iDunion

Remote Attestation for Wallet Authentication

Paul Bastian & Micha Kraus, Bundesdruckerei

Agenda

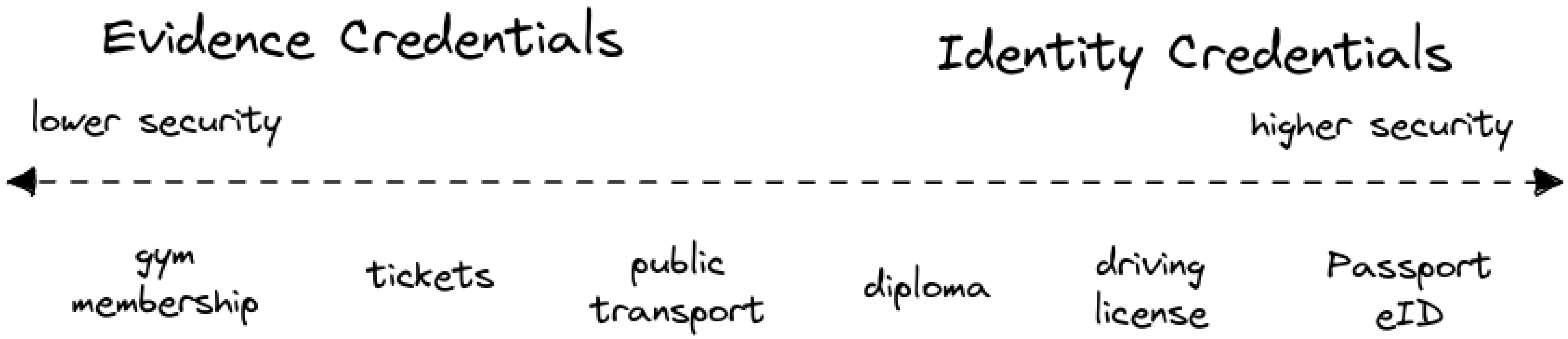
1. Motivation & Building Blocks
2. Activities
3. DIF Wallet Security Approach
4. OIDC4VC Approach
5. Next Steps and Outlook

Motivation

Create a framework for

- wallet ecosystem that supports **freedom of choice** and ensures **interoperable security statement**
  i. Integrity of the Credential
  ii. Authenticity of the Holder
  iii. Authenticity of the Wallet

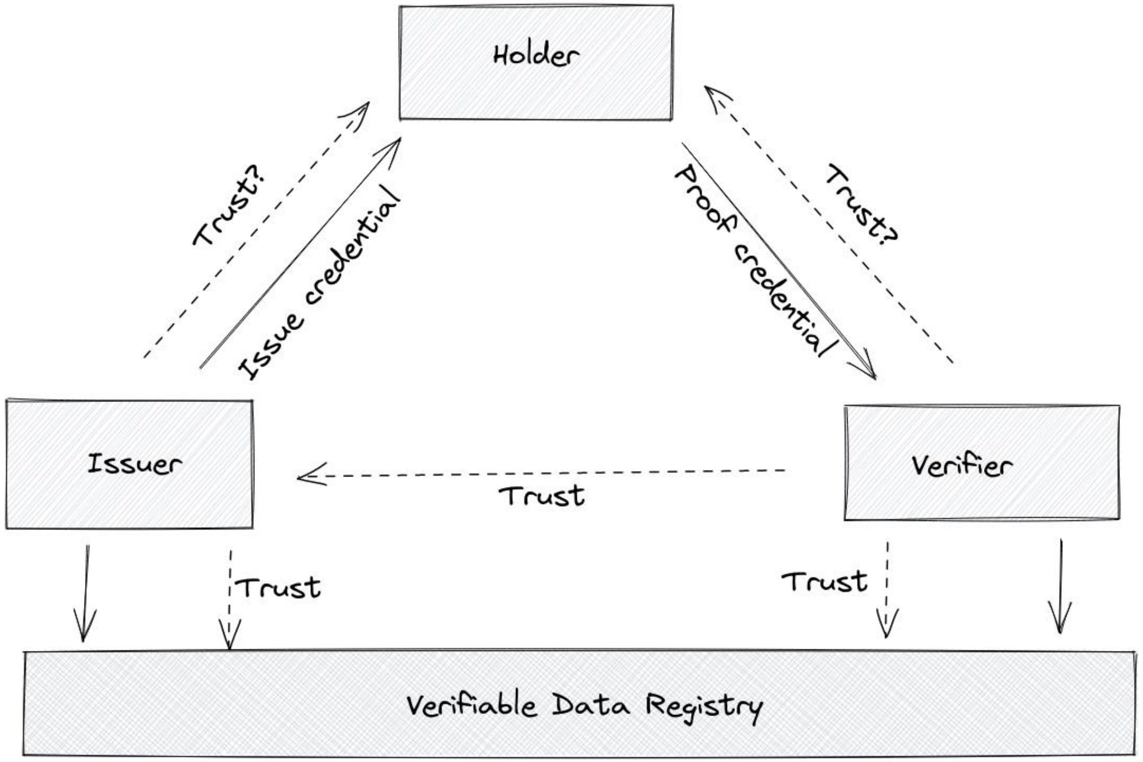
- wallets that support a **variety of applications** with and without regulation



Motivation: The Overlooked Trust Relation

Trust in the SSI Triangle

- Trust relationship to the holder / wallet is mostly overlooked so far
- More security-relevant use cases demand new requirements



Issuer

How can I prevent or hinder missuse of my issued credentials and maintain my credibility at all costs?

