Abstract

This document proposes an interoperability architecture for the secure transfer of assets between two networks or systems based on the gateway model.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 3, 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction ......................................................... 2
2. Terminology .......................................................... 3
3. Assumptions and Principles .......................................... 4
   3.1. Design Principles .............................................. 4
   3.2. Operational Assumptions ....................................... 5
4. Gateway Interoperability Modes ..................................... 5
5. Architecture .......................................................... 6
   5.1. Goal of Architecture .......................................... 6
   5.2. Overview of Asset Transfer .................................... 7
   5.3. Desirable Properties of Asset Transfer ....................... 7
   5.4. Event log-data, crash recovery and backup gateways ........ 8
   5.5. Overview of the Phases in Asset Transfer .................... 8
6. Pre-transfer Verification of Asset and Identities (Phase 1) .... 10
7. Evidence of asset locking or escrow (Phase 2) .................... 12
8. Transfer Commitment (Phase 3) ..................................... 13
9. Commitment sub-protocol ............................................. 15
10. Security Considerations ............................................ 16
11. Policy Considerations ............................................. 16
12. References ......................................................... 17
   12.1. Normative References ........................................ 17
   12.2. Informative References ...................................... 17
Authors’ Addresses .................................................... 19

1. Introduction

This document proposes an interoperability architecture based on gateways, which are points of interconnection between networks or systems.

There are several services that may be offered by a gateway, one of which being the direct transfer of a digital asset from one network to another via pairs of gateways without a mediating third party.

A given network or system may have one or more gateways to perform a unidirectional direct transfer of digital assets to another network possessing one or more compatible gateway.

Both gateways must implement a secure asset transfer protocol that must satisfy certain security, privacy and atomicity requirements.
The purpose of this architecture document is to provide technical framework within which to define the required properties of a gateway that supports the secure asset transfer protocol.

2. Terminology

There following are some terminology used in the current document. We borrow terminology from NIST and ISO as much as possible, introducing new terms only when needed:

- Asset network (system): The network or system where a digital asset is utilized.
- Asset Transfer Protocol: The protocol used to transfer (move) a digital asset from one network to another using gateways.
- Origin network: The current network where the digital asset is located.
- Destination network: The network to which a digital asset is to be transferred.
- Resource Domain: The collection of resources and entities participating within an asset network. The domain denotes a boundary for permissible or authorized actions on resources.
- Interior Resources: The various interior protocols, data structures and cryptographic constructs that are a core part of an asset network or system.
- Exterior Resources: The various protocols, data structures and cryptographic constructs that are outside of (external to) the network or system.
- Gateway: The collection of services which connects to a minimum of one network or system, and which implements the secure asset transfer protocol.
- Entity public-key pair: This the private-public key pairs of an entity, where the public-key is available and verifiable outside the network. Among others, it may be utilized for interactions other entities from outside the network. The term is used to distinguish this public-key from other key-pairs belonging to the same entity, but which is only available within the (private) network.
o Originator: Person or organization in an origin network seeking
the transfer of a digital asset to a beneficiary located in a
remote network.

o Beneficiary: Person or organization in an destination network
seeking to receive the transfer of a digital asset to from an
originator located in a remote network.

o Gateway device identity: The identity of the device implementing
the gateway functions. The term is used in the sense of IDevID
(IEEE 802.1AR) or EK/AIK (in TPM1.2 and TPM2.0) [IDevID].

o Gateway owner: The entity that owns and operates a gateway within
a network.

3. Assumptions and Principles

The following assumptions and principles underlie the design of the
current gateway architecture, and correspond to the design principles
of the Internet architecture.

3.1. Design Principles

o Opaque network resources: The interior resources of each network
is assumed to be opaque to (hidden from) external entities. Any
resources to be made accessible to an external entity must be made
explicitly accessible by a gateway with proper authorization.

o Externalization of value: The asset transfer protocol is agnostic
(oblivious) to the economic or monetary value (if any) of the
digital asset being transferred.

The opaque resources principle permits the architecture to be applied
in cases where one (or both) networks are private (closed
membership). It is the analog of the autonomous systems principle in
IP networking [Clar88], where interior routes in local subnets are
not visible to other external networks.

The value-externalization principle permits an asset transfer
protocol to be designed for efficiency, security and reliability --
independent of the changes in the perceived economic value of the
digital asset. It is the analog of the end-to-end principle in the
Internet architecture [SRC84], where contextual information is placed
at the endpoints of the transfer.
3.2. Operational Assumptions

The following conditions are assumed to have occurred, leading to the invocation of the asset transfer protocol between two gateways:

- Application layer transfer request: The transfer request from an originator in the origin network is assumed to have occurred prior to the execution of the asset transfer protocol by the gateways.

- Identification of originator and beneficiary: The originator and beneficiary are assumed to have been identified and that consent has been obtained from both parties regarding the asset transfer.

- Identification of origin and destination asset networks: The origin and destination networks is assumed to have been identified.

- Selection of gateway: The two corresponding gateways at the origin and destination networks is assumed to have been identified and selected.

- Identification of gateway-owners: The owners of the two corresponding gateways are assumed to have been identified and their ownership status verified.

4. Gateway Interoperability Modes

The current interoperability architecture based on gateways recognizes several types of transfer modes:

- Asset transfer: This refers to the transfer of a digital asset from the origin network to a destination network, where a successful asset transfer causes the asset to be extinguished in the origin network and be created (generated) at the destination network.

- Data transfer: This refers to the transfer of data only under authorization, in such a way that the data can be verified by a third party.

- Asset exchange (swap): This refers to the case where two users are present in two networks, and they perform concurrent and atomic swaps of two assets in the two corresponding networks, without transferring the assets outside the networks.

The data transfer mode addresses the use-cases where the state update in one network or system depends on the existence of state information recorded in a different network or system. A key use-
case is where a business workflow in one system depends on the advancement of a different workflow in another system, without requiring an asset transfer across the two systems. Here it is useful to distinguish data from asset, where the former can be copied in and across systems (with the authorization) and with the assurance of integrity and source-authenticity.

5. Architecture

5.1. Goal of Architecture

The goal of the interoperability architecture is to permit two (2) gateways belonging to distinct networks to conduct a transfer of digital assets transfer between them, in a secure, atomic and verifiable manner.

The asset as understood by the two gateway is expressed in an standard digital format in a way meaningful to the gateway syntactically and semantically.

The architecture recognizes that there are different networks currently in operation and evolving, and that in many cases the interior technical constructs in these networks maybe incompatible with one another.

