=print, abgerufen am 25.01.2021.

44. Lyastani, Sanam et al.: Is FIDO2 the Kingslayer of User Authentication? A Comparative Usability Study of FIDO2 Passwordless Authentication, CISPA Helmholtz Center for Information Security, Saarbrücken, https://publications.cispa.saarland/3146/1/oakland20.pdf, abgerufen am 30.12.2020.

45. Mell, Peter; Grance, Tim: The NIST Definition of Cloud Computing, National Institute of Standards and Technology (Hrsg.), September 2011, https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf, abgerufen am 05.02.2021.

46. OpenID Foundation (Hrsg.): Certified OpenID Connect Implementations, https://openid.net/developers/certified/, abgerufen am 15.02.2021.

47. OpenID Foundation (Hrsg.): OpenID Authentication 2.0 - Final, 05. Dezember 2007, https://openid.net/specs/openid-authentication-2_0.html, abgerufen am 01.03.2021.

48. OpenID Foundation (Hrsg.): OpenID Connect FAQ and Q&As, https://openid.net/connect/faq/, abgerufen am 01.03.2021.

49. O. V.: Asymmetrische Verschlüsselung, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/Empfehlungen/Verschluesselung/Verschluesseltkommunizieren/Grundlagenwissen/AsymmetrischeVerschluesselung/asymmetrische_verschluesselung_node.htm, abgerufen am 28.12.2020.

50. O. V.: Digitale Signatur, in: Elektronik-Kompendium.de, Patrick Schnabel (Hrsg.), https://www.elektronik-kompendium.de/sites/net/1910131.htm, abgerufen am 05.03.2021.

51. O. V.: Phishing E-Mails – Passwortdiebstahl durch Phishing, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/Risiken/SpamPhishingCo/Phishing/phishing_node.html, abgerufen am 29.12.2020.

52. O. V.: Single Sign-On, Auth0.com (Hrsg.), https://auth0.com/docs/sso, abgerufen am 25.01.2021.

53. O. V.: Social Engineering – der Mensch als Schwachstelle, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi.bund.de/DE/Themen/Verbraucherinnen-und-Verbraucher/Cyber-Sicherheitslage/Methoden-der-Cyber-Kriminalitaet/Social-Engineering/social-engineering_node.html, abgerufen am 02.02.2021.

54. O. V.: Trojaner, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi.bund.de/DE/Themen/Verbraucherinnen-und-Verbraucher/Cyber-Sicherheitslage/Methoden-der-Cyber-Kriminalitaet/Schadprogramme/Trojaner/trojaner_node.html, abgerufen am 02.02.2021.

55. O. V.: Verschlüsselt kommunizieren im Internet, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/Empfehlungen/Verschluesselung/Verschluesseltkommunizieren/verschluesselt_kommunizieren_node.html;jsessionid=2FEEAF0B27614EF2A50F0FB85DF5BF28.2_cid50, abgerufen am 28.12.2020.

56. O. V.: Was beim Einsatz von Verschlüsselung zu beachten ist, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/Empfehlungen/Verschluesselung/Datenverschluesselung/Grundlagen/Wissenswertes/wissenswertes_node.html, abgerufen am 29.12.2020.

57. O. V.: Zwei-Faktor-Authentisierung für höhere Sicherheit, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/DigitaleGesellschaft/OnlineBanking/Zwei_Faktor_Authentisierung/Zwei-Faktor-Authentisierung_node.html, abgerufen am 29.12.2020.

58. Parecki, Aaron: Differences Between OAuth 1 and 2, in: OAuth 2.0 Simplified, oauth.com (Hrsg.), https://www.oauth.com/oauth2-servers/differences-between-oauth-1-2/, abgerufen am 01.03.2021.

59. Parecki, Aaron (Hrsg.): It's Time for OAuth 2.1, 12. Dezember 2019, https://aaronparecki.com/2019/12/12/21/its-time-for-oauth-2-dot-1, abgerufen am 01.03.2021.