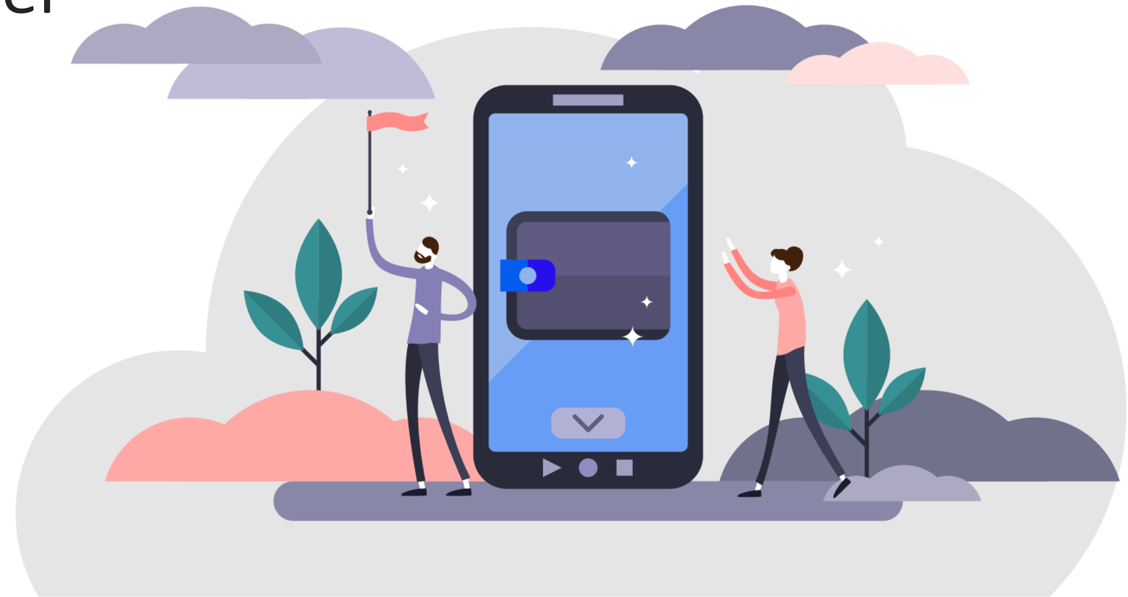
Verifier

Is the holder the rightful owner of this credential and to what degree can he plausibly prove that?

Is the holder‘s authentication strong enough to meet the requirements of my regulated use case?

Secure Wallet Building Blocks

- Binding credentials to the wallet
- Binding credentials to the holder
- Authenticating the wallet



Activities in the last months

DIF Wallet Security WG (Rolling Notes)

- **Wallet authentication/ Device binding with certifying entity**
- W3C/AnonCred Credential Schema Wallet Authentication VC
- RFC0729 Device Binding Attachments

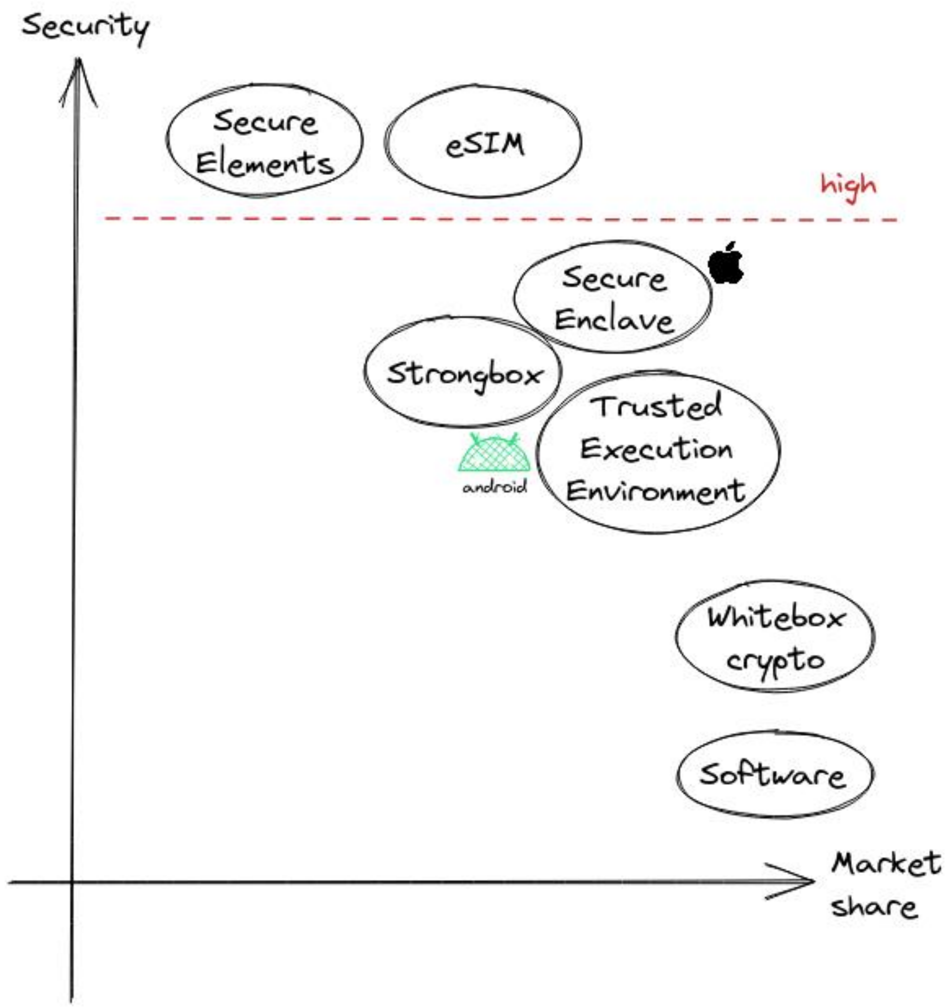
IIW #34

- Presentation and Discussion about wallet security ideas
- Talk: FIDO Authenticator for wallet security

IDunion AP3

- **Integration wallet authentication /device binding to OIDC4VC**
- Implementation building blocks (certifying entity)

The Existing Tools



Mobile Market

- The mobile device market is heavily fragmented
- This makes it difficult to build solutions for high market share
- Different solutions for secure storage
- Relying (partly) on OS security mechanism

Cloud

- possible with HSM in the cloud
- Hybrid approaches feasible
- eIDAS Toolbox and major players focus on mobile market
- Similar problems to mobile if not entire web-wallet

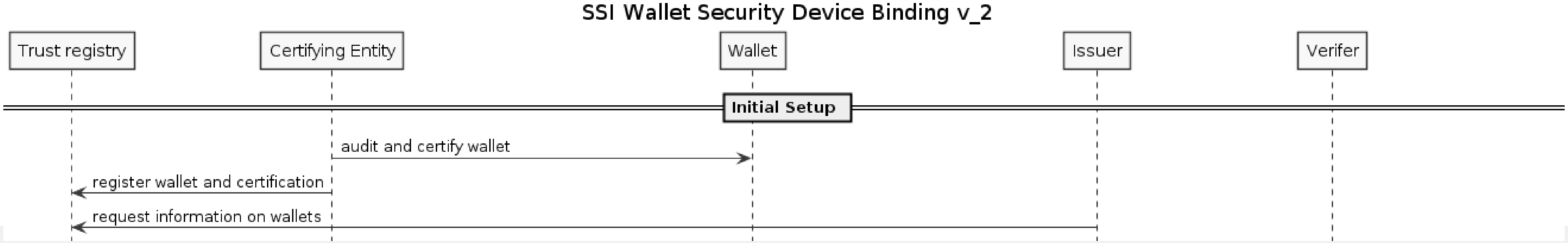
DIF Wallet Security Approach

Combined Approach

- Implement device binding, wallet authentication and holder authentication into one lifecycle
- Legal obstacles(SafetyNet/DeviceChecker) and technical complexity motivate an additional entity to perform the attestations
- Useable with W3C Verifiable Credentials or AnonCreds

