A-SIT

Zentrum für sichere Informationstechnologie - Austria

CONCLUSIONS FROM SSI APPLIED IN A TRUST POLICY LANGUAGE

Über die Integration von Self-Sovereign Identity in Policy-Sprachen

Autor:
Lukas Alber
Tel: +43 316 873 - 5559
Mail: lukas.alber@iaik.tugraz.at

Datum: 27.07.2022

Zusammenfassung:
Da dank Digitalisierung zunehmend viele Transaktionen im öffentlichen als auch im privaten Sektor elektronisch durchgeführt werden, wird eine automatisierte Abarbeitung dieser immer wichtiger. Für eine Automatisierung braucht es aber eine Policy bestehend aus Regeln, welche definiert, wann eine Transaktionsanfrage angenommen werden kann. Für traditionelle Identity Management Konzepte gibt es hierfür bereits einige Lösungen. Allerdings, für den dezentralen Identity Management Ansatz namens Self-Sovereign Identity (SSI) gibt es noch keine konkrete Umsetzung. Diese Lücke haben wir im Zuge dieses Projekts geschlossen. Nun wollen wir die nötigen Änderungen resultierend aus der Kombination aus Policy Language und SSI Revue passieren lassen und einen Ausblick auf zukünftige Weiterentwicklungen geben.

Inhaltsverzeichnis

1. Einleitung

Sowohl im E-Government als auch in Privatwirtschaft ist es wichtig schnell und effizient Entscheidungen zu treffen. Speziell Entscheidungen, die Vertrauen in das Gegenüber erfordern, sind oft aufwendig gewissenhaft zu verifizieren. Hier setzten Trust Policy Systeme, wie das von Mödersheim et al. [1] an. Sie erlauben solche sensiblen Entscheidungen anhand einer vorgegebenen Policy automatisiert zu lösen. Die Policy besteht aus einer genauen Beschreibung an Regeln und Bedingungen, an welcher sich die automatisierte Entscheidungsfindung orientiert.

Grundsätzlich besteht das Model aus zwei Seiten, die Anfragende auf einer Seite (z.B. ein Nutzer) und die Entscheidende auf der anderen Seite (oft als Verifier oder Service Provider bezeichnet). Der Service Provider bereitet anfangs eine Policy, seinen Anforderungen entsprechend, vor und kann dann bei eingehenden Anfragen das Trust Policy System automatisiert entscheiden lassen, ob die Anfrage akzeptiert oder abgelehnt wird. Die Anwendungsgebiete für dieses System sind vielfältig. Als Beispiel: Bei Vertragskündigung könnte ein

Mobilfunkanbieter das elektronisch signierte Kündigungsformular des scheidenden Kunden automatisiert prüfen, und beim Erfüllen aller Bedingungen, wiederum automatisiert die Löschung des Kunden im internen System einleiten.

Um solche Vertrauensentscheidungen zu treffen, ist es überwiegend nötig, den Nutzer eindeutig zu identifizieren. Dafür wird meist die elektronische Identität, ausgestellt von staatlicher Ebene, genutzt (z.B. ID Austria). Standardmäßig unterstützt das Trust Policy System von Mödersheim et al. [1] nur eIDAS basierende Lösungen der EU-Mitgliedsstaaten (wobei Trust Translation in andere Trust Schemes als eIDAS möglich sind [1]). Während dieses Model zentralisiert und DNSSEC-basiert ist [2], ist Self-Sovereign Identity (SSI) ein dezentrales Identitätsmanagement System, welches auf die Hoheit des Nutzers über seine Daten achtet und auf Distributed Ledger basiert. Das Ziel dieses Projektes war es, die bereits im Paper von Alber et al. [3] vorgeschlagenen Änderung an der Trust Policy Language (TPL) umzusetzen [4]. Wir wollen nun etwaige Erfahrungen und Schlüsse daraus festhalten und einen Ausblick auf die Zukunft des TPL Systems geben.