The architecture therefore assumes that in addition to implementing the bilateral secure asset transfer protocol, a gateway has the role of making opaque (i.e. hiding) the constructs that are local and specific to its network.

Overall this approach ensures a high degree of interoperability across these networks, where each network can operate as a true autonomous system. Additionally, this approach permits each network to evolve its interior technology implementations without affecting other (external) networks.

The current architecture focuses on unidirectional asset transfers, although the building blocks in this architecture can be used to support protocols for bidirectional transfers.

For simplicity the current architecture employs two (2) gateways per transfer as the basic building block, with one gateway in the origin and destination networks respectively. However, the architecture seeks to be extensible to address future cases involving multiple gateways at both sides.
5.2. Overview of Asset Transfer

An asset transfer between two networks is performed using a secure asset transfer protocol implemented by the gateways in the respective networks. The two gateways implement the protocol in a direct interaction (unmediated).

A successful transfer results in the asset being extinguished (deleted) or marked on the origin network, and for the asset to be introduced (generated) into the destination network.

The secure asset transfer protocol provides a coordination between the two gateways through the various message flows in the protocol that is communicated over a secure channel.

The protocol implements a commitment mechanism between the two gateways to ensure that the relevant properties atomicity, consistency, isolation, and durability is achieved in the transfer.

The mechanism to extinguish or introduce an asset from/into a network by its gateway is dependent on the specific network and outside the scope of the current work.

As part of the commitment mechanism, the sender gateway in the origin network must deliver proof to the received gateway in the destination network that asset in question has been extinguished (deleted) in the origin network.

Similarly, the receiver gateway in the destination network must deliver proof to the sender gateway in the origin network that the asset has been generated in the destination network.

These two tasks must be performed in a synchronized fashion between the two gateways, and the commitment mechanism must provide evidence of the asset transfer that is verifiable by a third party.

5.3. Desirable Properties of Asset Transfer

The desirable features of asset transfers between two gateway include, but not limited, to the following:

- Atomicity: A transfer must either commit or entirely fail (failure means no change to asset state).
- Consistency: A transfer (commit or fail) always leaves the networks in a consistent state (i.e. the asset is located in one network only at any time).
- Isolation: While the transfer is occurring, the asset state cannot be modified in the origin network.

- Durability: Once a transfer has been committed by both gateways, it must remain so regardless of subsequent gateway crashes.

- Verifiable by authorized third parties: The proof that the asset has been extinguished in the origin network, and the proof that the asset has been generated in the destination network must be verifiable by an authorized third party.

An implementation of the asset transfer protocol should satisfy these properties, independent of whether the implementation employs stateful messaging or stateless messaging between the two gateways.

5.4. Event log-data, crash recovery and backup gateways

Implementations of a gateway should maintain event logs and checkpoints for the purpose of gateway crash recovery. The log-data generated by a gateway should be considered as an interior resource accessible to other authorized gateways within the same network.

The mechanism used to provide gateway crash-recovery is dependent on the specific network. For interoperability purposes the information contained in the log and the format of the log-data should be standardized.

The resumption of an interrupted transfer session (e.g. due to gateway crash, network failure, etc.) should take into consideration the aspects of secure channel establishment and the aspects of the transfer protocol resumption. In some cases, a new secure channel (e.g. TLS session) may need to be established between the two gateways, before a resumption of the transfer can begin.

The log-data collected by a gateway acts also as a checkpoint mechanism to assist the recovered (or backup) gateway in continuing the transfer. The point at which to re-start the transfer protocol flow is dependent on the implementation of the gateway recovery strategy.

5.5. Overview of the Phases in Asset Transfer

The interaction between two gateways in the secure asset transfer protocol is summarized in Figure 1, where the origin network is NW1 and the destination network is NW2. The gateways are denoted as G1 and G2 respectively.
The phases are summarized as follows.

- Phase 0: Initiation of transfer at the application layer. The two applications utilized by the originator and beneficiary is assumed to interact as part of the asset transfer. In this phase, the applications App1 and App2 may establish some context information (e.g., Session-ID) that will be made available to their respective gateways G1 and G2. The legal verification of the identities of the Originator and Beneficiary may occur in this phase [FATF]. This phase is outside the scope of the current architecture.

- Phase 1: Pre-transfer Verification of Asset and Identities. In this phase, the gateways G1 and G2 must perform mutual identification and authentication. Gateway G1 must communicate to G2 the type/information of the asset to be transferred, while G2 must validate that it has the ability to support this type of asset in its network.

- Phase 2: Evidence of asset locking or escrow. In this phase, gateway G1 must provide gateway G2 with sufficient evidence that the asset on its network NW1 is in a locked state (or escrowed) under the control of G1.)
o Phase 3: Transfer commitment. In this phase gateways G1 and G2 commit to the unidirectional asset transfer using a 3PC (3-phase commit) subprotocol.

These transfer phases will be further discussed below.

6. Pre-transfer Verification of Asset and Identities (Phase 1)

The purpose of the first phase is to verify the various information relating to the asset to be transferred. This may include, among others, the correct identities of the originator and beneficiary (as provided by the respective applications), the identity and legal status of the entities who own and operate the gateways, the type of the network, and network parameters, and the device-identities of the gateways.

Orig DB1 G1 G2 DB2 Benef
|--request------>
||||
--------------|--------------|--------------|--------------|
Phase 1
(1.1) <-----Owner id----->
(1.2) <--Asset Profile-->
(1.3) <--Orig/Benef id-->

Figure 2

This phase starts with the assumption that in network NW1 the gateway to process the asset transfer has been selected (namely gateway G1). It also assumes that the destination network NW2 has been identified where the beneficiary is located, and that gateway G2 in NW2 has been identified.