Participants in the Lifecycle

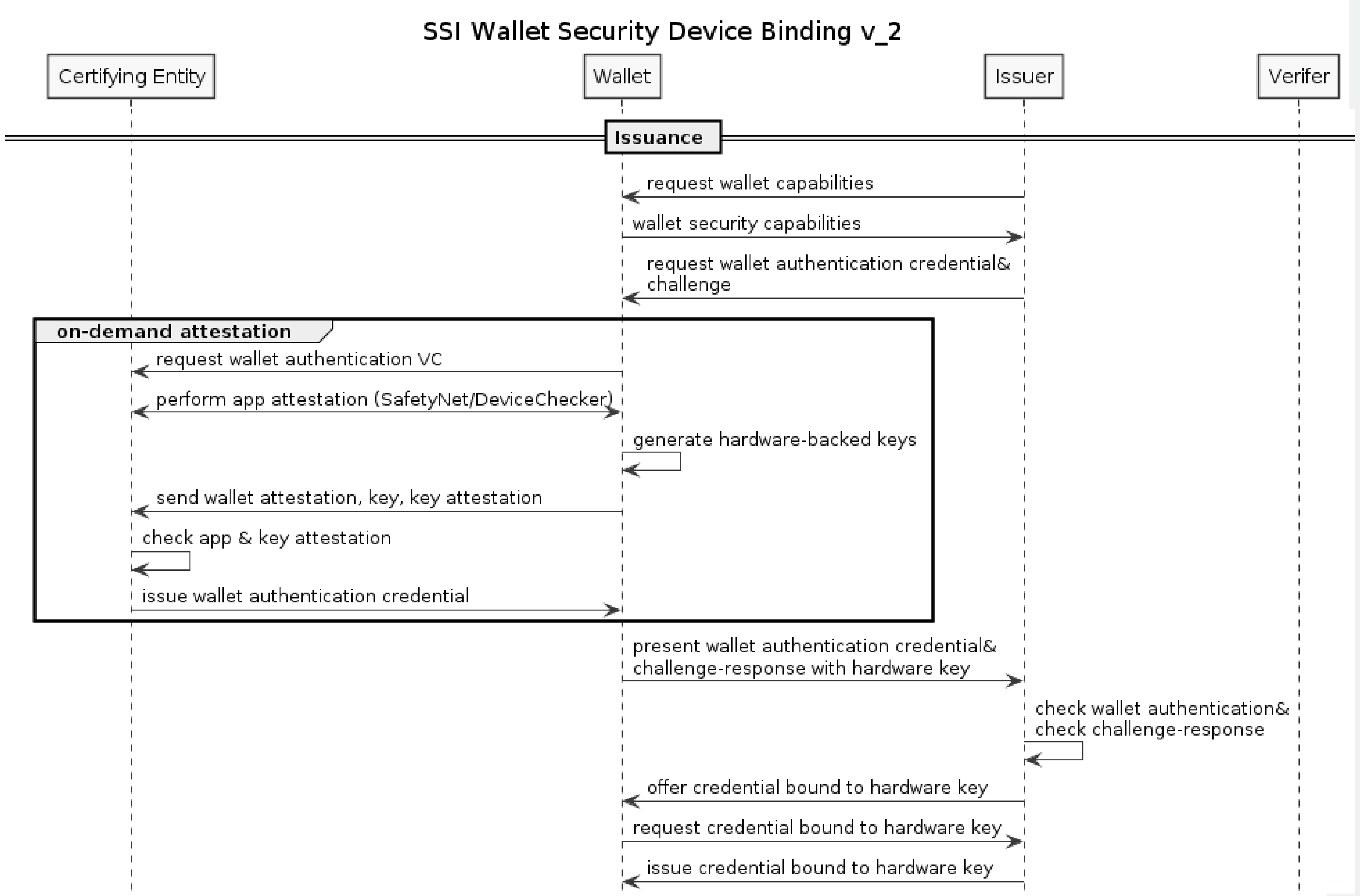
- Issuer / Holder / Verifier
- Certifying entity
  - Attests the device binding and wallet authentication
  - Either the wallet issuer backend service or trust framework-dependent trusted third party
- Trust registry
  - list of multiple wallet issuers and their certification status, legal representation, contact information


SSI Wallet Security Device Binding v_2
Trust registry
Certifying Entity
Wallet
Issuer
Verifer
Initial Setup
audit and certify wallet
register wallet and certification
request information on wallets

DIF Wallet Security Approach

Issuance process

- Integrate the attestation process on-demand instead of upfront at wallet installation
- Use established mechanisms like VC and Present Proof to transmit information
- On-demand advantages
  - Fresh attestations
  - Minimal load on certifying entity
- Interfaces
  - Aries Device Binding Attachments, e.g. for Present Proof v1/v2
  - Certifying Entity <-> Wallet must not be standardized, but can be
- Wallet Authentication VC:
  - identity of the certifying entity
  - wallet name and version
  - hardware public key
  - hardware type and attestation?
  - issuance/expiration date
  - holder authentication mechanism



SSI Wallet Security Device Binding v_2

Certifying Entity

Wallet

Issuer

Verifer

Issuance

request wallet capabilities

wallet security capabilities

request wallet authentication credential& challenge

on-demand attestation

request wallet authentication VC

perform app attestation (SafetyNet/DeviceChecker)