2. Hintergrundinformationen

In diesem Kapitel werden die wichtigsten Begriffe rund um Identitätsmanagement, Self-Sovereign Identity (SSI), Verifiable Credentials (VC), Distributed Ledgers (DL) und Trust Policies erklärt.

2.1. Identitätsmanagement

Als Identitätsmanagement wird in der IT die Verwaltung von Identitätsinformationen bezeichnet, mit dem Zweck, eine sichere und verlässliche Identifikation bzw. Authentifizierung eines Nutzers zu ermöglichen. Die Hauptfaktoren, welche alle Identitätsmanagement Systeme gemeinsamen haben, sind folgende: einen Service Provider (SP), welcher einen digitalen Service anbietet und daher eine Authentifizierung verlangt, einen Nutzer, der den digitalen Service des SP nutzen möchte und einen Identity Provider (IdP), welcher die Authentifizierung des Nutzers für den SP übernimmt. Folgende sind die geläufigsten Identitätsmanagement-Modelle:

Das einfachste und wohl älteste ist das sogenannte isolierte Model. Es kombiniert den SP mit dem IdP in derselben Komponente. Service und Identitätsmanagement werden folglich von derselben Entität betrieben. Der Nachteil bei diesem Model ist, dass sich der Nutzer bei jedem SP einzeln anmelden muss. Für Nutzer, der aus sicherheitstechnischen Gründen nicht für jeden Service dasselbe Passwort wiederverwenden sollte, bedeutet das eine fast unzumutbare Bürde, da er sich viele verschiedene merken muss.

Das zentralisierte Identitätsmodell löst dieses Problem, in dem es die Funktionalität des IdP auslagert. So kann ein IdP jetzt auch mehrere SP bedienen. Dadurch muss der Nutzer sich nur mehr einmal am IdP registrieren und kann dann über diesen auf alle Services der SPs zugreifen. Während dies aus Nutzersicht Vorteile bietet, führt es aus sicherheitstechnischer Sicht zu einem Single Point of Failure: Alle Identitätsinformationen, welche von einem Service verlangt werden, werden an einem zentralen Punkt gespeichert. Solche Datensilos sind beliebtes Ziel von Angriffen. Außerdem wird in diesem Modell jede Authentifizierung über den zentralen IdP ausgeführt, welcher nun leicht Informationen sammeln und Nutzer tracken kann. Geläufige Beispiele eines solchen Systems sind die Single Sign-Ons Google Identity[1] und Apple ID[2].

Des Weiteren gibt es noch das föderierte Identitätsmanagement Modell, welches an sich aus mehreren IdPs eines zentralisierten Identitätsmodelles besteht, welche ein Vertrauensverhältnis teilen. In diesem sogenannten „Circle of Trust", delegieren sie sich gegenseitig Authentifizierungsanfragen, ohne die Identitätsdaten selbst zu besitzen. Ein bekanntes Beispiel dieses Modells ist das europäische eIDAS Interoperability Framework [5], welches das grenzübergreifende Authentifizieren zwischen den Mitgliedsstaaten erlaubt.

[1] https://developers.google.com/identity, accessed on 11.07.2022
[2] https://support.apple.com/apple-id, accessed on 11.07.2022

In den bisher kennengelernten Modellen werden die Identitätsdaten am IdP gespeichert. Dieses zentrale Speichern macht aber den IdP zu einem attraktiven Ziel für Angreifer. Um sich das Speichern am IdP zu sparen, werden beim sogenannten Nutzer-zentrischen Modell die Identitätsdaten direkt beim Nutzer gespeichert. Die Identitätsdaten liegen dann z.B. auf dem Smartphone oder auf der Smart Card des Nutzers, geschützt mithilfe eines Secure Elements. Der Nutzer bleibt dadurch im alleinigen Besitz seiner Identitätsdaten, was privacy-technisch von Vorteil ist. Leider findet bei diesem Model immer noch eine Authentifizierung über den IdP statt, welcher als Proxy zwischen Nutzer und SP agiert und somit weiterhin Informationen sammeln kann. Das SSI Model will auch dem Abhilfe schaffen. In Unterkapitel 2.3 wird genauer darauf eingegangen.