There are several steps that may occur in Phase 1:
o Secure channel establishment between G1 and G2: This includes the mutual verification of the gateway device identities and the exchange of the relevant parameters for secure channel establishment. In cases where device attestation [RATS] is required, the mutual attestation protocol must occur between G1 and G2 prior to proceeding to the next phase.

o Mutual device attestations: In cases where device attestation [RATS] is required, each gateway must yield attestation evidence to the other regarding its configuration. A gateway may take on the role as a attestation verifier, or it may rely on an external verifier to appraise the received evidence.

o Validation of the gateway ownership: There must be a means for gateway G1 and G2 to verify their respective ownerships (i.e. entities owning G1 and G2 respectively). Examples of ownership verification mechanism include X.509 certificates, directories of gateways and owners, and others.

o Validation of owner status: In some jurisdictions, limitations may be placed for regulated asset service providers to transact only with other similarly regulated service providers. Examples of mechanisms used to validate legal status of service providers include directories, Extended Validation (EV) X.509 certificates, and others.

o Identification and validation of type/asset profile: Both gateways must agree on the type of asset being transferred based on the published profile of the asset. Gateway G1 must communicate the asset-profile identification to gateway G2, who in turn must validate both the legal status of the asset as well as the technical capability of its network to accept the type of asset. The policies governing network NW2 with regards to permissible incoming assets must be enforced by G2.

o Exchange of Travel Rule information and validation: In jurisdictions where the Travel Rule policies regarding originator and beneficiary information is enforced [FATF], the owners of gateways G1 and G2 must comply to the Travel Rule. Mechanisms must be used to permit gateways G1 and G2 to make available originator/beneficiary information to one another in such a away that the Travel Rule information can be logged as part of the asset transfer history.

o Negotiation of asset transfer protocol parameters: Gateway G1 and G2 must agree on the parameters to be employed within the asset transfer protocol. Examples include endpoints definitions for
resources, type of commitment flows (e.g. 2PC or 3PC), lock-time durations, and others [SAT].

7. Evidence of asset locking or escrow (Phase 2)

The asset transfer protocol can commence when both gateways G1 and G2 have completed the verifications in Phase 1.

The steps of Phase 2 are summarized in Figure 4, and broadly consists of the following:

- **Commencement (2.1):** Gateway G1 indicates the start of the asset transfer protocol by sending a transfer-commence message to gateway G2. Among others, the message must include a cryptographic hash of the information agreed-upon in Phase 1 (e.g. asset profile, gateway identities, originator/beneficiary public keys, etc.).

- **Acknowledgement (2.2):** The gateway G2 must send an explicit acknowledgement of the receipt of the commence message, which should include a hash of commencement message (2.1) and other relevant session parameters.

- **G1 lock/escrow asset (2.3):** Gateway G1 proceeds to establish a lock or escrow the asset belonging to the originator. This prevents other local transactions in NW1 from changing the state of the asset until such time the lock by G1 is finalized or released. A time-lock or escrow may also be employed.

- **G2 logs incoming asset (2.4):** Gateway G2 logs a notification to its local state data DB2 that the arrival of the asset is imminent. This may also act as a notification for the beneficiary regarding incoming the asset transfer.

- **Lock Evidence (2.5):** Gateway G1 sends a digitally signed evidence regarding the locked (escrowed) state on the asset in network NW1. The signature by G1 is performed using its entity public-key pair. This signifies that G1 (i.e. its owner) is legally standing behind its assertion regarding the locked/escrowed state on the asset.

- **Evidence receipt (2.6):** If gateway G2 accepts the evidence, G2 then responds with a digitally signed receipt message which includes a hash of the previous lock-evidence message. Otherwise, if G2 declines the evidence then G2 can ignore the transfer and let it time-out (i.e. transfer failed). The signature by G2 is performed using its entity public-key pair.
Figure 3

The precise form of the evidence in step 2.5 is dependent on the type of network NW1, and must be previously agreed upon between G1 and G2 in Phase 1.

The purpose of this evidence is for dispute resolution between G1 and G2 (i.e. the entities who own and operate G1 and G2 respectively) in the case that asset state inconsistencies in NW1 and NW2 are discovered later.

The gateway G2 must return a digitally signed receipt to G1 of this evidence in order to cover G1 (exculpatory proof) in the case of later denial by G2.

8. Transfer Commitment (Phase 3)

In Phase 3 the gateways G1 and G2 finalizes to the asset transfer by performing a commitment protocol (e.g. 2PC or 3PC) as a process (sub-protocol) embedded within the overall asset transfer protocol.

Upon receiving the evidence-receipt message in the previous phase, G1 begins the commitment (see Figure 5):
Commit-prepare (3.1): Gateway G1 indicates to G2 to prepare for the commitment of the transfer. This message must include a hash of the previous messages (message 2.5 and 2.6).

Ack-prepare (3.2): Gateway G2 acknowledges the commit-prepare message.

Lock-final (3.3): Gateway G1 updates its local state data to indicate lock-finalization or escrow finalization on the asset in network NW1. This signals the permanent extinguishment of the asset from network NW1. This state data must include a hash reference to the lock transaction previously in step (2.3). This indicates that the asset is no longer associated with its previous owner (originator) and that the asset instance is no longer recognized within network NW1.

Commit-final (3.4): Gateway G1 indicates to G2 that G1 has performed a local lock/escrow finalization in NW1. This message must be digitally signed by G1.

Asset-create (3.5): Gateway G2 issues a local state update in network NW2 to create (re-generate) the asset, associated with the beneficiary. This transaction must include a hash of the previous message (3.4) and hash reference to the log-incoming data previously in step (2.4). These hash references connect the newly re-generated asset with the overall transfer event originating from gateway G1.

Ack-final (3.6): Gateway G2 indicates to gateway G1 that gateway G2 has performed an asset-regeneration in network NW2. This message must be digitally signed by G2.

Location-record (3.7): Gateway G1 has the option to record the locally the information provided by G2 in the previous step. This record should include a hash reference to the confirmed lock-finalization data from step 3.3. This information may aid in future search, audit and accountability purposes from a legal perspective.

Transfer complete (3.8): Gateway G1 must explicitly close the asset transfer session with gateway G2. This allows both sides to close down the secure channel established in Phase 1.
9. Commitment sub-protocol

Within Phase 2, the gateways must implement one (or more) transactional commitment sub-protocols that permit the coordination between two gateways, and the final commitment of the asset transfer.

In the case that there are multiple commitment subprotocols supported by the gateways, the choice of the sub-protocol (type/version) and the corresponding commitment evidence must be negotiated between the gateways during Phase 1.

For example, in Phase 2 and Phase 3 discussed above the gateways G1 and G2 may implement the classic 2 or 3 Phase Commit (2PC or 3PC) sub-protocol [Gray81] as a means to ensure efficient and non-disputable commitments to the asset transfer.

---

Historically, transactional commitment protocols employ locking mechanisms to prevent update conflicts on the data item in question. When used within the context of digital asset transfers across networks, the fact that an asset has been locked in NW1 must be communicated to G2 (as the 3PC participant) in an indisputable manner.

Similarly, G2 must return evidence to G1 that the asset has been re-generated in NW2.

These evidences must be verifiable by an authorized third party, in the case that disputes occur (post event) or where legal audit is required on the asset transfer.