generate hardware-backed keys

send wallet attestation, key, key attestation

check app & key attestation

issue wallet authentication credential

present wallet authentication credential& challenge-response with hardware key

check wallet authentication& check challenge-response

offer credential bound to hardware key

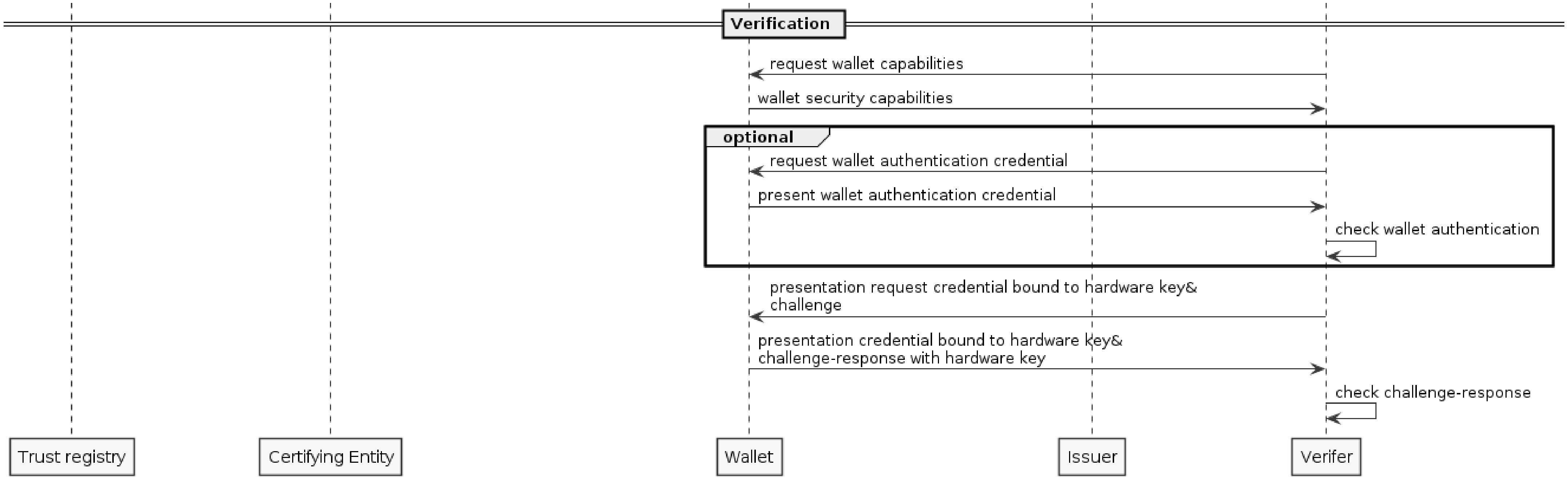
request credential bound to hardware key

issue credential bound to hardware key

DIF Wallet Security Approach

Verification process

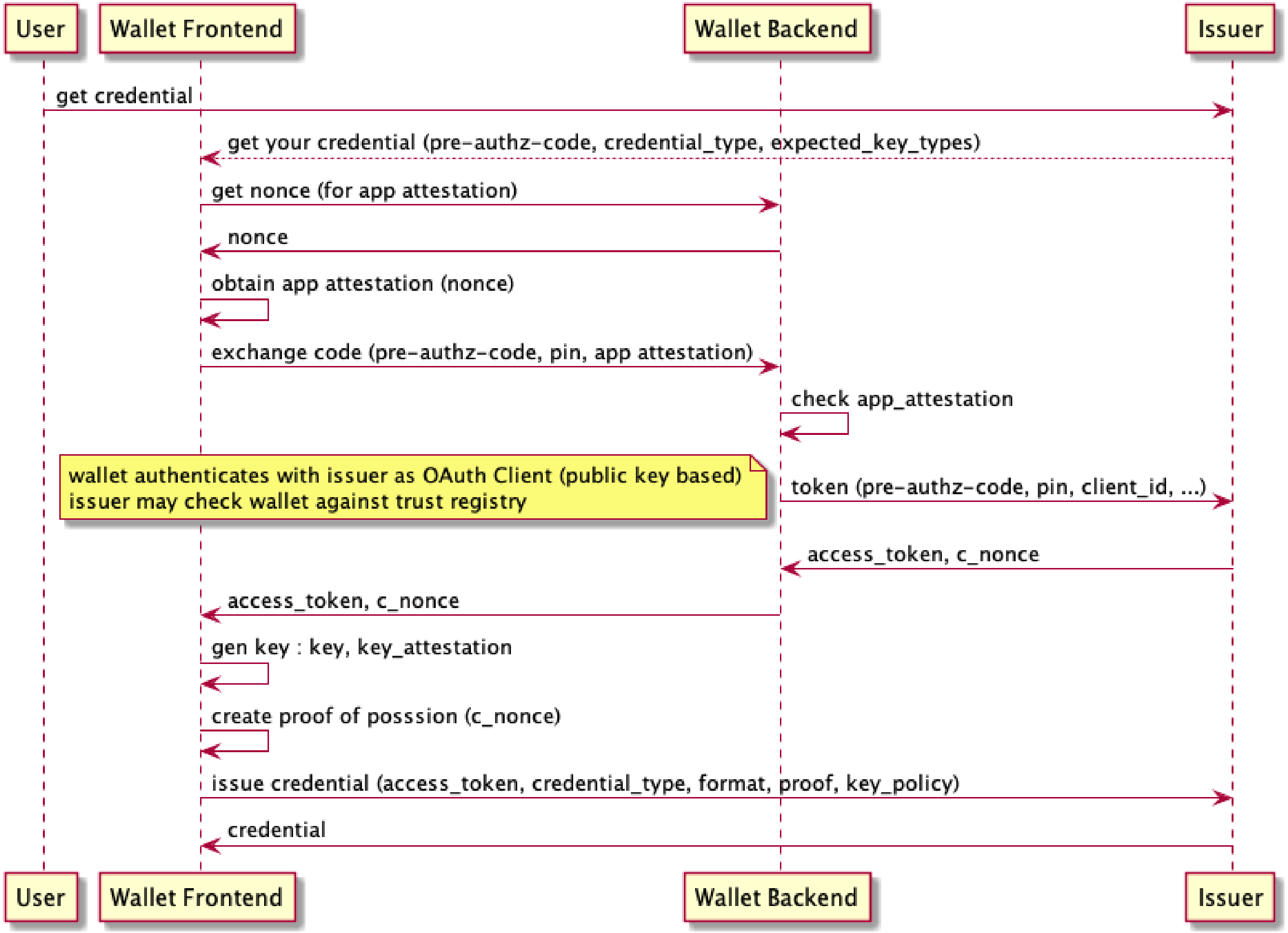
- Regular Present Proof protocol with Aries RFC0729 Device Binding Attachments
- Verifier framework checks challenge-response and matches public key with hardware-bound credential
- Optionally request Wallet Authentication VC and check attestations (actually not necessary if you trust the issuer)


Verification
request wallet capabilities
wallet security capabilities
optional
request wallet authentication credential
present wallet authentication credential
check wallet authentication
presentation request credential bound to hardware key& challenge
presentation credential bound to hardware key& challenge-response with hardware key
check challenge-response
Trust registry
Certifying Entity
Wallet
Issuer
Verifer

OIDC4VC

Issuance process

- app attestation in token endpoint
- key attestation in credential endpoint
- key information is part of credential
- alternative (more privacy preserving):
    - client assertion
    - wallet backend learns nothing about issuance process



User
Wallet Frontend
Wallet Backend
Issuer

get credential

get your credential (pre-authz-code, credential_type, expected_key_types)

get nonce (for app attestation)

nonce

obtain app attestation (nonce)

exchange code (pre-authz-code, pin, app attestation)

check app_attestation

wallet authenticates with issuer as OAuth Client (public key based)
issuer may check wallet against trust registry

token (pre-authz-code, pin, client_id, ...)

