MEMBERSHIP SERVICES FOR HYPERLEDGER

JUNE 23, 2016

GUARDED BY GENIUS

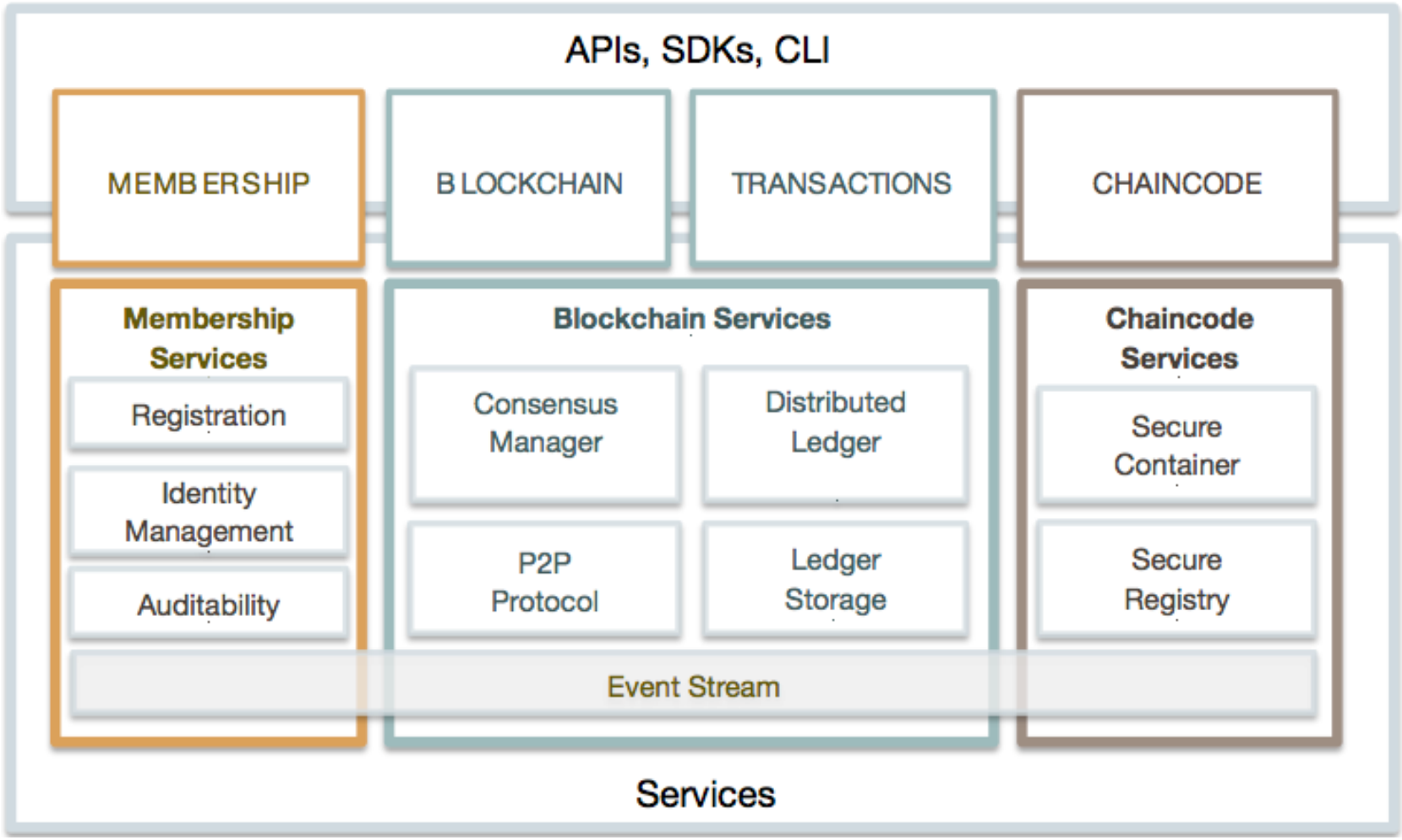
CONTENTS

01

Introduction

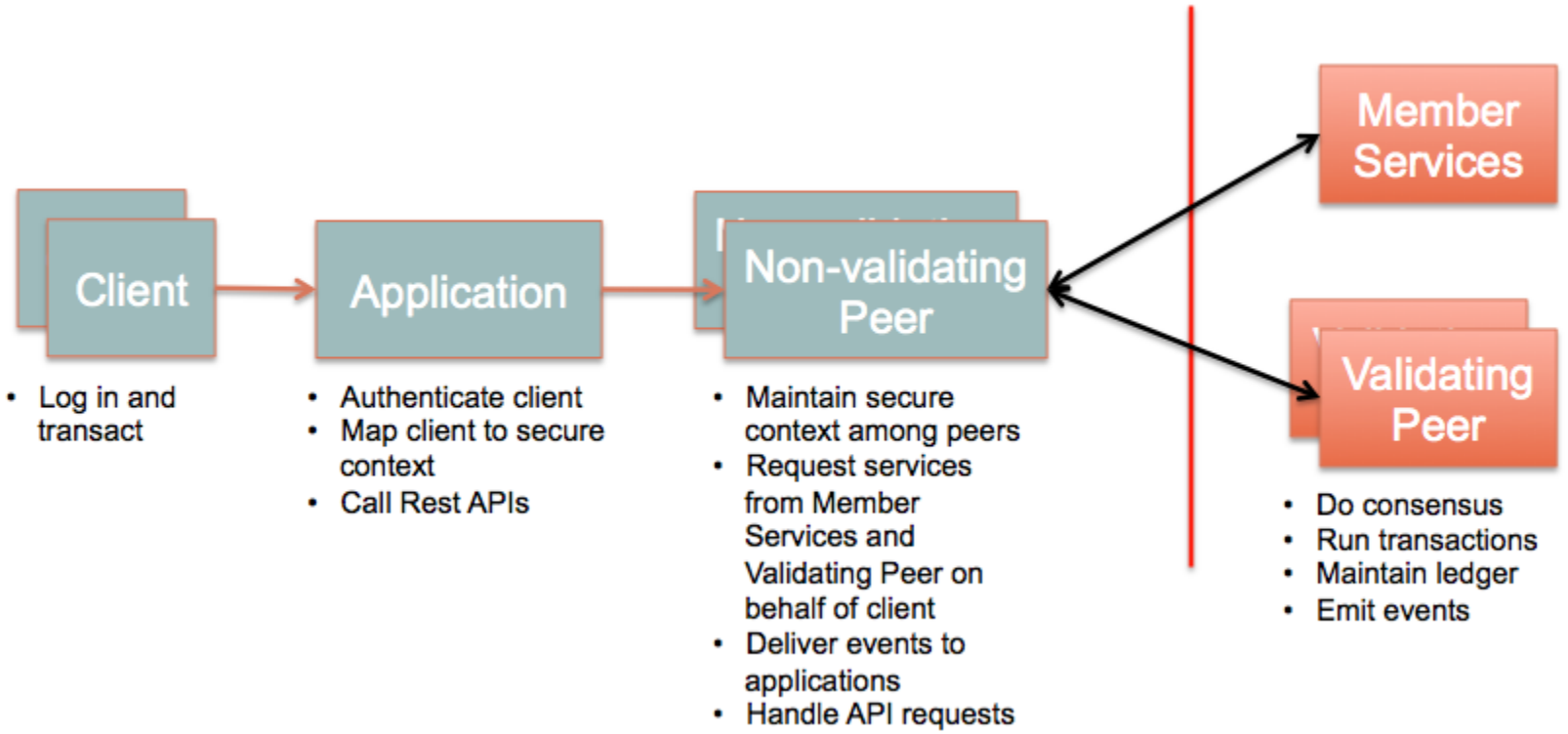
HIGH-LEVEL HYPERLEDGER FABRIC ARCHITECTURE

From: https://github.com/hyperledger/fabric/blob/master/docs/protocol-spec.md



HIGH-LEVEL HYPERLEDGER FABRIC TOPOLOGY

From: https://github.com/hyperledger/fabric/blob/master/docs/protocol-spec.md



MEMBERSHIP SERVICES: CAPABILITIES OF IDENTITY AND ATTRIBUTE MANAGEMENT

- Enables pseudonymous systems (unlike anonymous Bitcoin)
- Secure import of User properties:
  - Identities, affiliations, attributes (or entitlements/authorizations/licenses/accounts…)
- Granting read-access: Properties of transaction-creating User are selectively revealed (possibly in addition to other data) on a transaction-specific basis to intended transaction participants
- Granting write-access: Intended transaction participants are authorized to follow up, e.g., to retrieve/retransfer a transferred asset and/or utilize deployed chain-code
- Auditability (e.g., for regulatory compliance)
  - Group/cluster transactions according to Users
  - Can only access info of Users affiliated to auditor's jurisdiction

PUBLIC KEY INFRASTRUCTURE (PKI) AND PRIVILEGE MANAGEMENT INFRASTRUCTURE (PMI)