The exact form of this evidence of asset-locking must be standardized (for the given transactional commitment protocol) to eliminate any ambiguity.

10. Security Considerations

As an asset network holds an increasing number of digital assets, it may become attractive to attackers seeking to compromise the cryptographic keys of the entities, services and its end-users.

Gateways are of particular interest to attackers because they enable the transferal of digital assets to external networks, which may or may not be regulated. As such, hardening technologies and tamper-resistant crypto-processors (e.g. TPM, SGX) should be used for implementations of gateways [HS19].

11. Policy Considerations

Digital asset transfers must be policy-driven in the sense that it must observe and enforce the policies defined for the network. Resources that make-up a network are owned and operated by entities (e.g. legal persons or organizations), and these entities typically operate within regulatory jurisdictions [FATF]. It is the responsibility of these entities to translate regulatory policies into functions on networks that comply to the relevant regulatory policies.

At the application layer, asset transfers must take into consideration the legal status of assets and incorporate relevant asset-related policies into their business logic. These policies must permeate down to the gateways that implement the functions of asset transaction processing.
12. References

12.1. Normative References


12.2. Informative References


Authors’ Addresses

Thomas Hardjono
MIT
Email: hardjono@mit.edu

Martin Hargreaves
Quant Network
Email: martin.hargreaves@quant.network

Ned Smith
Intel
Email: ned.smith@intel.com

Venkatraman Ramakrishna
IBM
Email: vramakr2@in.ibm.com
Secure Asset Transfer Protocol
draft-hargreaves-sat-core-00

Abstract

This memo describes the Secure Asset Transfer (SAT) Protocol for digital assets. SAT is a protocol operating between two gateways that conducts the transfer of a digital asset from one gateway to another. The protocol establishes a secure channel between the endpoints and implements a 2-phase commit to ensure the properties of transfer atomicity, consistency, isolation and durability.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 4, 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents.
Table of Contents

1.  Introduction .............................................. 3
2.  Conventions used in this document .......................... 4
3.  Terminology ................................................. 4
4.  The Open Digital Asset Protocol ............................. 4
   4.1.  Overview ............................................. 4
   4.2.  SAT Model ............................................. 5
   4.3.  Types of APIs ......................................... 5
   4.4.  Types of Flows ......................................... 6
   4.5.  Resources and Identifiers .............................. 6
5.  SAT Message Format, identifiers and Descriptors .......... 7
   5.1.  Overview ............................................. 7
   5.2.  SAT Message Format .................................... 7
   5.3.  Digital Asset Resource Descriptors .................... 8
      5.3.1.  Organization Identifier ............................ 8
      5.3.2.  Gateway / Endpoint ID ............................... 8
      5.3.3.  Network or system Identifier ....................... 8
      5.3.4.  Resource ........................................... 9
      5.3.5.  Examples ........................................... 9
   5.4.  Digital Asset Resource Client Descriptors ............. 9
      5.4.1.  Organization Identifier ............................ 9
      5.4.2.  Gateway / Endpoint ID ............................... 9
      5.4.3.  Organizational Unit ................................ 10
      5.4.4.  Name ............................................... 10
      5.4.5.  Examples ........................................... 10
   5.5.  Gateway Level Access Control ........................... 10
   5.6.  Negotiation of Security Protocols and Parameters ...... 11
      5.6.1.  TLS Established .................................... 11
      5.6.2.  Client offers supported credential schemes ........ 11
      5.6.3.  Server selects supported credential scheme ......... 11
      5.6.4.  Client asserts or proves identity .................. 11
      5.6.5.  Sequence numbers initialized ....................... 11
      5.6.6.  Messages can now be exchanged ...................... 12
   5.7.  Asset Profile Identification ........................... 12
   5.8.  Application Profile Negotiation ........................ 12
   5.9.  Discovery of Digital Asset Resources ................... 13
6.  Identity and Asset Verification Flow (Phase 0) ............. 13
7.  Transfer Initiation Flow (Phase 1) .......................... 14
   7.1.  Initialization Request Message ........................ 14
   7.2.  Initialization Request Message Response (ACK) .......... 16
8.  Lock-Evidence Verification Flow (Phase 2) ................... 17
1. Introduction

This memo proposes a secure asset transfer (SAT) protocol that is intended to be deployed between two gateway endpoints, where the protocol itself is agnostic to the backend system or network behind the respective gateway.

The SAT protocol is defined for a unidirectional transfer, with the understanding that it can be deployed as a building block for a bidirectional transfer or exchange of assets.

There are several desirable properties of the protocol. The protocol must ensure that non-repudiability of transfers is achieved, and that the classic properties of atomicity, consistency, isolation, and durability (ACID) must be satisfied.

The requirement of consistency implies that the asset transfer protocol always leaves both networks in a consistent state (that the asset is located in one system/network only at any time).

Atomicity means that the protocol must guarantee that either the transfer commits (completes) or entirely fails, where failure is taken to mean there is no change to the state of the asset in the origin (sender) network.

The property of isolation means that while a transfer is occurring to a digital asset from an origin network, no other state changes can occur to the asset.

The property of durability means that once the transfer has been committed by both gateways, that this commitment must hold regardless
of subsequent unavailability (e.g. crash) of the gateways implementing the SAT protocol.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying significance described in RFC 2119.

3. Terminology

The following are some terminology used in the current document:

Client application: This is the application employed by a user to interact with a gateway.

Gateway: The computer system functionally capable of acting as a gateway in an asset transfer.

Sender gateway: The gateway that initiates a unidirectional asset transfer.

Recipient gateway: The gateway that is the recipient side of a unidirectional asset transfer.

Claim: An assertion made by an Entity [JWT].

Claim Type: Syntax used for representing a Claim Value [JWT].

Gateway Claim: An assertion made by a Gateway regarding the status or condition of resources (e.g. assets, public keys, etc.) accessible to that gateway (e.g. within its network or system).

4. The Open Digital Asset Protocol

4.1. Overview

The Secure Asset Transfer Protocol (SAT) is a gateway-to-gateway protocol used by a sender gateway with a recipient gateway to perform a unidirectional transfer of a virtual asset.

The protocol defines a number of API endpoints, resources and identifier definitions, and message flows corresponding to the asset transfer between the two gateways.
4.2. SAT Model

The model for SAT is shown in Figure 1.

The Client (application) interacts with its local gateway (G1) over an interface (API Type-1) in order to provide instructions to the gateway with regards to actions to assets and related resources located in the local system (L1).