2.2. Distributed Ledger

Ein Distributed Ledger (DL) ist eine verbreitete, verteilte Speicher-Technologie. Die Daten werden hierbei verteilt an verschiedenen Knoten gespeichert. Durch ein Consensus Protokoll [6] können die unabhängigen Knoten sich auf einen gemeinsamen Zustand einigen. Beides sorgt für eine gute Widerstandsfähigkeit der Speicher-Technologie. Übrigens bezeichnet der oft propagierte Begriff „Blockchain" eine Teilmenge der DL Speicher-Technologie. Im Allgemeinen lassen sich DL-Technologie nachfolgenden Kriterien kategorisieren: Private oder Public, das sich auf den Lesezugriff bezieht, und *permissionless* oder *permissioned*, das sich auf den Schreibzugriff bezieht [7].

2.3. Self-Sovereign Identity

Self-Sovereign Identity (SSI) [8] ist ein Sammelbegriff für dezentrales, Nutzer-zentrisches Identitätsmanagement, welches dem Nutzer mehr Hoheit über seine Identitätsdaten gibt (siehe Allen's 10 Prinzipien[3]). Das Ziel der SSI-Community ist es, Identitätsdaten unabhängig von zentralen Autoritäten zu machen und Authentifizierung ohne Umwege direkt zwischen Nutzer und Service Provider zu ermöglichen [9]. Im Vergleich zu anderen Nutzer-zentrischen Identitätsmanagement-Konzepten, ist der Nutzer bei SSI nicht auf den Identity Provider (IdP) angewiesen bei einem Authentifizierungsvorgang. Um trotzdem eine Verifizierung zu gewährleisten, werden neu ausgestellte Credentials auf einem Distributed Ledger (im Circle of Trust) registriert (in Form eines Decentralized Identifier (DID) Documents). Erhält also der Nutzer vom IdP ein Credential, so bewahrt er es in seiner digitalen Idenitity Wallet (cf. [10]) auf. Zeigt folgend ein Nutzer seine Identitätsdaten im Zuge einer sogenannten Presentation einem Service Provider, so kann dieser einen Gültigkeitsnachweis (z.B. in Form des DID Documents) vom Distributed Ledger abrufen, ohne den IdP einzubeziehen.

2.4. Verifiable Credentials

Verifiable Credential (VC) ist der Name eines W3C Standard, der versucht ein einheitliches Datenmodel für Attribute und Credentials zu spezifizieren, und sie einer bestimmten Entität oder Person (Subjekt) zuzuweisen[11]. Hierbei werden nicht nur das Subjekt samt zugehöriger Attribute in eine Datenstruktur (W3C empfielt Json[4]) gepackt, sondern auch die ausstellende Autorität und einen Beweis (meist eine Signatur). Verifiable Credentials werden also von Autoritäten ausgestellt, die als zuverlässig gelten. Danach kann der Nutzer anderen Entitäten das beglaubigte Attribut aus dem VC per Verifiable Presentation (VP) zeigen.

2.5. Trust Policy

Eine Trust Policy beschreibt eine Summe an Bedingungen, die nötig sind, um Vertrauen zwischen Parteien herzustellen. Oft sind diese Policies fester Bestandteil der spezifischen Service Provider Implementierung. Müssen Änderungen an der Policy vorgenommen werden, ist die Implementierung direkt davon betroffen.

