HOCHSCHULE
HANNOVER
UNIVERSITY OF
APPLIED SCIENCES
AND ARTS
–
Fakultät IV
Wirtschaft und
Informatik

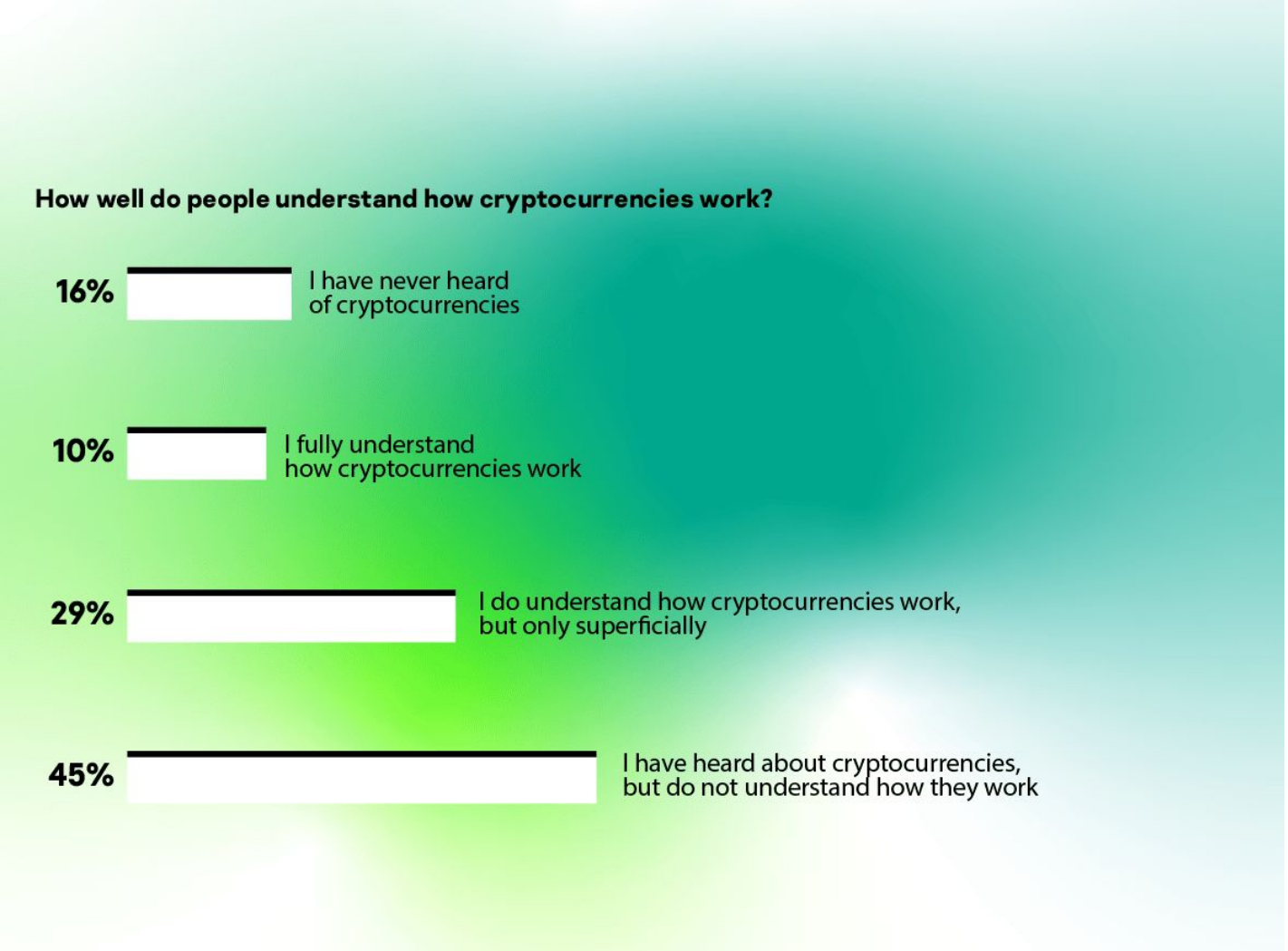
Seminarthema: DashPay

DashPay als Beispiel für ein dezentrales, nutzerfreundliches Zahlungssystem

Marc Herschel, 27.05.2021

Inhaltsverzeichnis

Kryptowährungen im alltäglichen Zahlungsverkehr



Quelle: https://www.kaspersky.com/blog/cryptocurrency-report-2019/

Kryptowährungen im alltäglichen Zahlungsverkehr

- Im Jahre 2021 bieten Kryptowährungen für durchschnittliche Verbraucher noch immer ein unangenehmes Nutzererlebnis, da viel mehr Vorwissen als für z. B. Online Banking oder PayPal benötigt wird.
- Kryptische Adressen wie z. B. **0x89136a83664fa0673930be34463e444260775dc** oder **3GVBSgLLLjAoNRKxw5hm7kANN2P2mEQJy** sind Teil des Problems.
- Fehleranfällige Eingabe: Daher meist Copy & Paste oder Einscannen eines QR-Codes (am PC ohne Webcam z. B. nicht immer möglich).
- Tipps aus der Krypto-Sphere wie „*Sendet erst mal eine kleine Menge, um zu schauen, ob eure Adresse auch korrekt angegeben ist!*“ zementieren das Problem noch mal.

