ODAP

Open Digital Asset Protocol

IETF112 Side Meeting
November 12, 2021

Thomas Hardjono (MIT), Martin Hargreaves (Quant), Ned Smith (Intel), Rama Ramakrishna (IBM)

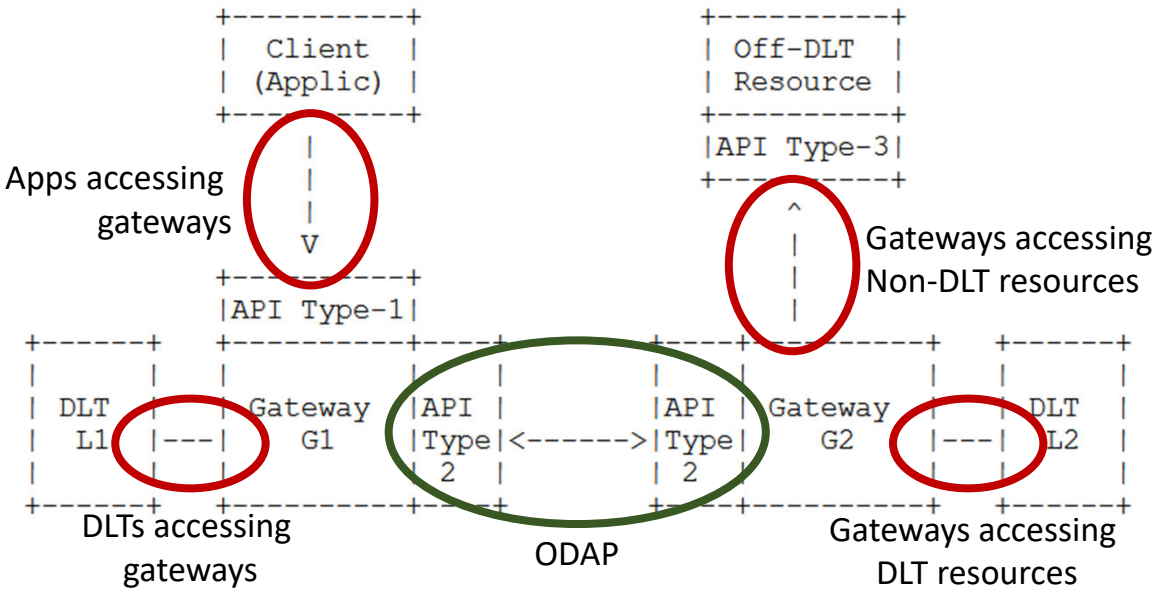
Open Digital Asset Protocol

- A protocol for the exchange of digital assets between distributed ledger networks (DLNs)
- Interoperability between DLNs is low
- Multiple proprietary schemes are proposed, often embedding particular distributed ledger technologies
- We propose a border gateway approach and protocol insulating applications from the underlying DLT / DLNs
- This approach allows standards based transfer of assets between DLNs, typically tokenised securities, debt, currency, data assets and cryptocurrencies

ODAP: Scope

Gateway to Gateway communications

Primarily to atomically transfer data objects between dissimilar DLTs



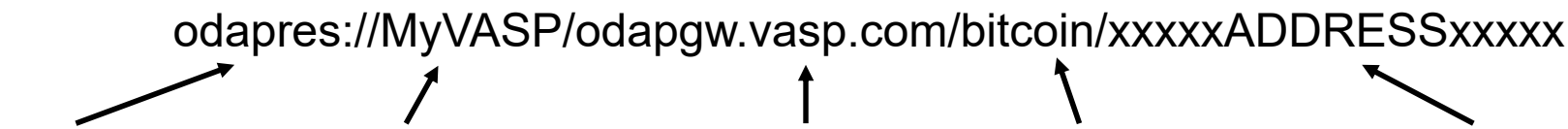
ODAP Draft 3 Coverage

- The phases of the ODAP protocol
- The format of ODAP messages
- The format for resource descriptors
- A method for gateways to implement access controls
- Protocol for negotiating security capabilities
- Discovery and accessing resources and provisions for backward compatibility with existing systems.

ODAP maps DLT resources to RFC 1738 (URL)

Resource addressing for DLTs, using the URL syntax, abstracting a diverse range of underlying schemes.

Client identification based on the URN format. These are for identifying clients (developers and applications) who access these resources, and which in some use-cases require access authorization.



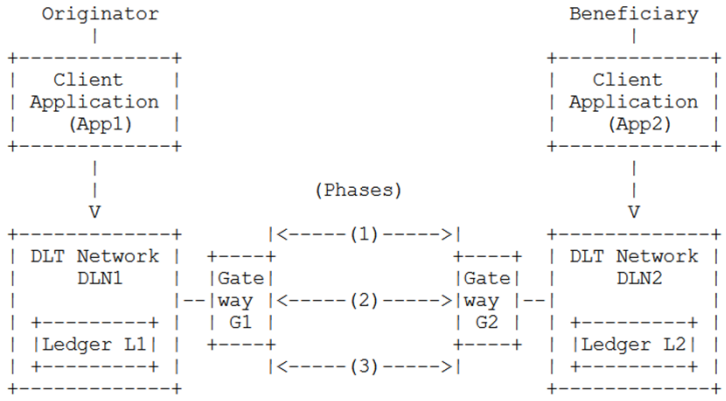
Scheme	Organisation	Gateway FQDN	DLT name	Resource name
application/odapres Application protocol: ODAP	This Legal Entity Identifier (LEI) or other identifier linking resource ownership to real world entity. Any scheme for identifying DLT Gateway owners may be implemented (e.g. LEI directory, closed user group membership, SWIFT BIC, etc.	FQDN of the ODAP compliant DLT gateway	Specific to the gateway behind which the target DLTs operates. This field is local to the DLT gateway and is used to direct ODAP interactions to the correct underlying DLT	Specifies a resource held on the underlying DLT. This field must be meaningful to the DLT in question but is otherwise an arbitrary string.