2.5.1. Policy Sprachen

Trust Policy Sprachen werden genutzt, um komplexe Bedingung abzubilden, unter welchen eine elektronische Transaktion zwischen zwei Parteien zustande kommen darf. Der resultierende Code ist, aber unabhängig von der Service Provider Implementation und kann somit leicht übertragen bzw. geändert werden. Als formale Sprachen

[3] http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity, accessed on 26.07.2022
[4] https://www.w3.org/TR/vc-data-model/#syntaxes, accessed on 26.07.2022

haben Policy Sprachen eine genau spezifizierte Syntax und Semantik, wobei viele dieser Sprachen nicht nur Vertrauensbedingungen (Trust Policies) spezifizieren, sondern auch Zugangsbestimmungen (Access Policies) [12], [13]. Ein frühes Beispiel einer solchen Trust Policy Sprache wurde von Herzberger et al. [12] implementiert.

2.5.2. TPL System

Im Jahre 2019 publizierten Mödersheim et al. [1] eine wissenschaftliche Abhandlung über ihre neu entwickelte Trust Policy Sprache namens TPL (Trust Policy Language) und deren Eigenschaften. Sie erlaubt Service Providern (SP) auf flexible Art und Weise Regeln zu definieren, wann sie eine eingehende Transaktionsanfrage akzeptieren. Da diese Richtlinien im Vorhinein klar definiert werden, ist es dem Service Provider System möglich eigenständige Entscheidungen zu treffen. Die Komponente, die dies im Service Provider System übernimmt, wird Automated Trust Verifier (ATV) genannt. In anderen Worten, der ATV stellt für den SP fest, ob eine eingehende Transaktionsanfrage vertrauenswürdig bzw. akzeptabel ist, anhand der a priori definierten Trust Policy. Um die Frage der Vertrauenswürdigkeit eindeutig zu klären, kann die Trust Policy z.B. die Verifizierung von Signaturen, Zertifikaten und Hashes fordern. Anfänglich wurde TPL für das Lightest EU-Projekt entwickelt [2], [14], [15], später wurde die Sprache als eigenständiges TPL System ausgegliedert, um verschlankt zukünftigen Forschungsprojekten zur Verfügung zu stehen (z.B. [3]).

2.5.3. Universal Resolver

The Universal Resolver[5] ist ein Interoperabilitätswerkzeug, um die Unterstützung verschiedener Ledger und DID-Methoden zusammenzufassen. Es wurde von der Decentralized Identity Foundation[6] für Test- und Entwicklungszwecke initiiert, ist aber nicht für den produktiven Einsatz gedacht. Für eine DID mit Form: did:<did-Methode>:<methodenspezifischer Identifier> kann das Tool die URI auflösen und entsprechend der Methode ein DID Document zur Verfügung stellen.

