Transaction Immutability and Reputation Traceability: Blockchain as a Platform for Access-controlled IoT and Human Interactivity

David W. Kravitz
Crypto Systems Research
DarkMatter
San Jose, California
David.Kravitz@darkmatter.ae

***Abstract*— Credible reputation lies at the core of users and devices communicating and transacting successfully. Identity fraud is becoming increasingly difficult to manage in the face of massive-scale database breaches. In critical infrastructure and public safety applications, as well as day-to-day personal and business transactions, it is imperative to have a significant degree of confidence in whom/what one communicates with – whether to know if the recipient can be entrusted with the sender's data, or if the sender's data is to be considered reliably sourced. Even where possible, lost reputation is substantially more cumbersome, time-consuming and expensive to replace than are compromised, stolen or defective devices and their embedded cryptographic keys. This paper focuses on two methodologies that have considerable implications relative to addressing the reputation issue: (1) blockchain-enabled anomaly detection and assessment that involves dynamically asserted identity at the network edge effected through end-user targeted release of trusted behavioral data; (2) IoT and human interaction that is securely facilitated through use of an "Inviter-Invitee" protocol to set up dedicated maintainable "communication lines." The judiciously applied combination of the cryptographic protocol suites that enable the two methodologies results in a practicably implementable system for smart city use cases.**

Keywords—IoT, blockchain, reputation, identity management, access control, audit, data integrity, entity authentication, identity fraud, impersonation attack, key agreement, digital signature, smart city

I. Introduction

The intent of this paper is to lay out major elements of a solution to the problem of accurately assessing trustworthiness as myriad types of devices and humans interact with one another in increasingly automated and consequential exchanges. Despite the attention being focused on exposure via massive database breaches and on attacks against data availability for bitcoin-payable ransom collection, the real killer app may be in assuring or at least detecting attacks against data integrity and in handling impersonation attempts (where the "person" may be a device). We tackle the problem by integrating and expanding upon previous research into machine-to-machine (M2M) and human-to-machine (H2M) access control mechanisms, with a focus on leveraging the immutability, transparency and availability of blockchain transactions in order to gauge, update and apply reputation scoring of individual devices and of humans utilizing devices. Although it can be argued that for meeting the scalability and throughput demands of resource-constrained IoT environments, the main priority is that controllers do not command actuation unless the inputs to associated sensors can be verified as to being appropriately sourced and passing integrity checks, with assurance of confidentiality perhaps being optional – a major premise of this work is that properly and efficiently enabled confidentiality actually enhances the quality of such entity authentication and data integrity evaluation. In particular, the blockchain-enabled live collection and evaluation by authorized entities of otherwise-confidential peer-assessments of performance and expressed indicators of warranted suspicion of anomalies can be an invaluable system remediation tool.

Whatever methods we pursue are intended to provide the flexibility of accommodating terminal nodes that may be resource-constrained devices. This is one reason that the system has been designed with layered auditability measures, i.e., rather than every facet having to be highly decentralized or made otherwise complex in attempts to prevent (or catch in real time) unauthorized or unintended processing at the expense of significantly reduced performance, in many cases it is sufficient and advisable to instead be able to monitor activity out of the way of critical path execution as long as such surveillance capabilities are suitably access-controlled. Such access control of monitoring/audit must be bi-directional, in the sense that it must be infeasible to successfully bypass or circumvent such audit, and the ability to successfully audit relies on authorization to do so. Such auditability mechanisms are also useful in managing risk and addressing compliance and regulatory requirements in verticals such as health care, and financial applications. If a blockchain platform is to realistically fulfill the promise of cutting across silos in order to provide true advances in management of the varied supply chains that comprise smart city operations, such platform must be versatile enough to consolidate across verticals and handle IoT elements of each such supported vertical.

Because a properly permissioned blockchain introduces accountability into the process of responding to queries

978-1-5386-2487-6/17/$31.00 ©2017 IEEE
DOI 10.1109/PST.2017.00012

concerning retrieval of parts of the blockchain by light nodes that are not equipped to store or compute against the full chain, this renders the architecture more resilient. Local networks of IoT devices may include a more fully capable node, such as a smart refrigerator, as gateway in a home.