ODAP defines three core flows

Transfer Initiation flow: This flow deals with the asset profile verification, asset ownership evidence verification and identities verification.

Lock-Evidence flow: This flow deals with the conveyance of evidence regarding the lock (escrow) status of an asset by one gateway, and the verification of the evidence by the other gateway.

Commitment Establishment flow: This flow deals with the asset transfer and commitment establishment between two gateways on behalf of their DLT systems.



ODAP Flows

Negotiation of Security Protocols and Parameters

- TLS Established
- Client offers supported credential schemes
- Server selects supported credential scheme
- Client asserts of proves identity
- Sequence numbers initialized
- Messages can now be exchanged

Transfer Initiation Flow (Phase 1)

- Initialization Request Message
- Initialization Request Message Response (ACK)

Lock-Evidence Verification Flow (Phase 2)

- Transfer Commence Message
- Transfer Commence Response Message (Ack)
- Lock Evidence Message
- Lock Evidence Response Message (Ack)

Commitment Establishment Flow (Phase 3)

- Commit Preparation Message
- Commit Preparation Response
- Commit Final Message
- Commit Final Response Message

ODAP Phases (asset transfer flows)

Phase 1

- Secure channel establishment between G1 and G2
- Mutual device attestations
- Validation of the gateway ownership
- Validation of VASP status
- Identification and validation of asset profile
- Exchange of Travel Rule information and validation
- Negotiation of asset transfer protocol parameters

Phase 2

- Commencement
- Acknowledgement
- G1 lock/escrow asset
- G2 logs incoming asset
- Lock Evidence
- Evidence receipt

Phase 3

- Commit-prepare
- Ack-prepare
- Lock-final
- Commit-final
- Asset-create
- Ack-final
- Location-record
- Transfer complete

ODAP Defines Message Formats

Transfer Initiation Flow Example

Request

- **Version**: ODAP protocol Version (major, minor).
  - **Developer** URN: Assertion of developer / application identity.
  - **Credential Profile**: Specify type of auth (e.g. SAML, OAuth, X.509)
  - **Payload Profile**: Asset Profile provenance and capabilities
  - **Application Profile**: Vendor or Application specific profile
  - **logging_profile** REQUIRED: contains the profile regarding the logging procedure. Default is local store.
  - **Access_control_profile** REQUIRED: the profile regarding the confidentiality of the log entries being stored. Default is only the gateway that created the logs can access them.
  - **Initialization Request Message signature** REQUIRED: Gateway EDCSA signature over the message
  - **Source_gateway_pubkey** REQUIRED: the public key of the gateway initiating a transfer
  - **Source_gateway_dlt_system** REQUIRED: the ID of the source DLT
  - **Recipient_gateway_pubkey** REQUIRED: the public key of the gateway involved in a transfer
  - **Recipient_gateway_dlt_system** REQUIRED: the ID of the recipient gateway involved in a transfer
  - **Escrow type**: faucet, timelock, hashlock, hashtimelock, multi-claim PC, destroy/burn (escrowed cross-claim).
  - **Expiry time**: when will the escrow or lock expire
  - **Multiple claims allowed**: true/false
  - **Multiple cancels allowed**: true/false
  - **Permissions**: list of identities (addresses or X.509 certificates) that can perform operations on the escrow or lock on the asset in the origin DLT network.
  - **Originator address**: along with the source gateway DLT, allows for the correct identification of the entity in the origin DLT transmitting the asset.
  - **Beneficiary address**: along with the recipient gateway DLT, allows for the correct identification of the entity in the destination DLT receiving the asset.
  - **Subsequent calls**: details possible escrow actions
  - **History** (optional): provides an history of the escrow, in case it has previously been initialized. This includes a list of the transactions on that escrow (transaction ID) and which action it performed (ActionCategory), the origin and destination, balance, current status, and ActionLockSpecificParameters.

Response

- **Session ID**: unique identifier (UUIDv2) representing a session.
  - **Sequence Number**: monotonically increasing counter that uniquely represents a message from a session.
  - **ODAP Phase**: The current ODAP phase.
  - **Hash of the Initialization Request Message** REQUIRED: the hash of the proposal.
  - **Destination**: if the recipient gateway calculates the destination address dynamically.
  - **Timestamp** REQUIRED: timestamp referring to when the Initialization Request Message was received.
  - **Processed Timestamp** REQUIRED: timestamp referring to when the Initialization Response Message was constructed.

Implementations (including planned)

- Hyperledger Cactus (code exists) - Open Source
- Hyperledger Weaver (in progress) - Open Source
- Quant Overledger Gateway (2022) - Commercial
- Compellio Gateway (planned) - Commercial

Thank You