Kryptowährungen sind in den meisten Fällen vollständig dezentralisiert! Es gibt keinen Manager, den man kontaktieren kann, falls eine Transaktion an die falsche Adresse gegangen ist. Der übertragene Wert ist dann weg!

Kryptowährungen im alltäglichen Zahlungsverkehr

Frage: Wer von euch hier im Raum hat schon mal eine „*Test-Transaktion*“ mit einem kleinen Betrag über z. B. PayPal / per Überweisung getätigt, um zu schauen, ob das Geld auch „*da ist*“?

- PayPal verwendet Email-Adressen / Benutzernamen und kann sogar mit der Telefonnummer verbunden werden, um mit Kontakten zu interagieren.
- IBAN Kontonummern verwenden eine Prüfsumme, um eventuelle Tippfehler zu vermeiden. Falsche Transaktionen können von der Bank zurückgerollt werden.

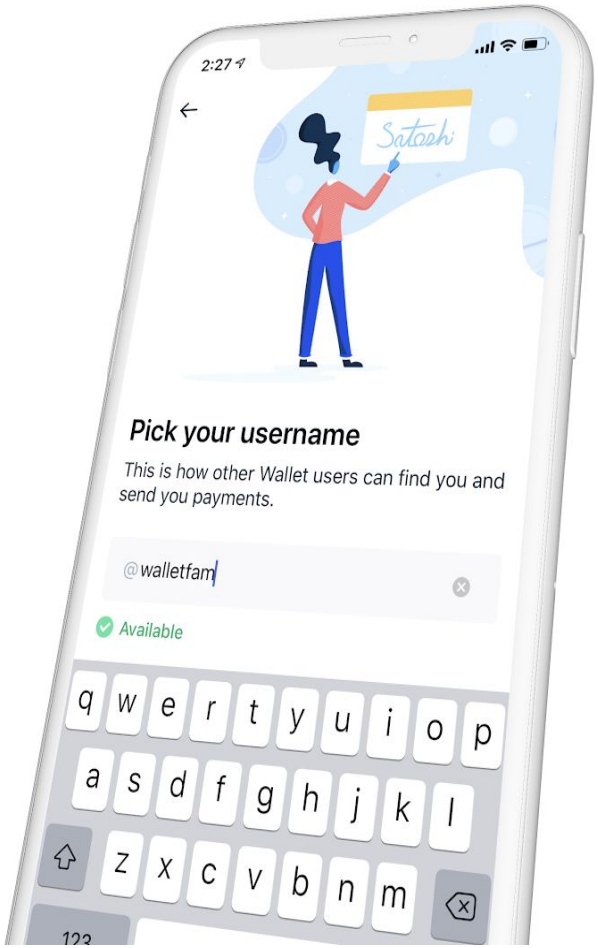
Es gibt bei Kryptowährungen eine Diskrepanz zwischen der Mensch-Computer-Interaktion. Computer verstehen diese Adressen wunderbar, für Menschen sind allerdings greifbare Identifier wie Benutzernamen / E Mail-Adressen oder Kontonummern geeignet.