The import of roots of trust that justify claims of identity and possession of other attributes by users or devices potentially enables an end-node Client to outsource retention and use of signature (and/or key agreement) private keys by allowing the Client to be split into two components: local User Agent that has the authorization capability to manage acquisition of Enrollment Certificate and Transaction Certificates that reflects the identity and other attributes of that user/device, and remote Signature Service Provider that has access to the private keys that correspond to the public keys in the Enrollment Certificate and Transaction Certificates. The Signature Service Provider cannot independently transact on behalf of the user/device, because the User Agent does not share with the Signature Service Provider the Transaction Certificate- specific symmetric key derivation keys that are securely provided to the User Agent by a Transaction Certificate Authority. This mechanism is qualitatively different than a per-transaction multi-signature scheme (such as used in Bitcoin [1]), in that the User Agent need not participate at all in signing transactions. Additionally, a function of a password or other secret made available by the User Agent to the Signature Service Provider only when logged in, can make the Signature Service Provider more resistant to unauthorized use of signature or key agreement private keys.

This paper documents research that is follow-on from the work in [2] and [3]. While [2] and [3] introduced the necessary key management constructs (including those that enable auditability), it limited its import of trust relationships to those that only involve the immediate user or device. We extend that to incorporation of bi-lateral and multi-lateral relationships that are critical for establishing, maintaining and utilizing the reputation elements that we capitalize on.

Identity-based-/attribute-based- key agreement [4] does not offer the following critically required property of the blockchain platform: it does not naturally extend to a selectively-releasable proof-of-possession (PoP) of attributes signature scheme that simultaneously achieves authorized, passive, non-circumventable auditor visibility of such attributes. Such visibility applies to both the transaction creator that signs the transactions as well as the intended recipients (which may include the transaction creator user/device). Because we need a flexible means to efficiently control auditor access to such attributes by (in some cases covertly) distributing the appropriate (symmetric hierarchical key derivation-) non-transaction-specific keys independently of (and before, during or after) transaction creation, validation and block consensus flow, the signature Transaction Certificates and key agreement Transaction Certificates are included in the clear within the blockchain transactions. Therefore, in order to maintain privacy and resistance against useful traffic analysis (even by entities that are permissioned to transact on the blockchain), the Transaction Certificates are cryptographically structured so that the default condition is that they are unlinkable to one another by non- authorized observers of the transactions. Such authorization is granted either through the independent audit process or through selective release within the transaction and by the transaction creator of one or more attributes of the included signature Transaction Certificate(s) and/or key agreement Transaction Certificate(s). The default case is that such selective release is under encryption that is targeted to be decipherable by specific intended recipient- and/or transaction validation- entities. The selective release can be extended at any later time to other designated entities, without allowing for undetected modification. The Transaction Certificates are bound to the transactions, and the attributes are bound to the Transaction Certificates.

But, if Transaction Certificates are designed to not be reused, how does the system address distribution of key agreement Transaction Certificates to potential transaction-creating entities that intend to communicate privately on the blockchain to owners (i.e., possessors of the private key) of such key agreement Transaction Certificates? This paper sets up and utilizes two ways to do that, one of which, (a), is in-band of the blockchain, and the other of which, (b), is handled through requests to Transaction Certificate Authorities. Neither of these methods requires off-chain communications between the entities that wish to communicate with one another on-chain. This is crucial, because we want a uniform methodology that is practicably usable in automated M2M IoT environments. With regard to (a), we make use of a setup blockchain transaction in which a transaction-creator Client includes one of its own key agreement Transaction Certificates, and proves ownership of it. With regard to (b), in the ensuing discussion, we will show how the Inviter-Invitee matched protocols [5] can be utilized as a transferable means to convey authorization of one entity to request generation of key agreement Transaction Certificates owned by another entity such that the responding Transaction Certificate Authority can determine that the two entities have a pre-existing relationship. With particular relevance to managing reputation and devices making automated decisions on which transactions they should act upon, these relationships need not be one-to-one, in that they can be constructed to include delegation, as well as chained endorsements that inherit relationships that the inviter previously established as either an inviter or invitee. Reciprocally, successfully concluded M2M transactions may result in initiation of new Inviter-Invitee-administered persistent dedicated "communication lines." Furthermore, the conjunction of overlapping pair-wise or group-wise communication lines can result in new pair-wise or group-wise communication lines.