Gateways interact with each other over a gateway interface (API Type-2). A given gateway may be required to access resources that are not located in network L1 or network L2. Access to these types of resources are performed over an off-network interface (API Type-3).

4.3. Types of APIs

The following are the types of APIs in SAT:

- **Gateway APIs for client (API Type-1):** This the REST APIs that permit a Client (application) to interact with a local gateway, and issue instructions for actions pertaining to resources accessible to the gateway.

- **Gateway APIs for peer gateways (API Type-2):** This is the REST APIs employed by two (2) peer gateways in performing unidirectional asset transfers.
o APIs for validation of off-network resources (API Type-3): This is the REST APIs made available by a resource server (resource owner) at which a gateway can access resources.

The use of these APIs is dependent on the mode of access and the type of flow in question.

4.4. Types of Flows

The SAT protocol defines the following three (3) flows:

- **Transfer Initiation flow**: This flow deals with commencing a transfer from one gateway to another. Several tasks are involved, including (but not limited to): (i) gateway identification and mutual authentication; (ii) exchange of asset type (definition) information; (iii) verification of the asset definition, and others.

- **Lock-Evidence flow**: This flow deals with the conveyance of evidence regarding the lock status (e.g. escrow) 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.

These flow will be discussed below.

4.5. Resources and Identifiers

- (a) Resource addressing for systems or networks, using the URL syntax.

- (b) 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.

- (c) Protocol message family for negotiating authentication, authorisation, and parameters for confidential channel establishment.

- (d) Resource discovery mechanism for developers and applications to discover resources hosted at a gateway. The gateway response is subject to the level of access granted to that developer or application.
5. SAT Message Format, identifiers and Descriptors

5.1. Overview

This section describes (i) the phases of the SAT protocol; (ii) the format of SAT messages; (iii) the format for resource descriptors; (iv) a method for gateways to implement access controls; (iv) protocol for negotiating security capabilities; (v) discovery and accessing resources and provisions for backward compatibility with existing systems.

5.2. SAT Message Format

SAT messages are exchanged between applications (clients) and gateways (servers). They consist of protocol negotiation and functional messages.

Messages are in JSON format, with protocol specific mandatory fields, support for several authentication and authorization schemes and support for a free format field for plaintext or encrypted payloads directed at the gateway.

JSON format message, mandatory fields are shown below:

- Version: SAT protocol Version (major, minor).
- Session ID: unique identifier (UUIDv2) representing a session
- Sequence Number: monotonically increasing counter that uniquely represents a message from a session.
- SAT Phase: The current SAT phase.
- Resource URL: Location of Resource to be accessed.
- Developer URN: Assertion of developer / application identity.
- Action/Response: GET/POST and arguments (or Response Code)
- Credential Profile: Specify type of auth (e.g. SAML, OAuth, X.509)
- Credential Block: Credential token, certificate, string
- Payload Profile: Asset profile and capabilities
- Application Profile: Vendor or Application specific profile
5.3. Digital Asset Resource Descriptors

Resources are identified by URL [RFC 1738] as described below:

- The type is new: application/satres
- The access protocol is SAT.

Data included in the URL includes the following:

5.3.1. Organization Identifier

This MAY be a Legal Entity Identifier (LEI) or other identifier linking resource ownership to a real world entity. Any scheme for identifying gateway owners may be implemented (e.g. LEI directory, closed user group membership, SWIFT BIC, etc.).

The developer or application MAY validate the identity with the issuing authority. The identifier is not a trusted identity, but MAY be relied on where trust has been established between the two parties (e.g. in a closed user group).

The mechanisms to determine organizations identifiers is out of scope for the current specification.

5.3.2. Gateway / Endpoint ID

FQDN of the SAT compliant gateway. Required to establish IP connectivity. This MUST resolve to a valid IP address.

5.3.3. Network or system Identifier

Specific to the gateway behind which the target network operates. This field is local to the gateway and is used to direct SAT interactions to the correct underlying network.

For example: "tradelens-network", "EU-supply-chain".
5.3.4. Resource

Specifies a resource held on the underlying network. This field must be meaningful to the network in question but is otherwise an arbitrary string. The underlying object it points to may be a network address, data block, transaction ID, alias, etc. or a future object type not yet defined.

5.3.5. Examples

satres://quant/api.gateway1.com/swift

5.4. Digital Asset Resource Client Descriptors

Resources are identified by URN as described below:

- The type is new: application/satclient

The URN format does not imply availability or access protocol.

Data included in the URN includes the following:

5.4.1. Organization Identifier

Legal Entity Identifier (LEI) or other identifier linking resource ownership to a real-world entity. Any scheme for identifying Gateway owners may be implemented (e.g. LEI directory, closed user group membership, BIC, etc.).

The Gateway MAY validate the identity with the issuing authority. The identifier is not a trusted identity, but MAY be relied on where trust has been established between the two parties (e.g. in a closed user group).

5.4.2. Gateway / Endpoint ID

Applications which interact with multiple networks can operate in a mode whereby the application connects to its local gateway, which then forwards application traffic to local networks and to remote networks via other SAT gateways.

Where this is the case, this field identifies the "home" gateway for this application. This may be required to carry out gateway to gateway handshaking and protocol negotiation, or for the server to look up use case specific data relating to the client.
5.4.3. Organizational Unit

The organization unit within the organization that the client (application or developer) belongs to. This assertion should be backed up with authentication via the negotiated protocol.

The purpose of this field is to allow gateways to maintain access control mapping between applications and resources that are independent of the authentication and authorization schemes used, supporting future changes and supporting counterparties that operate different schemes.

5.4.4. Name

A locally unique (within the OU) identifier, which can identify the application, project or individual developer responsible for this client connection. This is the most granular unit of access control, and gateways should ensure appropriate identifiers are used for the needs of the application or use case.

5.4.5. Examples

satclient:quant/api.overledger.quant.com/research/luke.riley

5.5. Gateway Level Access Control

Gateways can enforce access rules based on standard naming conventions using novel or existing mechanisms such as AuthZ protocols using the resource identifiers above, for example:

satclient://hsbc/api.overledger.hsbc.com/lending/eric.devloper

can READ/WRITE

satres://quant/api.gateway1.com/tradelens

AND

satres://quant/api.gateway1.com/ripple

These rules would allow a client so identified to access resources directly, for example:

satres://quant/api.gateway1.com/tradelens/xxxxxADDRESSxxxxx

This method allows resource owners to easily grant access to individuals, groups and organizations. Individual gateway implementations may implement access controls, including subsetting...
and supersetting or applications or resources according to their own requirements.