Kryptowährungen im alltäglichen Zahlungsverkehr

Quelle: https://blog.coinbase.com/send-crypto-more-easily-with-coinbase-wallet-c90a0c84927f

- Um das Problem zu „*lösen*“, bieten einige Exchanges wie z. B. Coinbase ein internes Namensystem an.
- Es ist somit möglich, Beträge ohne Gebühren an Nutzer des Exchanges zu senden.

Problem: Das Ganze ist zentralisiert und es greift „*not your keys, not your coins*“. Sprich, man kann es auch gleich lassen und bei PayPal bleiben.



Was wir eigentlich brauchen:

Ein dezentralisiertes Zahlungssystem, in dem Kontakte bidirektionale, direkte Zahlungskanäle aufbauen können. Kontakte sollten sich durch Namen identifizieren und somit einfache, schnelle und sichere Zahlungen ermöglicht werden.

DashPay strebt die oben genannten Ziele an und wird daher als Beispiel für eine über Dash Platform realisierte dezentrale Anwendung vorgestellt!

Inhaltsverzeichnis

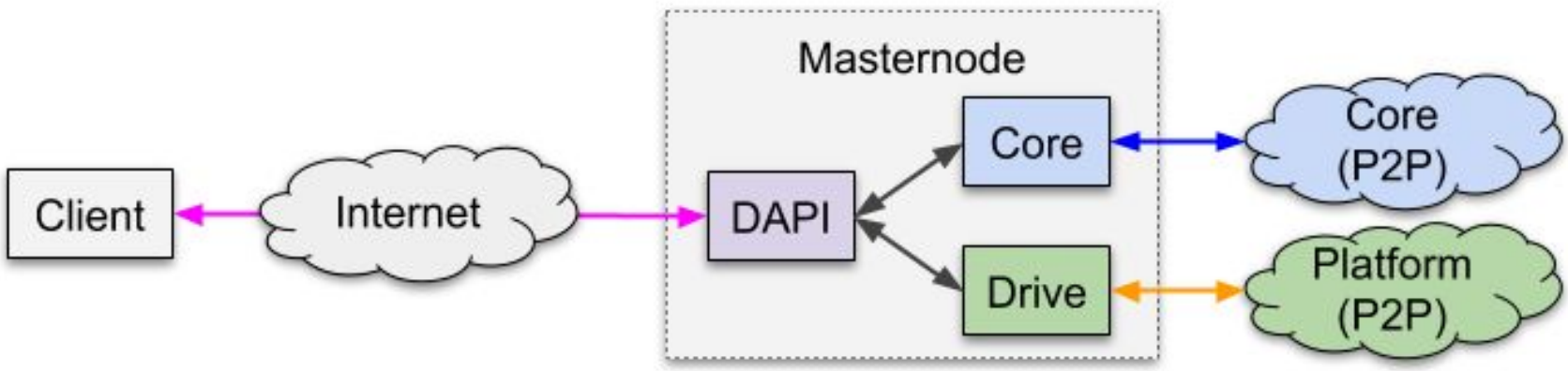
Quelle: Dashdevs.com

Dash Platform

- DashPay baut of Dash Platform, einem Projekt der Dash Core Group auf.
- Technologie-Stack für die Entwicklung von dezentralisierten Anwendungen.

Philosophie:

Statt Smart Contracts als kompilierter Code, der auf einer Blockchain residiert, zu realisieren, werden mithilfe von Data Contracts Dokumentenschematas definiert, über die Dokumente validiert werden. Nur valide Dokumente werden dann in den Speicher von Dash Platform übernommen.