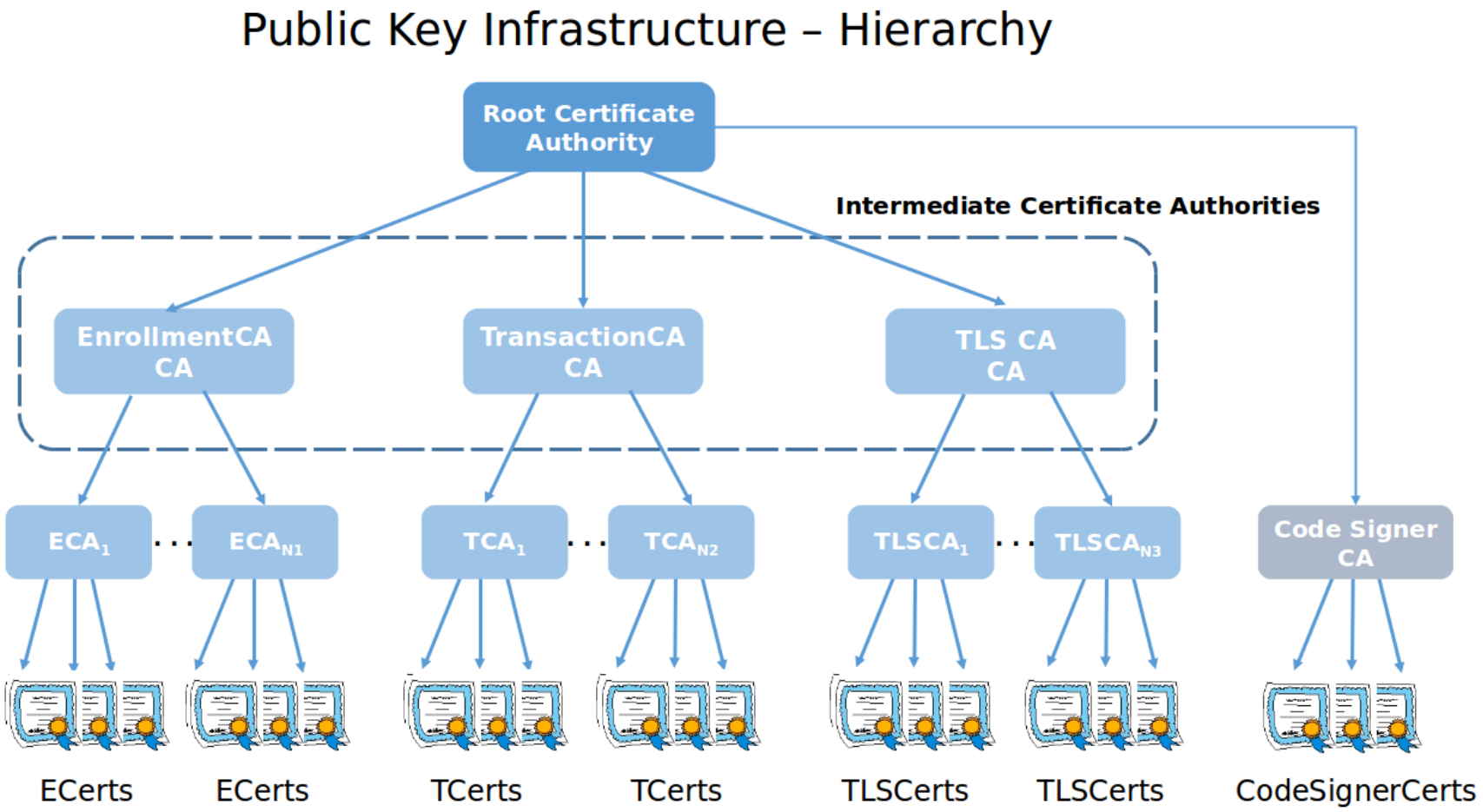
- Reconciling transaction privacy with identity management
  - Establishing a "permissioned" blockchain implies adding certificates to transactions
- We use long-term enrollment certificates – per standard practice

But the standard attributes mechanism for access control does not work: We can't use attribute certificates/authorization certificates that point to the User's enrollment certificate – because that would expose the common source of multiple transactions

- Enrollment Certificates (ECerts)
  - relatively static (long-term) certificates with UserID and Affiliation
  - acquired via registration with an Enrollment Certificate Authority (ECA) through its Registration Authority (RA) function
- Transaction Certificates (TCerts)
  - short-term certificates that have a behind-the-scenes dependency on the User's Enrollment Certificate
  - faithfully but pseudonymously represent enrolled Users
  - acquired via a Transaction Certificate Authority (TCA)
  - handle attributes that may be more dynamic in nature