5.6. Negotiation of Security Protocols and Parameters

5.6.1. TLS Established

TLS 1.2 or higher MUST be implemented to protect gateway communications. TLS 1.3 or higher SHOULD be implemented where both gateways support TLS 1.3 or higher.

5.6.2. Client offers supported credential schemes

Capability negotiation prior to data exchange, follows a scheme similar to the Session Description Protocol [RFC 5939]. Initially the client (application) sends a JSON block containing acceptable credential schemes, e.g. OAuth2.0, SAML in the "Credential Scheme" field of the SAT message.

5.6.3. Server selects supported credential scheme

The server (recipient Gateway) selects one acceptable credential scheme from the offered schemes, returning the selection in the "Credential Scheme" field of the SAT message.

If no acceptable credential scheme was offered, an HTTP 511 "Network Authentication Required" error is returned in the Action/Response field of the SAT message.

5.6.4. Client asserts or proves identity

The details of the assertion / verification step are specific to the chosen credential scheme and are out of scope of this document.

5.6.5. Sequence numbers initialized

Sequence numbers are used to allow the server to correctly order operations from the client, some of which may be asynchronous, synchronous, idempotent with duplicate requests handled in different ways according to the use case.

The initial sequence number is proposed by the client (sender gateway) after the finalization of credential verification. The server (recipient gateway) MUST respond with the same sequence number to indicate acceptance.
The client (sender gateway) increments the sequence number with each new request. Sequence numbers can be reused for retries in the event of a gateway timeout.

5.6.6. Messages can now be exchanged

Handshaking is complete at this point, and the client can send SAT messages to perform actions on resources, which MAY reference the SAT Payload field.

5.7. Asset Profile Identification

The client and server must mutually agree as to the asset type or profile that is the subject to the current transfer from the client and server. The client must provide the server with the asset-identification number, or the server may provide the client with the asset-identification numbers for the digital asset supported by the server.

Formal specification of asset identification is out of scope of this document. Global numbering of digital asset types or profiles is expected to be performed by a legally recognized entity.

5.8. Application Profile Negotiation

Where an application relies on specific extensions for operation, these can be represented in an Application Profile.

For example, a payments application tracks payments through the use of a cloud based API and will only interact with gateways that log messages to that API, a resource profile can be established:

Application Name: TRACKER

X-Tracker_URL: https://api.tracker.com/updates

X-Tracking-Policy: Always

As gateways implement this functionality, they support the TRACKER application profile, and the application is able to expand its reach by periodically polling for the availability of the profile.

This is an intentionally generalized extension mechanism for application or vendor specific functionality.
5.9. Discovery of Digital Asset Resources

Applications located outside a network or system SHOULD be able to discover which resources they are authorized to access in a network or system.

Resource discovery is handled by the gateway in front of the network. For instance using a GET request against the gateway URL with no resource identifier could return a list of URLs available to the requester. This list is subject to the access controls above.

Gateways MAY allow applications to discover resources they do not have access to. This should be indicated in the free text field, and gateways SHOULD implement a process for applications to request access.

Formal specification of supported resource discovery methods is out of scope of this document.

6. Identity and Asset Verification Flow (Phase 0)

Prior to commencing the asset transfer from the sender gateway (client) to the recipient gateway (server), both gateways must perform a number of verifications steps. The types of information required by both the sender and recipient are use-case dependent and asset-type dependent.

The verifications include, but not limited to, the following:

- Gateway identity mutual verification: This is the identity of the gateway at the protocol and network layer. This may include the public-keys of the gateways.
- Gateway owner verification: This is the verification of the identity (e.g. LEI) of the owners of the gateways.
- Gateway device and state validation: This is the device attestation evidence [RATS] that a gateway must collect and convey to each other, where a verifier is assumed to be available to decode, parse and appraise the evidence.
- Originator and beneficiary identity verification: This is the identity and public-key of the entity (originator) in the origin network seeking to transfer the asset to another entity (beneficiary) in the destination network.
These are considered out of scope in the current specifications, and are assumed to have been successfully completed prior to the commencement of the transfer initiation flow.

7. Transfer Initiation Flow (Phase 1)

This section describes the SAT initialization phase, where a sender gateway interacts with a recipient gateway, proposing a session.

For this, several artifacts need to be validated: asset profile, asset ownership evidence, identities, and logging-related operations (log profile, access control profile).

In this phase, gateways implement the Transfer Initiation Flow endpoint.

In the following, the sender gateway takes instructions from an application, while the recipient gateway may act on behalf of its applications.

The flow follows a request-response model. The sender gateway makes a request (POST) to the Transfer Initiation endpoint at the recipient gateway.

Gateways MUST support the use of the HTTP GET and POST methods defined in RFC 2616 [RFC2616] for the endpoint.

Clients (sender gateway) MAY use the HTTP GET or POST methods to send messages in this phase to the server (recipient gateway). If using the HTTP GET method, the request parameters may be serialized using URI Query String Serialization.

The client and server may be required to sign certain messages in order to provide standalone proof (for non-repudiation) independent of the secure channel between the client and server. This proof may be required for audit verifications (e.g. post-event).

(NOTE: Flows occur over TLS. Nonces are not shown).

7.1. Initialization Request Message

This message is sent from the sender gateway (client) to the recipient gateway (server).

The purpose of this message is for the client to initiate an asset transfer. Depending on the proposal, multiple rounds of communication between the client and the server may happen.
The parameters of this message consists of the following:

- **Version**: SAT 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
- **Sender_gateway_pubkey** REQUIRED: the public key of the gateway initiating a transfer
- **Sender_gateway_network_id** REQUIRED: the ID of the source network
- **Recipient_gateway_pubkey** REQUIRED: the public key of the gateway involved in a transfer
- **Recipient_gateway_network_id** REQUIRED: the ID of the recipient network
- **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 (public-keys or X.509 certificates) that can perform operations on the escrow or lock on the asset in the origin network.
o Origin: along with the sender gateway identification, allows identifying from where are the asset is escrowed/provided

o Destination: along with the recipient gateway identification, allows identifying to where are the escrowed asset is going

o Subsequent calls: details possible escrow actions

o History (optional): provides an history of the escrow, in case it has previously been initialized.

The sender gateway makes the following HTTP request using TLS.

Example: TBD.

7.2. Initialization Request Message Response (ACK)

After receiving an Initialization Request Message, the recipient gateway needs to validate the profiles.

This validation could be performed automatically (using a defined set of rules), or by requiring approval by an application.