60. Peyrott, Sebastian: Was ist Single Sign-on Authentication, und wie funktioniert sie?, auth0.com (Hrsg.), 05. März 2019, https://auth0.com/blog/de-what-is-and-how-does-single-sign-on-work/, abgerufen am 28.12.2020.

61. Recordon, D.; Fitzpatrick, B.: OpenID Authentication 1.1, OpenID Foundation (Hrsg.), Mai 2006, https://openid.net/specs/openid-authentication-1_1.html, abgerufen am 01.03.2021.

62. Reinhardt, Martin: IT-Systemzugriffe verwalten: Identity und Access Management mit Keycloak, in: iX 12/20, Heise Medien GmbH & Co. KG (Hrsg.).

63. Rescorla, Eric: The Transport Layer Security (TLS) Protocol Version 1.3, Internet Engineering Task Force (Hrsg.), August 2018, https://www.rfc-editor.org/rfc/pdfrfc/ rfc8446.txt.pdf, abgerufen am 29.01.2021.

64. Ries, Uli: Sie wurden gehackt! – Datenklau bei Yahoo, Dropbox & Co. betrifft Milliarden, in: C't 23/16, Heise Medien GmbH & Co. KG (Hrsg.), S. 78-84.

65. Saeed, Hasnat: Setup Keycloak Server on Ubuntu 18.04, Medium.com (Hrsg.), 31. Juli 2019, https://medium.com/@hasnat.saeed/setup-keycloak-server-on-ubuntu-18-04-ed8c7c79a2d9, abgerufen am 08.03.2021.

66. Sakimura, Nat et al.: OpenID Connect Core 1.0 incorporating errata set 1, OpenID Foundation (Hrsg.), https://openid.net/specs/openid-connect-core-1_0.html, abgerufen am 18.02.2021.

67. Schmidt, Jürgen: Verschlossen, nicht verrammelt – So funktioniert der passwortlose Login mit FIDO2, in: C't 19/18, Heise Medien GmbH & Co. KG (Hrsg.), S. 30ff.

68. Schmidt, Jürgen; Eikenberg, Ronald: Passwort-Nachfolger FIDO2, in: C't 19/22, Heise Medien GmbH & Co. KG (Hrsg.), S. 168ff.

69. Shikiar, Andrew: FIDO2: A Web without passwords – W3C track at the web conference, fido alliance (Hrsg.), 15. Mai 2019, https://www.w3.org/2019/05/15-w3c-track/assets/andrew-fido.PDF, abgerufen am 30.12.2020.

70. Steppan, Bernhard: Cloud statt on Premises: Java-Altanwendungen in AWS migrieren, in: iX 3/20, Heise Medien GmbH & Co. KG (Hrsg.), 20. Februar 2020, https://www.heise.de/ratgeber/Cloud-statt-on-Premises-Java-Altanwendungen-in-AWS-migrieren-4661891.html?view=print, abgerufen am 18.02.2021, S. 136.

71. Thiel, Barbara (Landesbeauftragte für den Datenschutz Niedersachsen) (Hrsg.): Handlungsempfehlung sichere Authentifizierung, 05. Februar 2020, https://lfd.niedersachsen.de/download/61775, abgerufen am 28.12.2020.

72. Wagner, Maik: Proseminar: "Electronic Commerce und Digitale Unterschriften" – Digitale Signaturen & Hashfunktionen, Technische Universität Chemnitz – Fakultät für Informatik – Professur Theoretische Informatik und Informationssicherheit (Hrsg.), https://www.tu-chemnitz.de/informatik/ThIS/downloads/courses/ss04/ecommerce/digsig.pdf, abgerufen am 03.02.2021.

73. Wintermeyer, Stefan: Nutzerpasswörter im Web intelligenter sichern, in: Linux-Magazin 08/19, https://www.linux-magazin.de/ausgaben/2019/08/passwoerter/, abgerufen am 03.03.2021.

74. Yubico.com (Hrsg.): FIDO2 passwordless authentication, https://www.yubico.com/authentication-standards/fido2/, abgerufen am 03.03.2021.

75. Yubico.com (Hrsg.): Yubikey 5 Series – For professionals, https://www.yubico.com/de/store/#for-professionals, abgerufen am 07.03.2021.