PKI WITH PMI ELEMENTS

From: https://github.com/hyperledger/fabric/blob/master/docs/protocol-spec.md



02

Interface with Internal and External Identity- and Attribute- Service Providers

STRENGTHEN THE PMI: ADD AN INTERNAL ATTRIBUTE CERTIFICATE AUTHORITY (ACA)

- The ACA interfaces with internal/external Attribute Authorities (AA) indirectly via a User Agent that enables the User to acquire verifiably fresh evidence of attribute ownership
    - Such evidence gets amalgamated into an encrypted database accessible by the TCA for establishing the legitimacy of requests for TCerts that include specified attributes
- This is analogous to the internal Registration Authority (RA) that interfaces indirectly with internal/external Identity Providers (IdP) in order to apprise the ECA of legitimate requests for Enrollment Certificates
- Use can be made of standardized methods such as mutually authenticated TLS, X.509 self-signed (User Agent) client certificates, and SAML holder-of-key assertions
- The combined Public Key Infrastructure (PKI)/Privilege Management Infrastructure (PMI) system may deploy intra-chain, inter-chain, cross-certification, multi-signature, and/or stealth multi-signature (i.e., threshold signature) elements for efficient decentralization that preserves interoperability with legacy/external systems in order to inherit/establish and maintain trust

03

More about Goals of Membership Services

ROLE-BASED ACCESS CONTROL

Authorized auditors have automatic access (long-term keys)

This User was granted limited-(i.e., transaction-specific-) access by transaction creator

WHAT DOES MEMBERSHIP SERVICES ENABLE?

- Users (reflexive transaction identification)
  - Check if a specific transaction belongs to that particular User
  - List pointers to all transactions of that particular User
- Auditors
  - Identify all transactions from Users of a particular grouping or affiliation – as long as auditor is authorized to see information about such affiliation

MEMBERSHIP SERVICES OBJECTIVES ACHIEVED

1. Leverage existing/legacy identity management systems and credentials via import at the RA-ECA/ACA-TCA edge of the permissioned blockchain system
2. Ensure proper handling of proprietary corporate data and preserve individual privacy, without sacrificing auditability or regulatory compliance
3. Role-based access control is efficiently achieved – asset transfers can be done just by knowing the unique name/identifier of intended recipient – no need for out-of-band exchange of cryptographic objects/addresses
4. Permission to see (one or more of) my attributes is offered by naming you or a group you belong to (by one of your attributes, such as UserID or the group identifier) within the transaction

04

Structure and Utilization of TCerts

STRUCTURE OF TCERTS: HIDDEN ATTRIBUTES

- Encryption of attributes using symmetric keys with the following properties:
  - Different per TCert (to ensure unlinkability, which encryption with a reused key would not)
  - Different per attribute (to enable User-initiated selective release, so as to make the acquisition of TCerts and their use in transactions efficiently asynchronous)
  - Independently accessible to only those auditors whose (hierarchical) authorization includes the Affiliation of that User as extracted by the TCA from an Enrollment Certificate used to authenticate the request for issuance of the TCert
- Mandatory attributes include (primary) Affiliation and UserID [mandatory for inclusion within TCerts for auditability, but not necessarily released during transactions]
- The TCert-specific symmetric key from which the attribute-specific symmetric keys above are derived is securely provided to the User with the TCert. As an aid in resilience against device crash, the key need no longer be retained once the TCert is used in a transaction

STRUCTURE OF TCERTS: HIDDEN MAPPING OF SHORT-TERM KEYS TO LONG-TERM KEY

- Encryption of a TCert-unique parameter (TCertIndex) that is used by the TCA for construction of the TCert public key from the ECert's public key and for recovery by the specific User of the TCert private key from the private key corresponding to the ECert's public key
  - The TCert private key is used to sign a transaction that is permissioned via the TCert
  - The TCert public key is used to verify the authenticity and integrity of the transaction
  - Enables bandwidth- and computationally- efficient request and production of TCerts
  - Enables portable security across User's devices without retention/synchronization of TCert-specific data, as optimized for blockchain-based access to unencrypted TCerts within past transactions
    - Conducive to secure instantiation of complex chained/sequenced transactions