Quelle: https://dashplatform.readme.io/docs/explanation-dapi

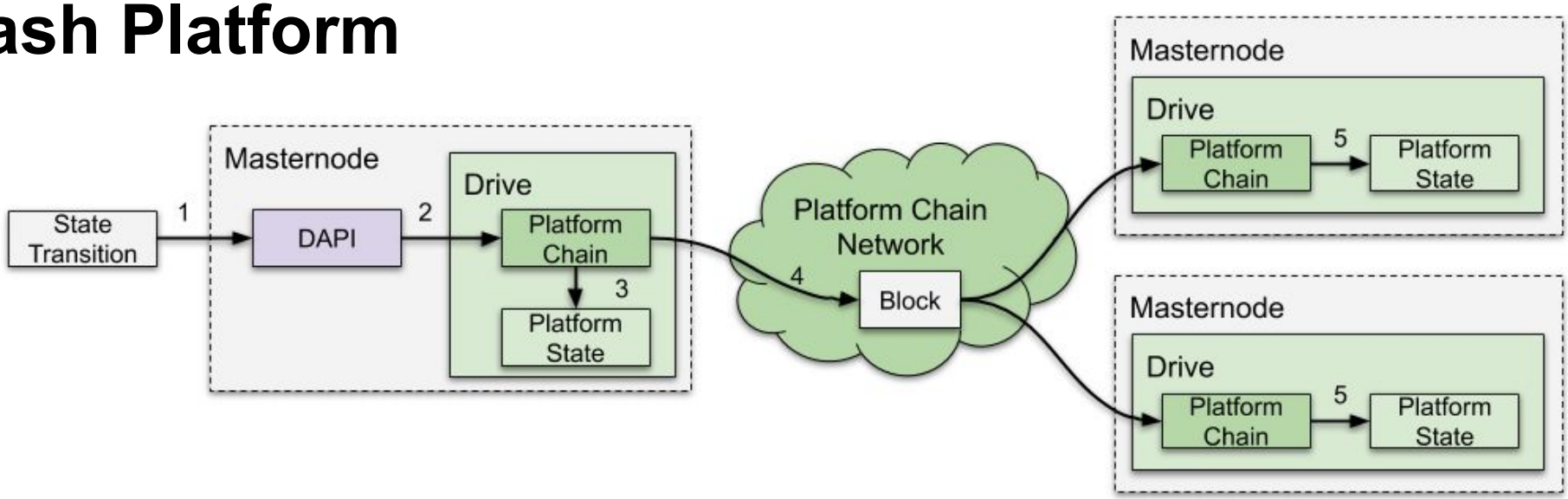
Dash Platform

- Die Persistierung findet über einer Komponente mit dem Namen ***Drive*** statt.
- Klienten können durch die ***DAPI*** über das Masternode-Netzwerk mit ***Dash Platform*** interagieren. Masternodes synchronisieren die ***Platform Chain*** untereinander.
- Aus Gründen der Effizienz: Separate ***Platform Chain*** mit niedriger Blockzeit -> Zustandsänderungen in ≤ 10 Sekunden möglich (statt 2,5 Minuten auf der Core Chain)
- Consensus: ***Byzantine Fault Tolerance*** -> Keine Miner notwendig (PoW) + deterministische Gebühren für Zustandsänderungen berechenbar.

Es handelt sich also um eine verteilte Dokumentendatenbank, mit der dezentralisierte Anwendungen interagieren. Der Code einer Anwendung selber ist austauschbar und stellt lediglich einen Klienten, der sich an die Regeln der Data Contracts hält, dar.

Quelle: https://dashplatform.readme.io/docs/explanation-drive

Dash Platform

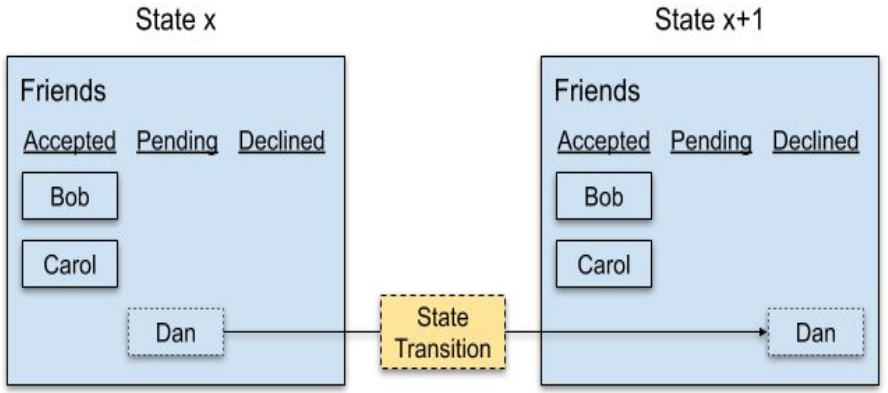


Zustandsänderungen

Werden durch ***State Transitions*** (ST) ausgeführt.