access_token, c_nonce

access_token, c_nonce

gen key : key, key_attestation

create proof of posssion (c_nonce)

issue credential (access_token, credential_type, format, proof, key_policy)

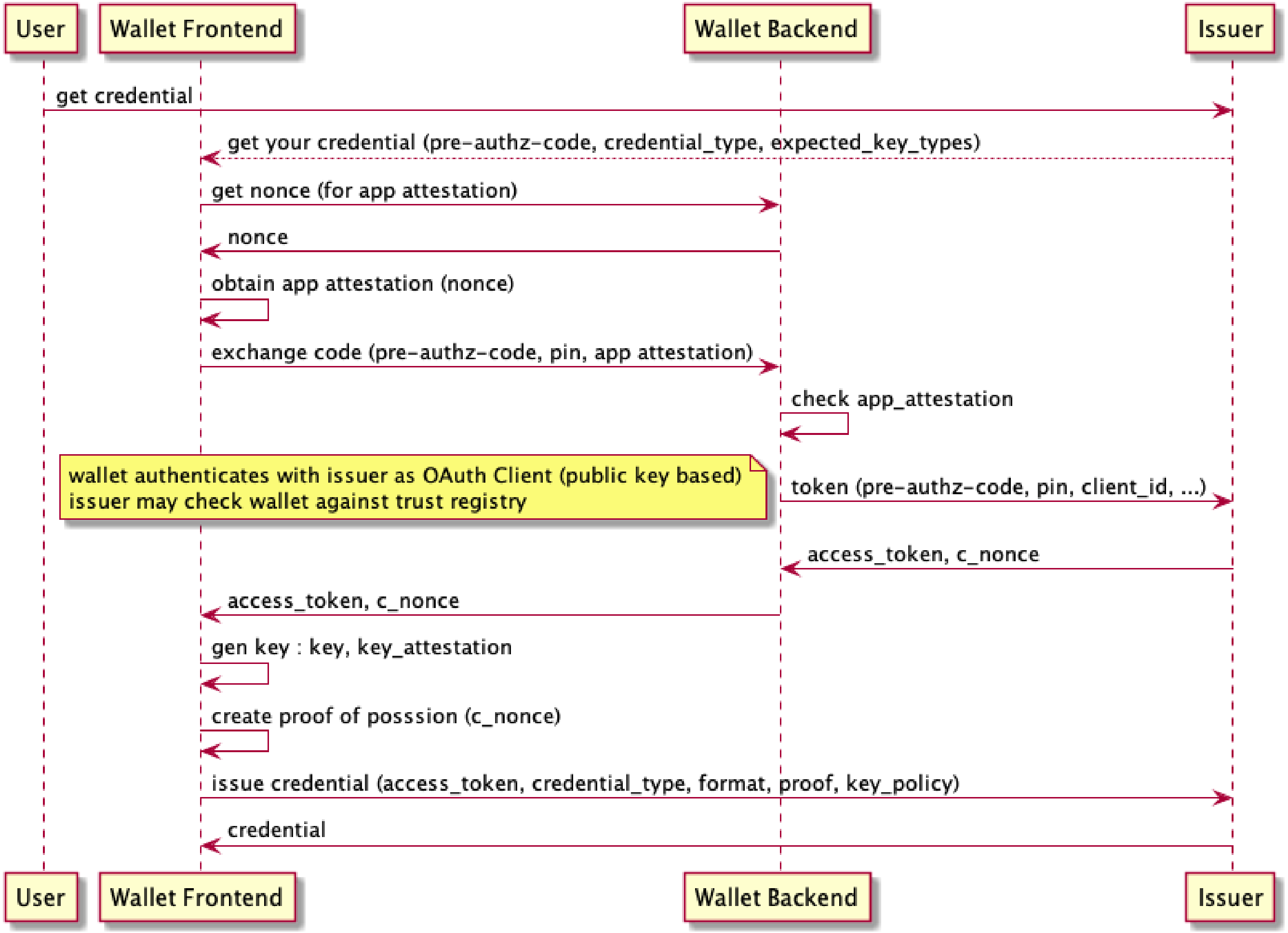
credential

User
Wallet Frontend
Wallet Backend
Issuer

OIDC4VC

Issuance process

- app attestation in token endpoint
- key attestation in credential endpoint
- key information is part of credential
- alternative (more privacy preserving):
  - client assertion
  - wallet backend learns nothing about issuance process



User

Wallet Frontend

Wallet Backend

Issuer

get credential

get your credential (pre-authz-code, credential_type, expected_key_types)

get nonce (for app attestation)

nonce

obtain app attestation (nonce)

exchange code (pre-authz-code, pin, app attestation)

check app_attestation

wallet authenticates with issuer as OAuth Client (public key based)
issuer may check wallet against trust registry

token (pre-authz-code, pin, client_id, ...)

access_token, c_nonce

access_token, c_nonce

gen key : key, key_attestation

create proof of posssion (c_nonce)