Weitere Quellen (offene Liste):

1. Arns, Tobias: Apps & Mobile Services – Tipps für Unternehmen, Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Hrsg.),2012, https://www.bitkom.org/sites/default/files/file/import/121214-Leitfaden-Apps-und-Mobile.pdf, abgerufen am 27.01.2021, S. 15.
2. Auth0.com (Hrsg.): How Siemens Centralized Their Login Experience With Auth0, https://auth0.com/case-studies/siemens/, abgerufen am 30.12.2020.
3. Barbir, Abbie; Ulrich, Amy: Part II: FIDO 2 Overview and use cases, FIGI Security Clinic (Hrsg.), 4-5 Dezember 2019, https://www.itu.int/en/ITU-T/Workshops-and-Seminars/201912/Documents/Amy%20Ulrich.pdf, abgerufen am 25.01.2021.
4. De Paula, Carlos: Adding authentication to your Kubernetes Web applications with Keycloak, Red Hat Inc. (Hrsg.), 02. April 2020, https://www.openshift.com/blog/adding-authentication-to-your-kubernetes-web-applications-with-keycloak, abgerufen am 30.12.2020.
5. Dierstein, Rüdiger: IT-Sicherheit und ihre Besonderheiten – Duale Sicherheit –, Technische Universität München – Fakultät für Informatik – Lehrstuhl für Datenbanksysteme, Oktober 2003, https://db.in.tum.de/index.shtml?lang=de, abgerufen am 25.01.2021, S. 8.
6. Farke, Florian et al.: “You still use the password after all” – Exploring FIDO2 Security Keys in a Small Company, Ruhr-Universität Bochum (Hrsg.), https://www.mobsec.ruhr-uni-bochum.de/media/mobsec/veroeffentlichungen/2020/06/15/fido2key.pdf, abgerufen am 30.12.2020.
7. FIDO Alliance (Hrsg.): How FIDO Works, https://fidoalliance.org/how-fido-works/, abgerufen am 25.01.2021.
8. FIDO Alliance (Hrsg.): FIDO2: WebAuthn & CTAP, https://fidoalliance.org/fido2/, abgerufen am 25.01.2021.
9. Horsch, Moritz; Tuengerthal, Max; Wich, Tobias: SAML Privacy-Enhancing Profile, Technische Universität Darmstadt (Hrsg.), https://dl.gi.de/bitstream/handle/20.500.12116/2629/11.pdf, abgerufen am 30.12.2020.

10. Hühnlein, Detlef; Korte, Ulrike: Grundlagen der elektronischen Signatur Recht Technik Anwendung, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), 25. März 2006, https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/ElekSignatur/esig_pdf.pdf;jsessionid=CCFEF65270910328C8D53EAA41E05AB3.internet472?__blob=publicationFile&v=1, abgerufen am 05.03.2021.

11. Keycloak.ch (Hrsg.): Tutorial 3 - Configuring WebAuthn, https://keycloak.ch/keycloak-tutorials/tutorial-webauthn/, abgerufen am 30.12.2020.

12. Kumar-Sinner, Susanne; Zugaro-Merimi, Tiziana: Was ist Informatik? – Eine Begriffsklärung, Technische Universität Dresden – Fakultät Informatik (Hrsg.), 2006, https://tu-dresden.de/ing/informatik/ressourcen/dateien/studium/dateien/sonstige_dokumente/zugangsvoraussetzungen/zv_Was_ist_Informatik_060509.pdf?lang=de, abgerufen am 26.01.2021, S. 1.

13. Lackes, Richard: Hardware – Wirtschaftsinformatik/Informatik, in: Gabler Wirtschaftslexikon, Springer Fachmedien Wiesbaden GmbH (Hrsg.), https://wirtschaftslexikon.gabler.de/definition/hardware-34131, abgerufen am 26.01.2021.

14. Mahn, Jan: Anmelde-Baukasten – Logins für Websites mit WebAuthn, in: C't 19/18, Heise Medien GmbH & Co. KG (Hrsg.), S. 26ff.