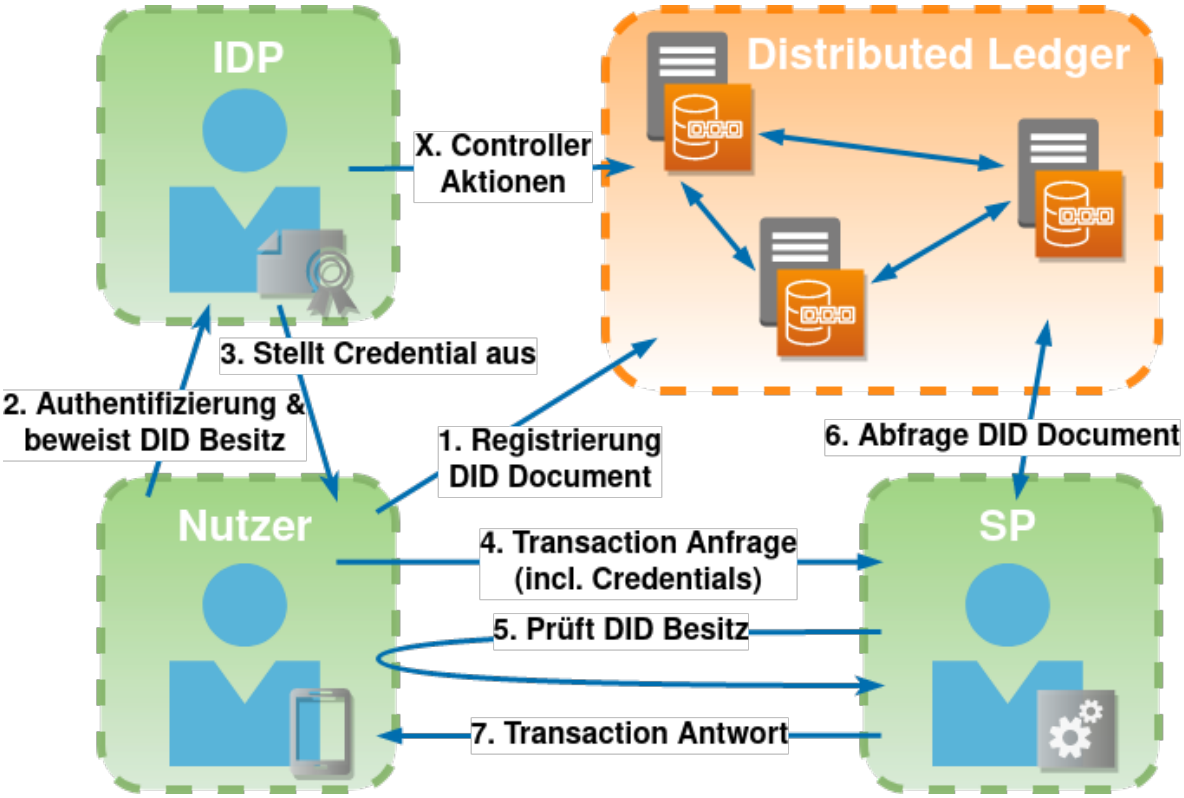

Abbildung 1: Dieses Container Diagramm skizziert den Lifecycle eines SSI Credentials im TPL System.

[5] https://dev.uniresolver.io/, accessed on 11.07.2022
[6] https://identity.foundation/, accessed on 11.07.2022

3. Konzept

In diesem Kapitel erklären wir, wie Self-Sovereign Identity (SSI) in ein verschlanktes TPL System integriert wurde. Dazu verwenden wir Konzepte, welche bereits in Kapitel 2 (Vorwissen) eingeführt wurden. Wir schauen uns die Ausstellung und Registrierung eines Credentials, als auch die Authentifizierung am Service Provider an. In Abbildung 1 wird der zuvor genannte Lifecycle eines SSI Credentials im TPL System schematisch dargestellt.

Im ersten Schritt wird ein dezentraler Identifier (DID) zusammen mit einem kryptographischen Schlüsselpaar in der Wallet generiert. Der DID wird anschließend auf einem Distributed Ledger in Form eines DID Documents registriert. Im nächsten Schritt authentifiziert sich der Nutzer auf sichere Art und Weise am Identity Provider (IdP) seiner Wahl (z.B. anhand eines staatlichen eID Systems wie ID Austria[7]). Zusätzlich beweist der Nutzer, dass er der Besitzer der zuvor generierten DID und des dazugehörigen Key Pairs ist. Das geschieht anhand eines Challenge-Response-Protokolls. Sobald sich der IdP der Identität des Nutzers vergewissert hat, generiert er die gewünschten Credentials anhand seiner vorliegenden Daten. Ein IdP kann natürlich nur Credentials mit Identitätsdaten als Attribute ausstellen, wenn der IdP die Daten einwandfrei beglaubigen kann. Am Ende wird das erstellte Credential vom IdP signiert (und somit offiziell ausgestellt), und dem Nutzer übergeben.