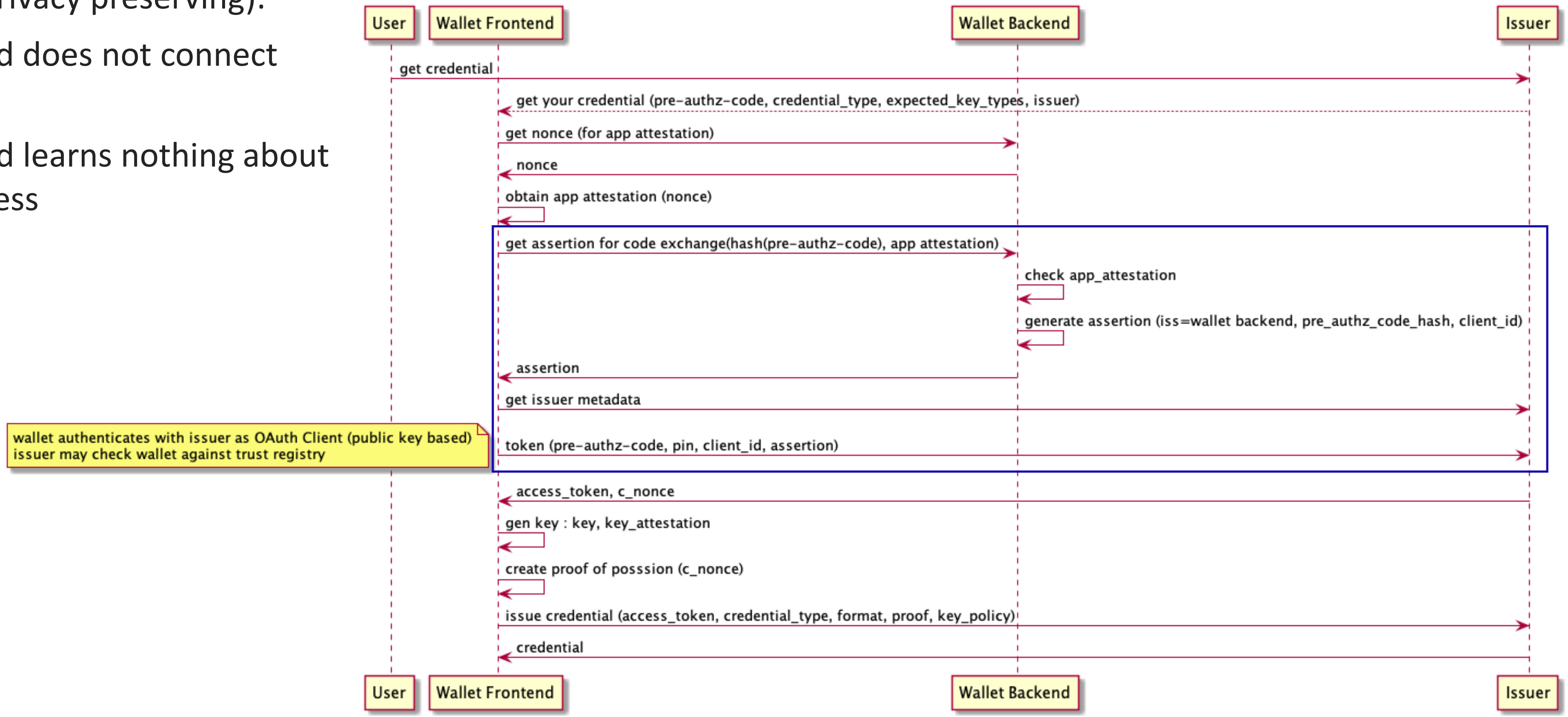
issue credential (access_token, credential_type, format, proof, key_policy)

credential

OIDC4VC

Issuance process (client assertion)

- alternative (more privacy preserving):
  - wallet backend does not connect to issuer
  - wallet backend learns nothing about issuance process



get credential

get your credential (pre-authz-code, credential_type, expected_key_types, issuer)

get nonce (for app attestation)

nonce

obtain app attestation (nonce)

get assertion for code exchange(hash(pre-authz-code), app attestation)

check app_attestation

generate assertion (iss=wallet backend, pre_authz_code_hash, client_id)

assertion

get issuer metadata

token (pre-authz-code, pin, client_id, assertion)

wallet authenticates with issuer as OAuth Client (public key based)
issuer may check wallet against trust registry

access_token, c_nonce

gen key : key, key_attestation

create proof of posssion (c_nonce)

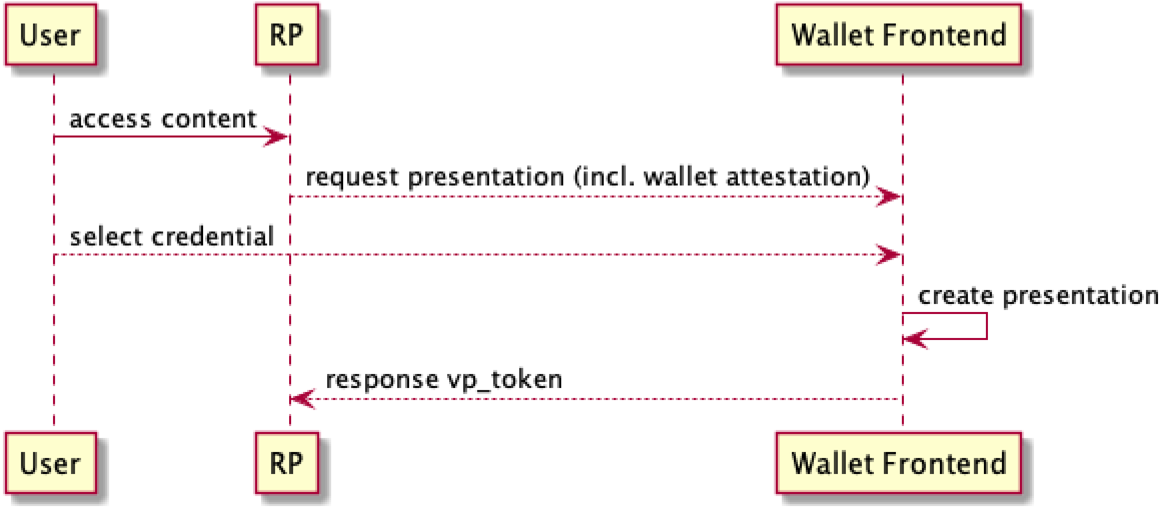
issue credential (access_token, credential_type, format, proof, key_policy)

credential

OIDC4VC

Verification process

- verification flows are matching to DIDComm flows
- verifiers could require app attestation for certain usecases
- not only issuers might require app attestationmore flexiblity if wallet authentication is encapsulated as verifiable credential



Current disussions/Open Questions

- Include the extended key/app attestations into Wallet Authentication VC?
- Implementation and demonstration of Aries RFC 0729
- Alternative Approach to Aries RFC 0729: Linked W3C Device Credential
  - Coupling between both credentials. 1:N, 1:1, N:1
  - Comparison
- Mechanisms for OpenID Connect frameworks
  - OIDC4VC Flows

Summary and Next Steps

Summary

- Successfully developed and tested multiple building blocks
- Improved wallet security for SSI ecosystem

Next steps

- Continue the discussion and work on wallet security
- Bring security mechanisms to standardization at DIF
- implement and test interoperable solutions

Thanks!

Paul Bastian, Bundesdruckerei GmbH
paul.bastian@bdr.de

in @idunion

@IDunion_SCE

contact@idunion.org

https://www.idunion.org/

- **Appendix**

Requirements for Identity Credentials

Requirements from Regulations

- eIDAS LoA / TR-03107 Elektronische Identitäten
  - low, substantial, high
- Evaluation factors:
  - Enrolment
    - Proof of identity
    - Issuance security
  - Multi-Factor-Authentication
    - Possession
    - Knowledge
    - Biometry
  - Revocation
  - Communication security
  - Cryptographic algorithms

- Protection according to ISO18045 attack potential
  - ISO29115 attack vectors:
    - Online guessing
    - Offline guessing
    - Credential duplication
    - Credential theft