15. Mainka, Christian et al.: SoK: Single Sign-On Security – An Evaluation of OpenID Connect, Ruhr-Universität Bochum – Fakultät für Elektrotechnik und Informationstechnik – Lehrstuhl für Netz- und Datensicherheit (Hrsg.), https://www.nds.ruhr-uni-bochum.de/media/ei/veroeffentlichungen/2017/01/30/oidc-security.pdf, abgerufen am 25.01.2021.

16. Mayer, McKenzie: Siemens zentralisiert seine Login-Dienste weltweit mit Auth0, auth0.com (Hrsg.), 04. Februar 2020, https://auth0.com/blog/de-siemens-zentralisiert-seine-login-dienste-weltweit-mit-auth0/, abgerufen am 28.12.2020.

17. Mizani, Maximilian: Evaluation von Systemen zur Mehr-Faktor-Authentifizierung am Beispiel des Leibniz-Rechenzentrums, Ludwig-Maximilians-Universität München – Institut für Informatik – Prof. Dr. Helmut Reiser (Hrsg.), 15. Januar 2019, http://www.nm.ifi.lmu.de/pub/Fopras/miza18/PDF-Version/miza18.pdf, abgerufen am 25.01.2021.

18. Monjau, Dieter: Was ist ein System?, TU Chemnitz-Zwickau – Fakultät für Informatik – Lehrstuhl Rechnersysteme (Hrsg.), WS 95/96, https://www-user.tu-chemnitz.de/~knmat/V/Alt/Prof%20Monjau/monjau4.pdf, abgerufen am 25.01.2021.

19. Nickel, Cristoph: Sicherheitsanalyse von OAuth 2.0 mittels Web Angriffen auf bestehende Implementierungen, Ruhr-Universität Bochum – Fakultät für Elektrotechnik und Informationstechnik – Lehrstuhl für Netz- und Datensicherheit (Hrsg.), 01. Dezember 2013, https://www.nds.ruhr-uni-bochum.de/media/ei/arbeiten/2014/12/04/OAuth_Security.pdf, abgerufen am 25.01.2021.

20. O. V.: Offener Schlüssel, in: C't 20/4, Heise Medien GmbH & Co. KG (Hrsg.), S. 82.

21. O. V.: Passwörter, Technische Universität Darmstadt – Hochschulrechenzentrum (Hrsg.), 01. September 2018, https://www-cgi.hrz.tu-darmstadt.de/wiki/www/doku.php?id=passwoerter, abgerufen am 01.03.2021.

22. O. V.: PKI und Digitale Signatur, Bundesamt für Sicherheit in der Informationstechnik (Hrsg.), https://www.bsi-fuer-buerger.de/BSIFB/DE/Empfehlungen/Verschluesselung/Verschluesseltkommunizieren/Grundlagenwissen/DigitaleSignatur/digitale_signatur_node.html, abgerufen am 25.01.2021.

23. O. V.: Redirections in HTTP, developer.mozilla.org (Hrsg.), 19. Februar 2021, https://developer.mozilla.org/en-US/docs/Web/HTTP/Redirections, abgerufen am 22.02.2021.

24. Ragouzis, Nick et al.: Security Assertion Markup Language (SAML) V2.0 Technical Overview – Committee Draft 02, Oasis (Hrsg.), 25. März 2008, https://www.oasis-open.org/committees/download.php/27819/sstc-saml-tech-overview-2.0-cd-02.pdf, abgerufen am 28.12.2020, S. 8, 31.

25. Richer, Justin: OAuth 2.0 Token Introspection, Internet Engineering Task Force (Hrsg.), Oktober 2015, https://tools.ietf.org/pdf/rfc7662.pdf, abgerufen am 25.01.2021.

26. Statista GmbH (Hrsg.): Platform-as-a-Service, https://de.statista.com/outlook/tmo/public-cloud/platform-as-a-service/weltweit, abgerufen am 01.03.2021.