Quelle:https://dashplatform.readme.io/docs/explanation-platform-protocol-state-transition

1.) ST über DAPI einreichen
2.) Masternode validiert, ordnet und fügt die STs zu einem Block auf der Platform Chain hinzu.
3.) Valide STs werden auf den Platform State angewandt.
4.) Block über das Masternode an das Netzwerk propagieren.
5.) Restliche Masternodes synchronisieren ihren Platform State durch die STs in dem Block.

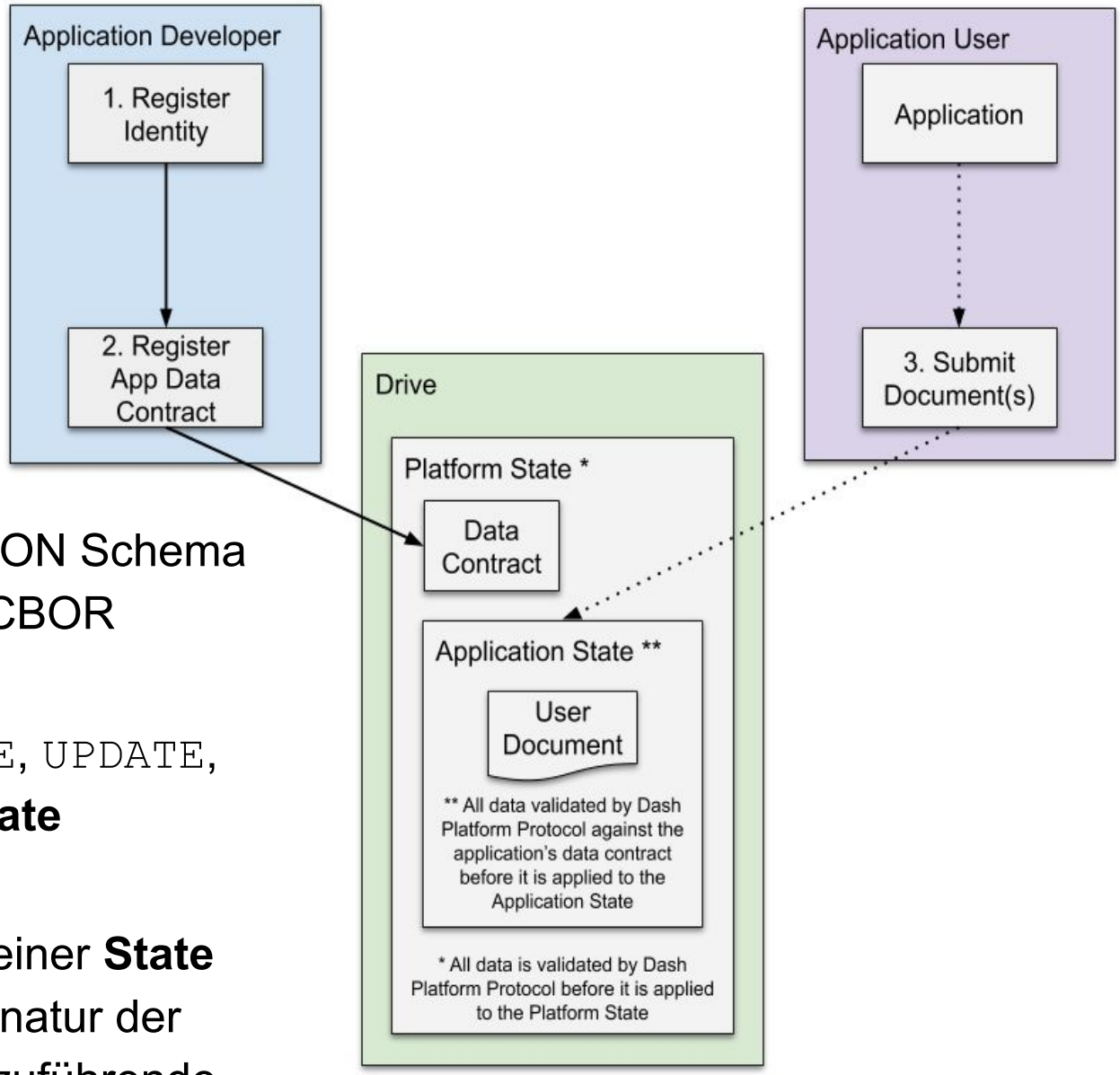


Quelle: https://dashplatform.readme.io/docs/explanation-platform-protocol-data-contract

Dash Platform

Data Contract: Schema um Daten zu validieren.

Document: Daten einer Anwendung, die einem zugeordneten Schema entsprechen.



- **Data Contracts** werden durch JSON Schema definiert und zum Persistieren in CBOR serialisiert.
- Abbilden der Operationen `CREATE`, `UPDATE`, `DELETE` für **Documents** durch **State Transitions**
- Gewährleistung der Authentizität einer **State Transition** durch eine digitale Signatur der beinhalteten Daten durch die auszuführende **Identity**.

Identities

- Legen den Grundstein für Benutzernamen in DashPay.
- Öffentlicher Schlüssel, der auf der Platform Chain gespeichert ist.

Verwendung: Verwaltung von Benutzerprofilen dezentralisierter Anwendungen. (Nur State Transitions mit einer validen Signatur werden für ein assoziiertes Dokument als gültig anerkannt)

Um die Transaktionskosten der State Transitions zu zahlen, ist für jede Identity ein Betrag an gesperrten Dash (per *Lock Transaction*) hinterlegt. Falls diese verbraucht sind, muss ein Top Up geschehen, um den Account aufzustocken.

Identities bieten einen Mechanismus an um Dienste auf Layer 2 durch Zahlungen auf Layer 1 in Anspruch zu nehmen.

Außerdem: Kryptografie durch hinterlegte öffentliche Schlüssel möglich.