Integrity of the Credential

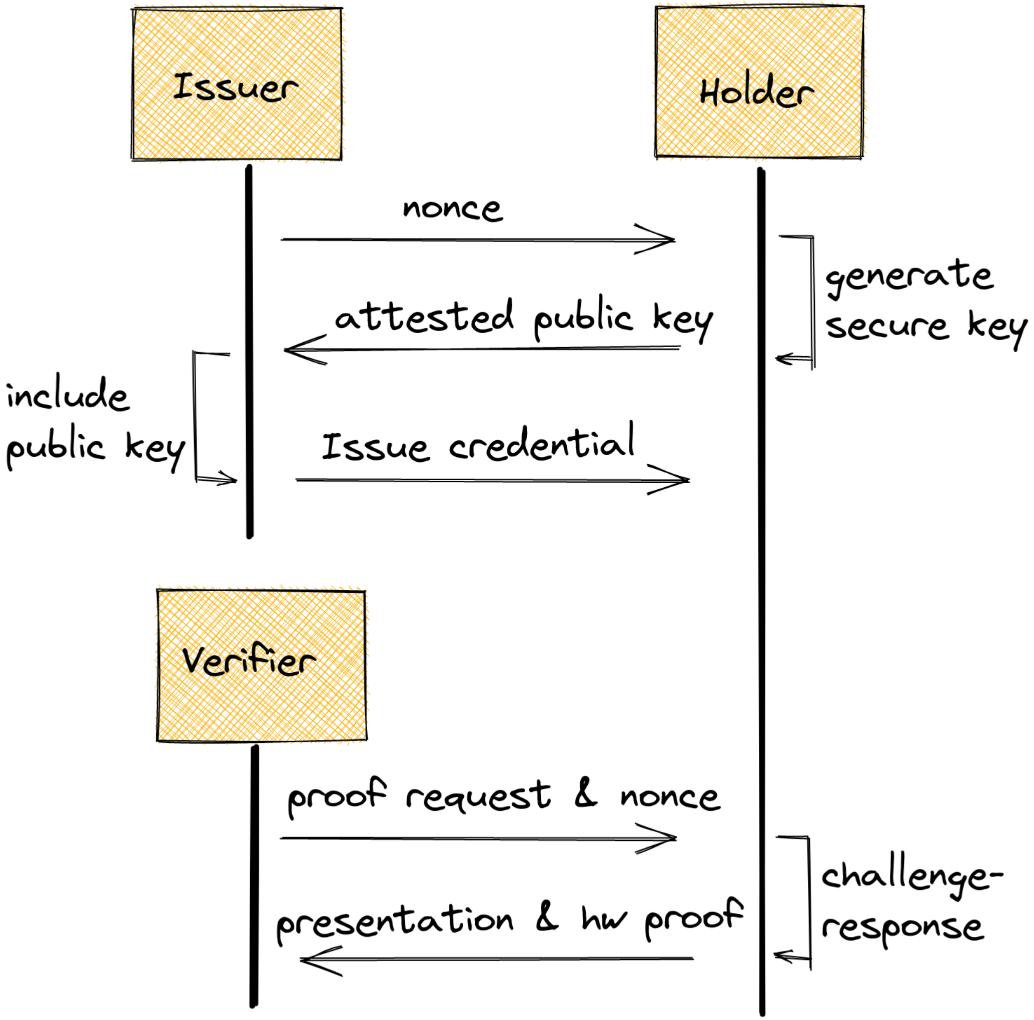
Device Binding Intermediate Solution

- Smallest common denominator for all hardware-backed crypto systems
  - Elliptic Curve NIST P256 with ECDSA-SHA256
  - No support for ZKP in hardware (also for BBS+)
- Simple challenge-response scheme
  - Attested hardware public key as VC attribute
  - Separate challenge-response check
- Pro:
  - DID-method and SSI-stack independent
  - simple, well-understood crypto system
- Contra:
  - No backup & recovery strategy possible (more on this later)
  - Adding a unique, trackable attribute

Device Binding Longterm Solution

- ZKP in mobile hardware takes 5-10 years
- Hybrid cloud?

DIF Wallet Security WG



Issuer

Holder

nonce

generate secure key

attested public key

include public key

Issue credential

Verifier

proof request & nonce

challenge-response

presentation & hw proof

Authenticity of the Holder

Enable Two-factor-authentication

- Knowledge factor (e.g. PIN)
- Inherence factor (e.g. biometrics)

Binding holder to the wallet

- Holder’s authentication reference data is stored in the wallet
- Holder authentication check is performed internally in the wallet
- Wallet is a trusted device that the issuer and verifier must rely on
- better protection for biometric data, but requirement for trusted wallet

Best practices

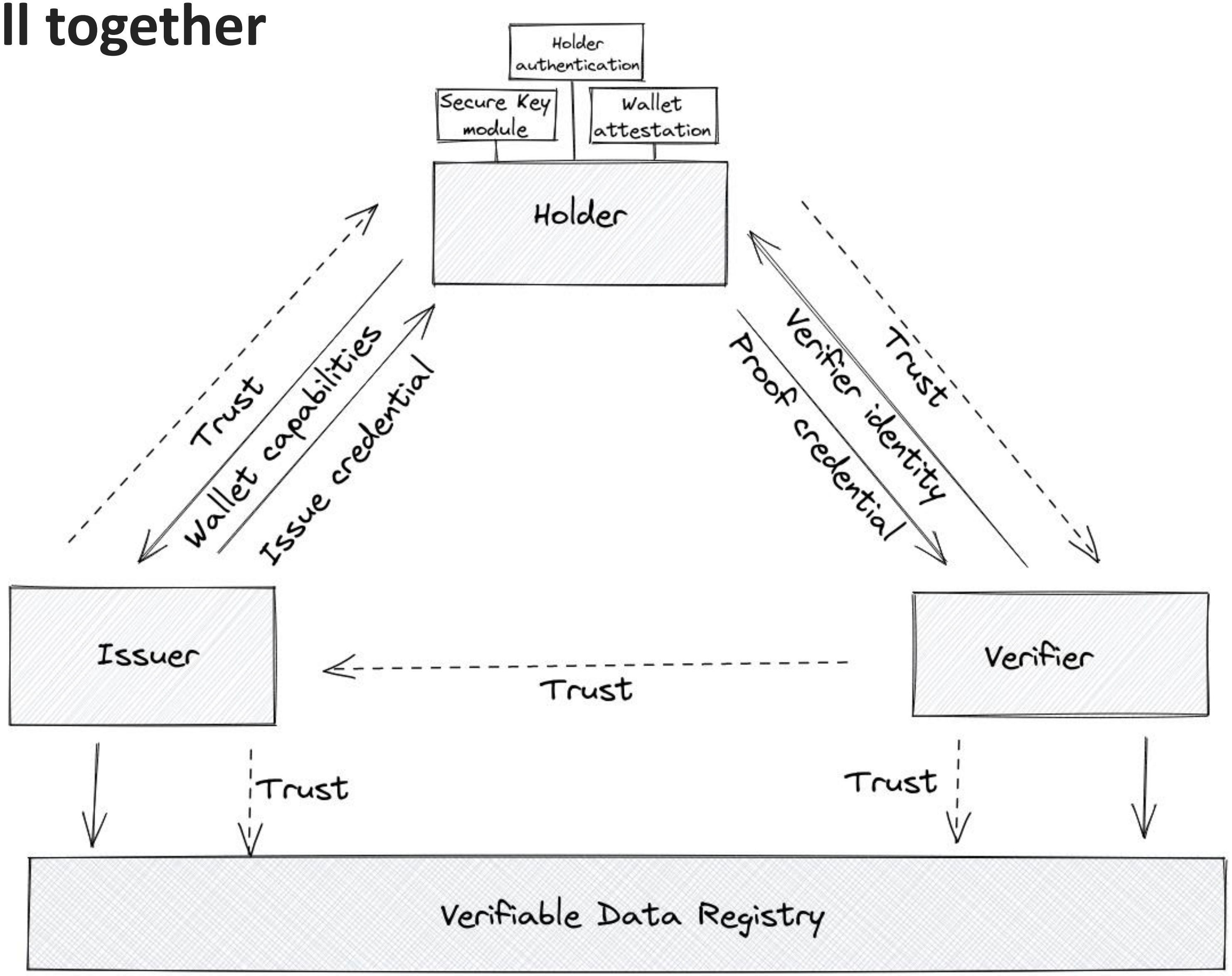
- Biometry on mobile phones is easy to circumvent and not yet sufficient for regulated use cases
  - BSI TR-03166 Technical Guideline for Biometric Authentication Components in Devices for Authentication
- PIN is a secure and necessary method
  - System-PIN (operating system)
  - separate App-PIN or SE-PIN