Der Nutzer bewahrt das Credential anschließen unabhängig von Dritten in seiner privaten, digitalen Wallet App auf, z.B. am Smartphone. Um für die Sicherheit der Identitätsdaten zu sorgen, sollte die Wallet unter anderem ein Hardware-Sicherheitsmodul, welches in jedem modernen Smartphone verbaut ist, nutzen. Eine genaue Definition einer Wallet und deren wichtigsten Eigenschaften, speziell aus der Sicht der IT-Sicherheit kann in der systematischen Literaturreview von Podgorelec et al. [10] nachgelesen werden.

Will der Nutzer anschließend eine Transaktion in Wechselwirkung mit dem Service Provider (SP) durchführen, dann sendet er eines oder mehrere Credentials (mit Identitätsdaten als Attribute) samt einer Transaktionsanfrage zum Service Provider. Der Service Provider entscheidet dann, ob er die vorgeschlagene Transaktionsanfrage annimmt, in dem er die Trust Policy ausführt. In letzterer können nun die mitgesendeten Credentials abgefragt werden, um festzustellen, ob der anfragende Nutzer vertrauenswürdig bzw. seine Transaktionsanfrage zulässig ist. Wird ein Credential abgefragt, so wird es auf die Gültigkeit geprüft. Dafür prüft der Service Provider den DID-Besitz mithilfe einer Challenge-Response und fragt das registrierte DID Document am Distributed Ledger nach. Weiters muss der Service Provider noch klären, ob der IdP, der das Credential ausstellte, auch dazu berechtigt war bzw. vertrauenswürdig ist. Dafür benutzt der SP die Vertrauensinfrastruktur aus dem Lightest EU-Projekt (eIDAS trust scheme), oder vertraut auf einen Distributed Ledger, der die DID Documents selbst prüft vor dem Veröffentlichen.

Falls zu einem späteren Zeitpunkt Änderungen am DID Document bzw. am Revokation Status nötig werden, kann (in Schritt X. abgebildet) der Nutzer als sogenannter Controller der DID, eine aktualisierte Form auf den Distributed Ledger hochladen.

4. Diskussion

In diesem Kapitel werden wir einzeln die Schritte des SSI TPL Lifecycles durchgehen (cf. Abbildung 1) und die jeweilige Herangehensweise der prototypischen Umsetzung durchbesprechen. Wir weisen darauf hin, dass diese Implementation sich der DID Methode *key* bedient und somit nicht ein DID Document auf einem Distributed Ledger schreibt. Diese vereinfachte Version kodiert den Public Key direkt im sogenannten *methodenspezifischen Identifier* (cf. Unterkapitel 2.5.3). Somit entfällt Schritt 1 (cf. Abbildung 1), die Registrierung des DID Documents.

[7] https://www.oesterreich.gv.at/id-austria, accessed on 11.07.2022

Die Authentifizierung des Nutzers und die Ausstellung des (Verifiable) Credentials durch den IdP wurden gemockt, da diese nicht zentral für den Prototypen war. Dazu haben wir mehrere Softwarebibliotheken genutzt: für die DID und DID Document Erstellung eine von Digital Bazaar[8] und eine weitere von Danubetech[9], um Verifiable Credentials auszustellen.

Die Transaktionsanfrage (Schritt 4 Abbildung 1) ist ein JSON Objekt, angelehnt an eine Verifiable Presentation, mit folgenden Key/Value-Paaren: Der erste Key *transaction_type* gibt die Art der Transaktionsanfrage an. Hier spielen vor allem die verwendeten Technologien (z.B. SSI, eIDAS, etc.), aber auch der Einsatzbereich des Systems (z.B. Kauftransaktion, Registrierung, etc.) eine Rolle. Dank des *transaction_type* Keys kann das TPL System den geeigneten Parser wählen. Folglich bestimmt dieser Key in weiterer Folge die übrigen Keys in der Transaktionsanfrage. Zum Beispiel der nächste Key *verifiable_credential* wird von dieser SSI Transaktionsart definiert. Er verweist auf ein Verifiable Credential (VC). Des Weiteren besitzt die Transaktionsanfrage den Key *data*. Dieser enthält zusätzliche transaktionsbedingte Daten, welche stark von der Transaktionsart abhängig sind. Zu guter Letzt gibt es noch den *proof* Key. Er enthält die Mittel, um Authentizität und Integrität der Transaktion zu gewährleisten. Zu meist ist dies eine Signatur samt Zusatzinformationen.