If one of the profiles is rejected, the recipient gateway constructs a Initialization Denied Message, stating what was rejected, and proposing an alternative (if applicable).

Otherwise, if approved, the recipient gateway constructs an Initialization Request Message Response.

The purpose of this message is for the server (recipient gateway) to indicate agreement to proceed with the proposed operations, under the proposed profiles.

This message is sent from the recipient gateway to the sender gateway (client) in response to a Initialization Request from the sender gateway.

The message must be signed by the server.

The parameters of this message consists of the following:

o Session ID: unique identifier (UUIDv2) representing a session.

o Sequence Number: monotonically increasing counter that uniquely represents a message from a session.

o SAT Phase: The current SAT phase.
8. Lock-Evidence Verification Flow (Phase 2)

This section describes the conveyance of claims regarding the status of the asset (or resource) from a sender gateway to a recipient gateway.

In this phase, the gateways implement the Lock-Evidence Agreement endpoint.

In the following, the sender gateway takes the role of the client while the recipient gateway takes the role of the server.

The flow follows a request-response model. The client makes a request (POST) to the Lock-Evidence Agreement endpoint at the server.

Gateways MUST support the use of the HTTP GET and POST methods defined in RFC 2616 [RFC2616] for the endpoint.

Clients MAY use the HTTP GET or POST methods to send messages in this phase to the server. If using the HTTP GET method, the request parameters may be serialized using URI Query String Serialization.

The client and server may be required to sign certain messages in order to provide standalone proof (for non-repudiation) independent of the secure channel between the client and server. This proof may be required for audit verifications post-event.

(NOTE: Flows occur over TLS. Nonces are not shown).

8.1. Transfer Commence Message

This message is sent from the client (sender gateway) to the Transfer Request Endpoint at the server (recipient gateway). It signals to
the server that the client is ready to start the transfer of the digital asset.

The message must contain claims related to the information from the previous flow (Phase 1). It must be signed by the client.

The parameters of this message consists of the following:

- message_type REQUIRED. MUST be the value urn:ietf:sat:msgtype:transfer-commence-msg
- originator_pubkey REQUIRED. This is the public key of the asset owner (originator) in the origin network or system.
- beneficiary_pubkey REQUIRED. This is the public key of the beneficiary in the destination network.
- sender_gateway_network_id REQUIRED. This is the identifier of the origin network or system behind the client.
- recipient_gateway_network_id REQUIRED. This is the identifier of the destination network or system behind the server.
- client_identity_pubkey REQUIRED. The public key of client who sent this message.
- server_identity_pubkey REQUIRED. The public key of server for whom this message is intended.
- hash_asset_profile REQUIRED. This is the hash of the asset profile previously agreed upon in Phase 1.
- asset_unit OPTIONAL. If applicable this is the unit amount of the asset being transferred, previously agreed upon.
- hash_prev_message REQUIRED. The hash of the last message in Phase 1.
- client_transfer_number OPTIONAL. This is the transfer identification number chosen by the client. This number is meaningful only the client.
- client_signature REQUIRED. The digital signature of the client.

For example, the client makes the following HTTP request using TLS (with extra line breaks for display purposes only):
POST /token HTTP/1.1
Host: server.example.com
Authorization: Basic awHCaGRSa3F0MzpWDFmQmF0M2ZG
Content-Type: application/x-www-form-urlencoded

{
    "message_type": "urn:ietf:sat:msgtype:transfer-commence-msg",
    "originator_pubkey": "zGy89097hkbfgkjkvVbNH",
    "beneficiary_pubkey": "mBGHJjjuijh67yghb",
    "sender_net_system": "originNETsystem",
    "recipient_net_system": "recipientNETsystem",
    "client_identity_pubkey": "fgH654tgeryuryuy",
    "server_identity_pubkey": "dFgdfgdgt43tetr55teyrfge4t54334",
    "hash_asset_profile": "nbvcwertyhgfdsertyhgf2h3v4bd3v21",
    "asset_unit": "ghytredcfvhfhr",
    "hash_prev_message": "DRVr654vgreDerverv654nhRbver4",
    "client_transfer_number": "ji9876543ewdfgh",
    "client_signature": "fdw34567uyhgfger45"
}

Figure 2

8.2. Transfer Commence Response Message (Ack)

The purpose of this message is for the server to indicate agreement to proceed with the asset transfer.

This message is sent from the server (recipient gateway) to client (sender gateway) in response to a Transfer Commence Request from the client.

The message must be signed by the server.

The parameters of this message consists of the following:

- message_type REQUIRED urn:ietf:sat:msgtype:transfer-commence-ack-msg
- client_identity_pubkey REQUIRED. The client for whom this message is intended.
- server_identity_pubkey REQUIRED. The server who sent this message.
- hash_commence_request REQUIRED. The hash of previous message.
o server_transfer_number OPTIONAL. This is the transfer identification number chosen by the server. This number is meaningful only to the server.

o server_signature REQUIRED. The digital signature of the server.

An example of a success response could be as follows:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache

