SATP Core Protocol
Updates & Overview of Flows

draft-hargreaves-sat-core-02

IETF116 Yokohama

Martin Hargreaves, Thomas Hardjono and Rafael Belchior
Updates to Draft

• Flows in draft-02 now matches v16 of the annotated flow diagram (in Github repo)
  • Session-ID in each message
• Lock (burn) Assertion and Receipt
  • Claims body carries AssetID & AssetProfileID
• Request WG to adopt as Work Item

Annotated message flow diagram: https://bit.ly/3Lzeup1
Lock (Burn) Assertion Fields (Section 8.3)

- session_id REQUIRED: 128-bit value identifying the current transfer-session
- client_identity_pubkey REQUIRED. The client who sent this message.
- server_identity_pubkey REQUIRED. The server for whom this message is intended.
- lock_assertion_claims REQUIRED. The lock assertion claim or statement by the client.
- lock_assertion_format OPTIONAL. The format of the assertion.
- lock_assertion_expiration REQUIRED. Duration of validity of assertion.
- hash_prev_message REQUIRED. The hash of the previous message.
- client_transfer_number OPTIONAL. This number is meaningful only to the client.
- client_signature REQUIRED. The digital signature of the client.
Information in lock_assertion_claims

"gatewayId": "did:gateway:tz1aaYoabvj2DQtpHz74Z83fSNjY29asdBfZ",
"networkId": "tezos:NetXdQprcVkpaWU",
"assets": [
  {
    "assetId": "tezos:NetXdQprcVkpaWU/tzip16:tz1YWK....mVre7xC/1",
    "assetData": {},
    "assetProfileId": "tezos:NetXdQprcVkpaWU/tzip16:tz1CAK1..GG433/1",
    "assetState": "Burned",
    "assetStateTimestamp": "2023-02-22T20:20:39+00:00"
  }
]
Overview of Message Flows

[Diagram showing message flows between two networks, labeled "Network 1" and "Network 2", with gateways and blockchain options.]
3 Stages of Protocol (Burn-Mint)

• Stage 1: Transfer Initiation:
  • Gateways agree on the asset to be transferred

• Stage 2: Lock Assertion & Receipt:
  • Asset Lock Assertion from sending Gateway

• Stage 3: Commitment Establishment
  • Three-Phase Commit (3PC) for ACID properties

Stage 1: Transfer Initiation

- **Stage 0**: Alice request cross-network transfer
  - G1 & G2 validate general network capabilities
- **Stage 1**: G1 & G2 validate transaction-specific parameters [Session-ID]
  - SATP Core - IETF116
Stage 2: Lock-Assertion & Receipt

Stage 2

NW1

G1

2.1
Transfer-Commence

Session open [Session-ID]

ACK-Commence

Use Transaction-ID selected by G1

2.2

2.3A
Perform Lock

If asset is unlocked, G1 performs lock

2.3B
Status-check (locked?)

G1 checks the status of asset.

2.4
Log evidence

G2 logs the received signed evidence

2.5
Lock-Assertion

G1 sends signed assertion (and metadata) of locked status of asset

2.6
Lock-Assertion-Receipt

G2 sends signed Receipt corresponding to the Lock-Assertion (includes hash of 2.5)

G2

NW2
Stage 3: Commitment Establishment

Stage 3

NW1

G1

3.1
Commit-Prepare
Includes hash of msg 2.5 & 2.6

3.2
ACK-Prepare
Acknowledge commit-prep msg

G2

3.3A
Create-Asset
G2 creates asset, assigned to Self

3.3B
Status-check (created/locked?)
G2 checks the status of asset

NW2
Stage 3 (cont)

3.5A Extinguish
G1 extinguishes (disables) asset

3.5B Status-check (deleted?)

3.6 Commit-Final
G1 sends signed message asserting that asset in NW1 has been extinguished.

3.7A Assign-Asset
G2 assigns asset to Bob

3.7B Status-check (finalized?)

3.8 ACK-Final
G2 sends an ACK. It may carry a signed Creation-Assertion (and metadata, including block-number etc.)

3.10 Transfer-Completed
Session closed
ACID Properties

- **Atomicity**: Transfer must either commit or entirely fail (failure means no change to asset ownership)

- **Consistency**: Transfer (commit or fail) always results in asset located in one blockchain network only

- **Isolation**: While transfer occurring, asset ownership cannot be modified (no double-spend)

- **Durability**: Once transaction committed, must remain so regardless of gateway crashes
Thank You and Q&A

Contact: hardjono@mit.edu
Blank

• Header 1
  • Subheader
• Header 2