Um verschiedene Transaktionsarten, wie oben erwähnt, zu erlauben, unterstützt das TPL System sogenannte Formate. Je nach Transaktionsart spezifiziert ein eigenes Format den Aufbau der Transaktion. Der Name des Formats wird im bereits erwähnten *transaction_type* Key der Transaktionsanfrage angegeben. Aber das Format spezifiziert nicht nur die Struktur, sondern definiert auch die möglichen Aktionen und deren Logik, welche auf

```
{
  "transaction_type" : "transactionFormat",
  "verifiable_credential" : {
    "@context" : ...,
    "type" : [ "VerifiableCredential", "BirthdateCredential" ],
    "id" : ...,
    "issuer" : "did:key:t3GkiB7DQAaevLaxUtarRuyyrfetEomwoUaPzurruGgg54gG",
    "issuanceDate" : ...,
    "credentialSubject" : {
      "id" : "did:key:g546roMuSBx5sV7JxkuTumPd5gkt9bQGqH5dT5VwfzS9g55g",
      "birthDate" : "1990-08-20"
    },
    "proof" : {
      "type" : "Ed25519Signature2020",
      "created" : ...,
      "domain" : ...,
      "nonce" : "useOnce-v58678m09864",
      "proofPurpose" : "verificationMethod",
      "verificationMethod" : ...,
      "proofValue" : "gih4684OIJHPOIJHoihoiu5766jlkg856IUHLiuohp4558Ji5..."
    }
  },
  "data" : {
  ...
  },
  "proof" : {
    "type" : "Ed25519Signature2020",
    "created" : ...,
    "domain" : ...,
    "nonce" : "useOnce-45bn56u5uh56",
    "proofPurpose" : "verificationMethod",
    "verificationMethod" : ...,
    "proofValue" : "j46KJLOIHPO5684683iojpoiJHP+656ioJOIoiuh6874UIHOI56644..."
  }
}
```

Abbildung 2: Listing einer Transaktionsanfrage, der ein Verifiable Credentials enthält (angepasst von [4]).

diesen Daten möglich sind. Neben den verschiedenen möglichen Formaten für die Transaktion gilt dasselbe auch für das DID Document.

[8] https://github.com/digitalbazaar/did-method-key, accessed on 11.07.2022
[9] https://github.com/danubetech/verifiable-credentials-java, accessed on 11.07.2022

Eine Aktion ist für die SSI-Unterstützung neu hinzugekommen: *resolveDID*. Wie der Name schon suggeriert, ist das Ziel das DID Document mithilfe des DID aufzulösen bzw. herunterzuladen. Dies wurde per API Aufruf an den Universal Resolver gelöst, um eine Unterstützung verschiedener Ledger und DID-Methoden zu ermöglichen. Langfristig ist es aber das Ziel, eine Alternative zum Universal Resolver für den produktiven Einsatz zu finden.

5. Fazit

Dieser Bericht hält nötige Änderungen an Policy Systemen im Generellen und an der TPL Implementation im Speziellen fest, um Kompatibilität zu Self-Sovereign Identity zu erlangen. Diese Inhalte sind besonders wichtig für Policy Systeme im Hinblick auf die baldig erscheinende Revision der eIDAS Regulation[10], welches auf eine Einführung von Identity Wallets abzielt und somit Self-Sovereign Identity weiter in den Fokus rückt.