{
  "message_type": "urn:ietf:sat:msgtype:transfer-commenceack-msg",
  "client_identity_pubkey": "fgH654tgeryuryuy",
  "server_identity_pubkey": "dFgdfgdgt43tetrt535teyrfge4t54334",
  "hash_commence_request": "DRvfrb654vgreDerverv654nhRbvder4",
  "server_transfer_number": "ji9876543ewdfgh",
  "server_signature": "aaw34567uyhgfer66"
}
```

Figure 3

8.3. Lock Evidence Message

The purpose of this message is for the client (sender gateway) to deliver the relevant asset locking (or escrow) evidence to the server (recipient gateway).

The format of the evidence is dependent on the network or system fronted by the client and is outside the scope of this specification.

This message is sent from the client to the Evidence Validation Endpoint at the server.

The server must validate the lock evidence claims (payload) in this message prior to the next step.

The message must be signed by the client (sender gateway).

The parameters of this message consist of the following:

o message_type REQUIRED urn:ietf:sat:msgtype:lock-evidence-req-msg
o client_identity_pubkey REQUIRED. The client who sent this message.

o server_identity_pubkey REQUIRED. The server for whom this message is intended.

o lock_evidence_claim REQUIRED. The lock or escrow evidence (on the network behind the client).

o lock_claim_format OPTIONAL. The format of the evidence.

o lock_evidence_expiration REQUIRED. The duration of time of lock on the asset (after which the lock is released).

o hash_commence_ack_request REQUIRED. The hash of the previous message.

o client_transfer_number OPTIONAL. This is the transfer identification number chosen by the client. This number is meaningful only to the client.

o client_signature REQUIRED. The digital signature of the client.

8.4. Lock Evidence Response Message (Ack)

The purpose of this message is for the transfer server (recipient gateway) to indicate acceptance of the asset-escrow (or lock) evidence delivered by the client (sender gateway) in the previous message.

The message must be signed by the server.

The parameters of this message consists of the following:

o message_type REQUIRED urn:ietf:sat:msgtype:lock-evidence-ack-msg

o client_identity_pubkey REQUIRED. The client for whom this message is intended.

o server_identity_pubkey REQUIRED. The server who sent this message.

o hash_lockevidence_request REQUIRED. The hash of previous message.

o server_transfer_number OPTIONAL. This is the transfer identification number chosen by the server. This number is meaningful only to the server.
9. Commitment Establishment Flow (Phase 3)

This section describes the transfer commitment agreement between the sender gateway to a recipient gateway.

This phase must be completed within the asset-lock duration time specified in the previous lock_evidence_expiration parameter.

In this phase gateways implement the Transfer Commitment endpoint.

In the following, the sender gateway takes the role of the client while the recipient gateway takes the role of the server.

The flow follows a request-response model. The client makes a request (POST) to the Transfer Commitment endpoint at the server.

Gateways MUST support the use of the HTTP GET and POST methods defined in RFC 2616 [RFC2616] for the endpoint.

Clients MAY use the HTTP GET or POST methods to send messages in this phase to the server. If using the HTTP GET method, the request parameters maybe serialized using URI Query String Serialization.

The client and server may be required to sign certain messages in order to provide standalone proof (for non-repudiation) independent of the secure channel between the client and server. This proof maybe required for audit verifications post-event.

(NOTE: Flows occur over TLS. Nonces are not shown).

9.1. Commit Preparation Message

The purpose of this message is for the client to indicate its readiness to begin the commitment of the transfer.

The message must be signed by the client.

The parameters of this message consists of the following:

- message_type REQUIRED. It MUST be the value urn:ietf:sat:msgtype:commit-prepare-msg
- client_identity_pubkey REQUIRED. The client who sent this message.
9.2. Commit Preparation Response

The purpose of this message is for the server to indicate to the client its readiness to proceed with the commitment finalization step.

The message must be signed by the server.

The parameters of this message consists of the following:

- message_type REQUIRED. It MUST be the value urn:ietf:sat:msgtype:commit-prepare-ack-msg
- client_identity_pubkey REQUIRED. The client for whom this message is intended.
- server_identity_pubkey REQUIRED. The server who sent this message.
- hash_commitprep REQUIRED. The hash of previous commit preparation message.
- server_transfer_number OPTIONAL. This is the transfer identification number chosen by the server. This number is meaningful only the server.
- server_signature REQUIRED. The digital signature of the server.

9.3. Commit Final Message

The purpose of this message is for the client to indicate to the server that the client (sender gateway) has completed local extinguishment of the asset on its network or system (L1), and that now on its part the server (recipient gateway) must re-generate the asset on its network or system (L2).
The message must contain standalone claims related to the extinguishment of the asset by the client. The claim must be signed by the client.

The parameters of this message consists of the following:

- message_type REQUIRED. It MUST be the value urn:ietf:sat:msgtype:commit-final-msg
- client_identity_pubkey REQUIRED. The client who sent this message.
- server_identity_pubkey REQUIRED. The server for whom this message is intended.
- commit_final_claim REQUIRED. This is one or more claims signed by the client that the asset in question has been extinguished by the client in its local network or system.
- commit_final_claim_format OPTIONAL. This is the format of the claim provided by the client in this message.
- hash_commitprepare_ack REQUIRED. The hash of previous message.
- client_transfer_number OPTIONAL. This is the transfer identification number chosen by the client. This number is meaningful only the client.
- client_signature REQUIRED. The digital signature of the client.

### 9.4. Commit Final Response Message

The purpose of this message is for the server to indicate to the client that the server has completed the asset re-generation at its network or system (L2).

The message must contain standalone claims related to the re-generated of the asset by the server. It must be signed by the server.

The parameters of this message consists of the following:

- message_type REQUIRED. It MUST be the value urn:ietf:sat:msgtype:commit-final-ack-msg
- client_identity_pubkey REQUIRED. The client for whom this message is intended.
- server_identity_pubkey REQUIRED. The server who sent this message.
- commit_acknowledgement_claim REQUIRED. This is one or more claims signed by the server that the asset in question has been regenerated by the server in its local network or system.
- commit_acknowledgement_claim_format OPTIONAL. This is the format of the claim provided by the server in this message.
- hash_commitfinal REQUIRED. The hash of previous commit final message.
- server_transfer_number OPTIONAL. This is the transfer identification number chosen by the server. This number is meaningful only the server.
- server_signature REQUIRED. The digital signature of the server.

9.5. Transfer Complete Message

The purpose of this message is for the client to indicate to the server that the asset transfer has been completed and that no further messages are to be expected from the client in regards to this transfer instance.

The message logically closes the first message of Phase 2 (Transfer Commence Message). It must be signed by the client.

The parameters of this message consists of the following:
- message_type REQUIRED. It MUST be the value urn:ietf:sat:msgtype:commit-transfer-complete-msg
- client_identity_pubkey REQUIRED. The client who sent this message.
- server_identity_pubkey REQUIRED. The server for whom this message is intended.
- hash_commit_final_ack REQUIRED. The hash of previous message.
- hash_transfer_commence REQUIRED. The hash of the Transfer Commence message at the start of Phase 2.
- client_transfer_number OPTIONAL. This is the transfer identification number chosen by the client. This number is meaningful only the client.
client_signature REQUIRED. The digital signature of the client.

10. Security Consideration

Gateways are of particular interest to attackers because they are a kind of end-to-end pipeline that enable the transferral of digital assets to external networks or systems. Thus, attacking a gateway may be attractive to attackers instead of the network behind a gateway.

As such, hardware hardening technologies and tamper-resistant crypto-processors (e.g. TPM, Secure Enclaves, SGX) should be considered for implementations of gateways.

11. IANA Consideration

(TBD)

12. References

12.1. Normative References


12.2. Informative References


Authors’ Addresses

Martin Hargreaves
Quant Network

Email: martin.hargreaves@quant.network

Thomas Hardjono
MIT

Email: hardjono@mit.edu

Rafael Belchior
Technico Lisboa

Email: rafael.belchior@tecnico.ulisboa.pt