Authenticity of the Wallet

Wallet Authentication

- mobile OS presents a less-trusted, complex layer in front of trusted, high secure hardware key storage
- use existing mechanisms to verify and increase trust into the mobile phone
  - Android SafetyNet
  - iOS device check
- use key attestations to proof keys were generated in trusted hardware
- additional certification processes are possible
  - Hardware key storage
  - Accompanying mobile phone app

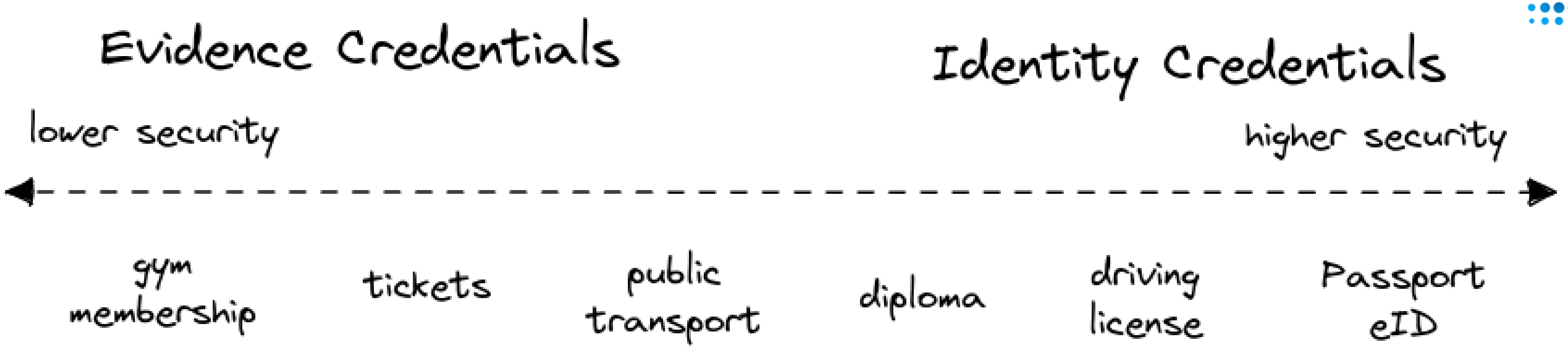
Bringing it all together



- Are we building another boarded-up eID safe?
- Do I really need that much security?
- Is this still self-sovereign identity?

Differential Credential Security Concept

Differential Credential Security

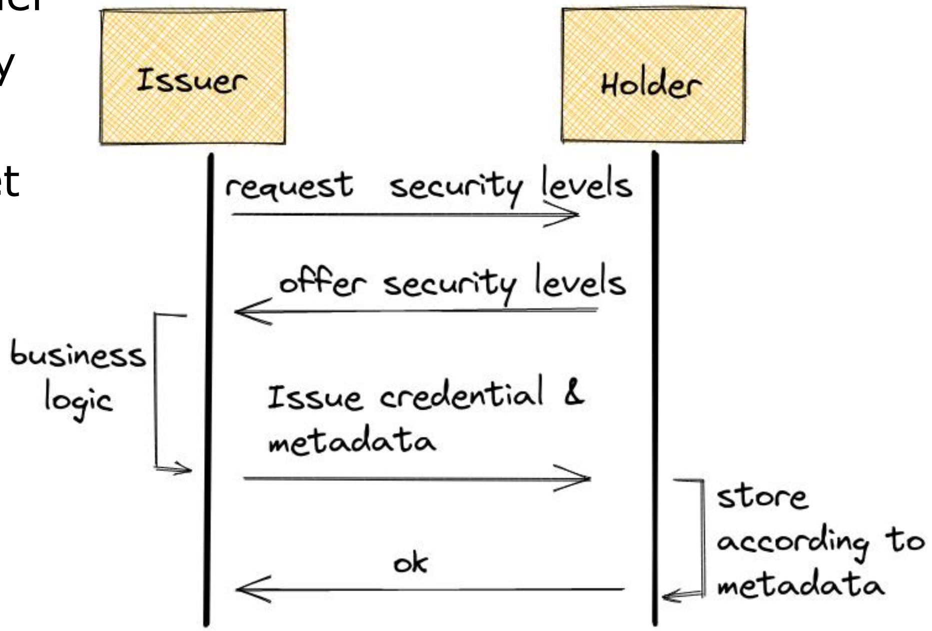


Motivation

- SSI ecosystems brings use cases from different domains together
  - Regulated and non-regulated issuers have different security requirements
- Differential Credential Security model is a core feature for wallet security to address this flexibility
  - Wallet offers multiple LoA based on existing os/hardware
  - Issuer selects an option based on his usecase

Goals

- Pain points of regulations only apply to necessary credentials
- Majority of credentials benefits for convenience like backup and biometrics


Issuer
Holder
request security levels
offer security levels
business logic
Issue credential & metadata
ok
store according to metadata