Um dieses Projekt weiterzuentwickeln, planen wir in Zukunft, die SSI Fähigkeiten des TPL Systems zu erweitern: So wollen wir z.B. ermöglichen, dass verschiedenste Attribute, welche die Policy abfragt, als Zero Knowledge Beweise vom Nutzer geliefert werden, um die Privacy des Nutzers zu verbessern. Der Verifier würde daher nur den Beweis eines Attributwertes verifizieren, anstatt die genauen Attributwerte zu erhalten. Dafür sind mehrere, tiefgreifende Änderungen nötig, welche wir in einem zukünftigen Projektbericht behandeln wollen.

Referenzen

[1] S. Mödersheim, A. Schlichtkrull, G. Wagner, S. More, and L. Alber, "TPL: A Trust Policy Language," in *IFIPTM*, 2019, vol. 563, pp. 209–223.

[2] G. Wagner, S. Wagner, S. More, and M. Hoffmann, "DNS-based Trust Scheme Publication and Discovery," in *Open Identity Summit*, 2019, vol. P-293, pp. 49–58.

[3] L. Alber, S. More, S. Mödersheim, and A. Schlichtkrull, "Adapting the TPL Trust Policy Language for a Self-Sovereign Identity World," in *Open Identity Summit 2021, Copenhagen, Denmark, Juni 1-2, 2021*, 2021, vol. P-312, pp. 107–118. [Online]. Available: https://dl.gi.de/20.500.12116/36506

[4] Bernhard Zenz, "A Trust Policy Language for the Self-Sovereign Identity World," Master's Thesis, Technical University of Graz, (noch in Arbeit befindlich), 2022.

[5] E. Union, "Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on Electronic Identification and Trust Services for Electronic Transactions in the Internal Market and Repealing Directive 1999/93/EC.," *Official Journal of the European Union L*, vol. 257, 2014.

[6] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A Survey of Distributed Consensus Protocols for Blockchain Networks," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020.

[7] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: a survey," *Int. J. Web Grid Serv.*, vol. 14, no. 4, pp. 352–375, 2018.

[8] A. Abraham, "Self-Sovereign Identity Whitepaper About the Concept of Self-Sovereign Identity Including Its Potential," *E-Government Innovationszentrum, Graz*, 2017.

[9] B. Zwattendorfer, T. Zefferer, and K. Stranacher, "An Overview of Cloud Identity Management-Models," in *WEBIST (1)*, 2014, pp. 82–92.

[10] https://www.europarl.europa.eu/RegData/etudes/BRIE/2022/699491/EPRS_BRI(2022)699491_EN.pdf, accessed on 11.07.2022

[10] B. Podgorelec, L. Alber, and T. Zefferer, “What is a (Digital) Identity Wallet? A Systematic Literature Review,” 2022. doi: DOI 10.1109/COMPSAC54236.2022.00131.

[11] M. Sporny, D. Longley, and D. Chadwick, “Verifiable Credentials Data Model 1.0,” 2019. [Online]. Available: https://www.w3.org/TR/2019/REC-vc-data-model-20191119/

[12] A. Herzberg, Y. Mass, J. Mihaeli, D. Naor, and Y. Ravid, “Access Control Meets Public Key Infrastructure, Or: Assigning Roles to Strangers,” in *IEEE S&P*, 2000, pp. 2–14.

[13] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. D. Keromytis, “The KeyNote Trust-Management System Version 2,” *RFC*, vol. 2704, pp. 1–37, 1999, doi: 10.17487/RFC2704.

[14] B. P. Bruegger and P. Lipp, “LIGHT^est - A Lightweight Infrastructure for Global Heterogeneous Trust Management,” in *Open Identity Summit*, 2016, vol. P-264, pp. 15–26.

[15] H. Roßnagel, “A Mechanism for Discovery and Verification of Trust Scheme Memberships: The Lightest Reference Architecture,” in *Open Identity Summit*, 2017, vol. P-277, pp. 81–92.