Dash Platform Name Service

- Dash Platform Name Service (DPNS) erlaubt es, Identities mit einem menschenlesbaren Namen zu verknüpfen (erste praktische Implementierung eines Data Contracts).
- Namen werden von den Inhabern einer Identity registriert.
- Durch den öffentlich einsehbaren Schlüssel einer Identity können Nutzer die Kontrolle eines Namens beweisen.
- Namen sind als eindeutig unterscheidbar zu betrachten und kompatibel zu DNS.
- Momentan nur Auflösung zu Identities unterstützt. Zukünftig: URLs, Data Contract IDs, ...

Registrierungsprozess durch zwei aufeinanderfolgende Dokumente:

1. *Preorder*-Schritt: Verhinderung von MitM-Angriffen -> Registrierung des Namens ohne den Namen zu enthüllen durch das bilden eines Hashwertes p = h(`Name` + `Salt`).
2. *Registration*-Schritt: Enthüllung des Namens nach Finalisierung von 1. durch aufdecken des Salts + Namen. Beweis zur Validierung der Gültigkeit p == h(`Name` + `Salt`).

Inhaltsverzeichnis

DashPay

- DashPay ist ein Data Contract welches einer dezentralisierten Anwendung den Aufbau von direkten, bidirektionalen Zahlungskanälen zwischen Identities ermöglicht.
- Dank DPNS verstecken sich diese Identities hinter menschenlesbaren Nutzernamen.
- Zahlungen werden zwischen Kontakten, die einmalig untereinander etabliert werden (*friendship*), durchgeführt.
- Durch die eindeutig registrierten Benutzernamen ist jede Identity für DashPay Teilnehmer unverwechselbar zuordenbar.
- Graph der Kontakte wird auf Dash Platform über die Drive-Komponente persistiert.
- Möglichkeit Benutzerprofile zu pflegen (Profilbild, Kurzbeschreibung, …)

DashPay

- Zahlungen können durch einen erweiterten öffentlichen Schlüssel in den hinterlegten Dokumenten des Kontakt-Graphen über die Benutzernamen abgewickelt werden.