27. Statista GmbH (Hrsg.): Prognose Umsatz mit Infrastructure-as-a-Service weltweit bis 2022, 01. Februar 2021, https://de.statista.com/statistik/daten/studie/307025/umfrage/umsatz-mit-infrastructure-as-a-service-weltweit-seit-2010/, abgerufen am 01.03.2021.

28. Suhl, Leena; Wiese, Jörg: Intranet, in: Enzyklopädie der Wirtschaftsinformatik, Universität Potsdam – Lehrstuhl für Wirtschaftsinformatik – Norbert Gronau (Hrsg.), 29. September 2020, https://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/technologien-methoden/Rechnernetz/Intranet/index.html?searchterm=Intranet, abgerufen am 08.02.2021.

29. Volk, Matthias: Vor- und Nachteile der Nutzung von Cloud-Diensten (mit mobilen Endgeräten) in Organisationen und deren Einfluss auf die Nachhaltigkeit, Otto von Guericke Universität Magdeburg – Fakultät für Informatik – Arbeitsgruppe Wirtschaftsinformatik (Hrsg.), 02. Juni 2014, http://bauhaus.cs.uni-magdeburg.de:8080/miscms.nsf/FEA8C8150500AA14C1257449004F79A9/2903471F8768C568C1257DD4004DEDD1/$FILE/Bachelorarbeit%20Matthias%20Volk.pdf, abgerufen am 01.03.2021, S. 31.

30. Weiss, Eugen: SAML-basierte Single Sign On Frameworks, Ruhr-Universität Bochum – Lehrstuhl für Netz- und Datensicherheit (Hrsg.), 20. August 2010, https://www.nds.ruhr-uni-bochum.de/media/attachments/files/2012/03/Weiss_2010_SAML-basierte_Single_Sign_On_Frameworks.pdf, abgerufen am 30.12.2020.

Impressum

Reihe: **Arbeitspapiere Wirtschaftsinformatik** (ISSN 1613-6667)

Bezug: http://wi.uni-giessen.de

Herausgeber: Prof. Dr. Axel Schwickert
Prof. Dr. Bernhard Ostheimer

c/o Professur BWL – Wirtschaftsinformatik
Justus-Liebig-Universität Gießen
Fachbereich Wirtschaftswissenschaften
Licher Straße 70
D – 35394 Gießen
Telefon (0 64 1) 99-22611
Telefax (0 64 1) 99-22619
eMail: Axel.Schwickert@wirtschaft.uni-giessen.de
http://wi.uni-giessen.de

Ziele: Die Arbeitspapiere dieser Reihe sollen konsistente Überblicke zu den Grundlagen der Wirtschaftsinformatik geben und sich mit speziellen Themenbereichen tiefergehend befassen. Ziel ist die verständliche Vermittlung theoretischer Grundlagen und deren Transfer in praxisorientiertes Wissen.

Zielgruppen: Als Zielgruppen sehen wir Forschende, Lehrende und Lernende in der Disziplin Wirtschaftsinformatik sowie das IT-Management und Praktiker in Unternehmen.

Quellen: Die Arbeitspapiere entstehen aus Forschungs-, Abschluss-, Studien- und Projektarbeiten sowie Begleitmaterialien zu Lehr-, Vortrags- und Kolloquiumsveranstaltungen der Professur BWL – Wirtschaftsinformatik, Prof. Dr. Axel Schwickert, Justus-Liebig-Universität Gießen sowie der Professur für Wirtschaftsinformatik, insbes. medienorientierte Wirtschaftsinformatik, Prof. Dr. Bernhard Ostheimer, Fachbereich Wirtschaft, Hochschule Mainz.

Hinweise: Wir nehmen Ihre Anregungen zu den Arbeitspapieren aufmerksam zur Kenntnis und werden uns auf Wunsch mit Ihnen in Verbindung setzen.

Falls Sie selbst ein Arbeitspapier in der Reihe veröffentlichen möchten, nehmen Sie bitte mit einem der Herausgeber unter obiger Adresse Kontakt auf.

Informationen über die bisher erschienenen Arbeitspapiere dieser Reihe erhalten Sie unter der Web-Adresse
http://wi.uni-giessen.de/