STRUCTURE OF TCERTS: REVOCATION; KEY AGREEMENT

- A revocation field can be included within the TCert to enable efficient management of certificate revocation lists (CRL). If a particular batch of TCerts is revoked by including a single entry within the CRL, then only the TCerts of that batch become linkable as pertaining to the same User. TCerts of all previous and future batches remain unaffected
- Incorporate a key agreement public key (in place of signature verification TCert public key) into a streamlined variant of TCerts. Such key agreement TCerts, when one or more of these are included within a transaction, allow TCert owner(s) to independently determine whether or not they are a participant of the particular transaction, and if so, to retrieve a per-transaction decryption key by recovering the key agreement private key corresponding to the key agreement public key within the TCert. They also allow an authorized auditor to determine the identities of the transaction participants intended by the transaction creator

USING KEY AGREEMENT FOR SELECTIVE RELEASE OF TRANSACTION CREATOR'S ATTRIBUTES

- Use Validator group key-agreement public key
- Use intended recipients' key-agreement public keys
- Reflexive case: use transaction creator key-agreement public key
- This mechanism also enables targeted plaintext-access to chain-code and/or other data/metadata that may be included as ciphertext within the transaction
- Intended transaction participants are not necessarily granted equal access relative to one another or to the particular Validator group or to the transaction creator

TWO WAYS TO HANDLE TRANSACTION PARTICIPANT CONTINUITY: ACCESS CONTROL LIST (ACL) OF TCERTS VS. USER-UNIQUE ATTRIBUTES

- In order to securely determine whether a follow-on transaction is created only by an appropriate User, an intended transaction participant can be designated by one of their TCerts or by one of their attributes. It is easier to retain privacy if TCerts are not included within the ACL, since then a follow-on transaction need involve only a single TCert owned by the transaction creator (rather than an encrypted proof of ownership of one of the TCerts that were included in the ACL as well as (cleartext) inclusion of a previously-unused TCert)

- In a given transaction sequence, an attribute may at first be non- User-unique, and then migrate to a User-unique attribute once the User is "identified": e.g., a request is made for the services of a User possessing a certain skill/license/certification; a respondent proves possession of that requested non- User-unique attribute as well as possession of a User-unique attribute (such as a license number). Future transactions in the sequence entail that User-unique attribute

TWO WAYS TO HANDLE TRANSACTION PARTICIPANT ACCESS: KEY AGREEMENT TCERT VS. QUERY TRANSACTION

- If a query transaction is used by intended transaction participants to retrieve data/keys that the transaction creator authorized them to access, then such data/keys need be encrypted within the transaction itself for access by only the appropriate Validator group (where encryption involves a Validator group key agreement public key that is available to the transaction creator via a certificate or other reliable means)

- If key agreement TCert(s) of intended participants are available to and used by the transaction creator, then query transactions can be limited in scope (such as to acquire resultant state values). Use of key agreement TCert(s) of intended participant(s) enables additional feature of independent access by auditors authorized to decrypt the encrypted UserID and Affiliation fields (by virtue of the fact such Affiliation is included within such an auditor's jurisdiction)

TWO WAYS TO IDENTIFY TRANSACTIONS CREATED BY A SPECIFIC USER: GENERATING A LIST OR CHECKING A PARTICULAR TRANSACTION

- For use by Users and/or by authorized auditors
- The User-specific key needed to (re-)generate such a list is securely provided to the User with each successful request for one or more TCerts, where a successful request requires use of the private key corresponding to the ECert's public key. Such user-specific key may be securely provided out-of-band to an auditor that is authorized to access that User's TCert information (or the key from which the User-specific key is derived may be provided to a more powerful auditor that is authorized to access information for all Users)
- Each entry of the list is comprised of the encrypted TCertIndex field
- The TCertIndex is encrypted along with a known pad, so that a User can check this field in any particular transaction to see if the pad is present upon decryption with that User's User-specific key. An auditor that is authorized for the Affiliation to which the User belongs can check the encrypted UserID attribute field instead

05

Asset Transfer

ASSET TRANSFER FAMILY OF USE CASES

- A special case of asset transfer entails authorized import of an asset by, e.g., a bank administrator into a User's account with that bank
- The use of subaccounts/sub IDs dilutes leaked information: If a transfer is done by proving ownership of one subaccount and designating requested transfer of the asset to a distinct subaccount, then even the specific Validator group does not necessarily know whether or not those two subaccounts are owned by the same User. In this model, subaccount IDs are considered attributes that are included (encrypted) within TCerts – Similarly if a TCert includes a primary account ID. Typically, in a given transaction, the transaction creator would only selectively release one of the subaccount IDs (or a primary account ID). A transfer may be between distinct subaccounts, or between distinct primary accounts, or between a primary account and a subaccount (in either direction)
- If there is a high number of subaccounts associated with a User by the bank, the User can elect to include only a subset of these within their request to the TCA for a particular batch of TCerts