- Keine Adressen auf Benutzerebene mehr notwendig, da dieser Schritt von DashPay-Klienten durch Adressableitung weg abstrahiert wird.

- Durch die Verschlüsselung des erweiterten öffentlichen Schlüssels ist eine Offenlegung des Zahlungsverkehrs zwischen zwei Kontakten durch Dritte nicht möglich.

- Kontakt-Graph momentan noch öffentlich einsehbar und somit analysierbar.

- Transparente und vollständige Zahlungshistorie zwischen Kontakten, da beide Parteien jeweils im Besitz der eigenen privaten und gegenseitigen öffentlichen Schlüssel sind (Im klassischen Wallet nicht ohne organisatorischen Aufwand auf beiden Seiten möglich).

DashPay

Um mit DashPay zu interagieren, muss ein Benutzer im Besitz einer Identity sein.

Insgesamt ist DashPay in die folgenden drei Data Contracts unterteilt:

Kontaktanfragen (*contactRequest*):

- Aufbau von Beziehungen zwischen Identities + erweiterten öffentlichen Schlüssel für den bei einer *friendship* zur Verfügung stehenden Zahlungskanal.

Benutzerprofile (*profile*):

- Persistieren von öffentlich einsehbaren Informationen (Profilbild, Biographie, Alias) eines Zahlungsteilnehmers.

Kontaktinformationen (contactInfo**):**

- Persistieren von privaten Informationen bezüglich anderer im Kontakt stehenden Identities (z. B. Spitznamen oder Notizen).

Kontaktanfragen

Kontaktanfragen stellen eine Einwegbeziehzung zwischen einer sendenden und empfangenden Identity dar. Zwei Kontaktanfragen mit vertauschten Absendern bilden eine *friendship*.

Die folgenden Felder sind durch das contactRequest Data Contract vorgegeben:

$ownerId, **toUserId**: Beteiligte Idendeties einer Kontaktanfrage.

$createdAt: Zeitstempel wird mit **$ownerId** und **toUserId** verwendet um eigene und eingehende Anfragen abzurufen. Um Bandbreite zu sparen, wird nach einer Initialisierung nur nach Anfragen mit einem früheren als dem gecachten Zeitstempel gesucht.

senderKeyIndex, **recipientKeyIndex**: Indexnummer, um an der Stelle mit dem privaten/öffentlichen Schlüssel des Senders/Empfängers ein shared secret zu bilden und durch ECDH einen gemeinsamen Schlüssel für den erweiterten öffentlichen Schlüssel zu bilden.

Kontaktanfragen

accountReference: Indexnummer zur Abbildung von mehreren Accounts einer Identity.

encryptedAccountLabel: Zeichenkette um mehrere Accounts zu unterscheiden.

encryptedPublicKey: Verschlüsselter erweiterter öffentlicher Schlüssel.

$coreHeightCreatedAt: Blockhöhe des zuletzt bekannten Blockes mit ChainLock in den letzten 5 bekannten Blöcken der Core Chain. Benötigt, um Zahlungen auf verschiedenen Geräten mit unterschiedlicher Blockhöhe zu erhalten.

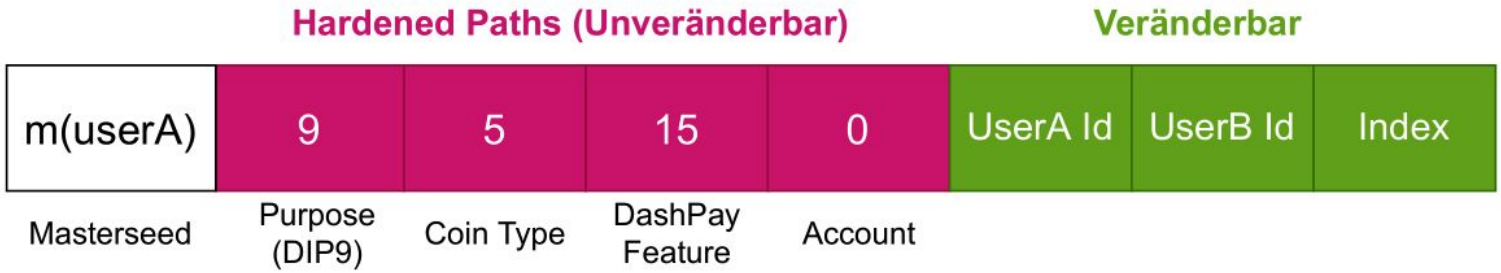
Scenario: A und B haben eine unterschiedliche Blockhöhe. Friendship mit anderer Identity wird gebildet + Tätigung einer Zahlung. B hinkt hinterher und kann durch die Blockhöhe der Anfrage neu synchronisieren und somit die Zahlung für sich sichtbar machen.

Kontaktanfragen

Um nachweislich eindeutige Ableitungspfade zu garantieren, müssen diese 256-Bit lang sein statt der in BIP32 auf 31-Bit limitierten Ableitungspfade.

Dies ist notwendig, um eine Bruteforce-Attacke auf alle möglichen erweiterten öffentlichen Schlüssel (2^31) zu unterbinden.

Mit diesen erweiterten öffentlichen Schlüssel könnte dann der Zahlungsverkehr zwischen zwei Nutzern rekonstruiert werden.



Kontaktanfragen-Dokumente sind unveränderlich und können nicht gelöscht werden, da Nutzer ihren erweiterten öffentlichen Schlüssel nicht ändern sollen.

Um Anfragen automatisch zu akzeptieren kann durch das Feld **autoAcceptProof** ein Schlüssel mitgegeben werden, der es erlaubt die Anfrage der Gegenpartei zu signieren und damit mit einzureichen.

Profile / Kontaktinformationen

Benutzerprofile (*profile*):

$ownerId: Besitzende Identity.

avatarUrl, avatarHash, avatarFingerprint: URL die auf einen Avatar zeigt (optional).

publicMessage: 250 Zeichen lange Biographie in UTF-8.

displayName: 25 Zeichen langer veränderbarer Name in UTF-8.

$createdAt, $updatedAt: Zeitstempel um aktuellstes Profil zu finden.

Kontaktinformationen (contactInfo**):**

- Verschiedene nicht-öffentlich einsehbare Informationen eines Nutzers bezüglich bestehender Kontakte wie z. B. ein Alias oder eine Notiz.
- Mit dem Feld **displayHidden** ist es möglich Kontakte zu ignorieren.

Order of Synchronization

- Synchronization betrifft sowohl Layer 1 (Core Chain) als auch Layer 2 (Platform Chain)
- Klienten sollten daher DIP16 (*Headers First Synchronization on SPV Wallets*) implementieren.
- Anfragen an Dash Platform können erst stattfinden, nachdem die deterministische Liste der Masternodes geladen ist, da eine Verbindung zur DAPI notwendig ist, um die Kontaktdaten zu synchronisieren.
- Blockheader vom Ende der Blockchain sind notwendig, um die List der Masternodes zu generieren (DIP4).
- Anschließend können die Kontakte über Dash Platform synchronisiert werden, um die Zahlungshistorie zu rekonstruieren.

Live Demo

Quelle: https://camius.com/camius_best_suvreillance_ip_cameras_with_advanced_centralized_video_management_software/start-live-demo/

Fazit

Quelle: https://preview.redd.it/24vvmwtfk3t61.jpg

- DashPay befindet sich aktuell in der vierten der fünf Runden des öffentlichen Alpha-Programmes.
- Mithilfe von DashPay wird bewiesen, das auch Kryptowährungen die Benutzerfreundlichkeit bekannter Zahlungsdienste wie z. B. PayPal oder CashApp emulieren können.
- Dash Platform stellt sich mit dem datenorientierten Ansatz als eine zuverlässige Alternative für dezentralisierte Anwendungen gegenüber Smart Contracts heraus.
- Es bleibt abzusehen, wie die Weiterentwicklung von DashPay zukünftig voranschreitet und wann eine Migration aus dem Testnet heraus auf das Mainnet geschehen wird.

Vielen Dank für Ihre Aufmerksamkeit!

Fragen können nun gestellt und beantwortet werden.