II. Trusted Provenance as a Precursor of Trusted Transactions

Since trusted transactions require trusted provenance, we make use of standard/legacy means to import static attributes (such as identities and affiliations) via assertions generated by existing Identity Providers (IdP) and dynamic attributes (such as pseudonymous identifiers, and properties that get updated (such as software version- dependent device functionality / security ratings, device/user reputation scores, active accounts, position titles, and resource entitlements) via assertions generated by existing Attribute Authorities (AA). Dependent on the device type (including its factory provisioning, hardware, storage and software capabilities) and requirements set by a particular IdP or

AA, proving justification for such assertions may involve the use of public-key cryptography (such as public key certificates and (keyless) attribute certificates that reference such public keys) and/or challenge-response scenarios that entail the device-side use of physically unclonable functions (PUF) for device authentication. PUFs may also be utilized for key/random nonce generation, memoryless repeatable-key storage, and anti-counterfeit [6]. As explained in section VI, IdP-issued assertions are converted to uniformly-constructed Enrollment Certificates (ECerts), and AA-issued assertions are converted to uniformly-constructed Transaction Certificates (TCerts). The aspect of uniformity renders irrelevant the means by which requests for assertions were justified with respect to how the TCerts are cryptographically structured, although the initial (overall and/or attribute-specific) reputation scores, if any, assigned within the resultant TCerts may reflect the robustness (or lack thereof) of the procedures and/or of the claimed device type used to request assertions.

III. Breach resistance via dynamically asserted identities

The efficacy of bringing anomaly detection to the edge of the network is only as strong as the trustworthiness of the end-entity users and devices.

Dynamically comprised neighborhoods of devices can be constructively utilized to corroborate or contradict claimed presence and activity of one another. This is useful in identifying candidates for revocation, and such "geo-metrics" can be supplemented with device biometrics where there are H2M interfaces.

Where there are human operators at both ends, users can rate one another's transactions within response transactions. In order to defend against undetected device misappropriation and to maintain determinations of user-identity continuity across devices, this may entail, in particular, providing an indication of level of confidence that the current user of the other device is who they claim to be by comparing against past ratings of the purported device and/or purported user.

Subsets/snippets of validated, time-stamped, immutable transactions propagated on the blockchain can later be released by users. Such correlated transactions provide circumstances of involvement in specific activities or locations, supplying evidence of whereabouts and behavior that is not spoofable -- even by fraudsters using misappropriated personally identifiable information (PII) as acquired, e.g., via Office of Personnel Management (OPM) and more recent, wider-scale Equifax database breaches. Involvement may include when and where devices used by the transaction-releasing individual were or weren't present, taking into account the veracity of such devices as attested/corroborated/contradicted by (time- or space-) neighboring devices/users. This brings aspects of anomaly- and identity fraud- detection and management to the edge of the network.

An app on a device may be designed and configured to be activated only when receiving biometric signals recognized as associated with the expected user. A transaction of value over a set threshold may require a TCert of specified wearable (where TCerts of wearable and smart phone are efficiently combinable [7]). In the absence of such biometrics, machine learning/big-data processing/predictive analytics may come to bear. Although our focus expands beyond Know Your Customer (KYC) to also accommodate KYM (Know Your Machine, [8]), our proposed strategy is consistent with the KYC goals of civic.com: "At scale it will no longer be enough to simply have PII to commit identity fraud, but proof of data ownership will be required as well" [9].

User reputation / device reputation reflected as attributes or as attribute qualifiers is selectively releasable (publicly, or confidentially to Validators and/or intended transaction recipients). Reputation thresholds as a condition of suitability of transactions used to determine if or how candidate transactions are processed may be set by use-case- specific policy, as enforceable by Validators of transactions submitted to the blockchain. Such reputation thresholds may apply to signature TCerts (owned by transaction creators) and/or key agreement TCerts (owned by transaction recipients).

IV. Identity and reputation feedback loops

The composite trust model is based on trusted users, trusted devices, and users trusted based, in-part, on use of trusted devices. Device trust can be enhanced by making adversarial intrusion more difficult to succeed. This may entail (a) server-authorized dynamic transformation of the client that is differentially detectable from adversarial modification of the client (as achievable through client responses to server challenges) and/or (b) server-based dynamically refreshed locking/unlocking of client-local key store modules [10].

As explained further in section VII , the system incorporates "Inviter-Invitee" protocol runs to accommodate endorsements via attribute certificate chaining incorporated into resultant "communication lines" [5]. As part of an inviter protocol run, an inviter may be able to opt whether or not unambiguous references (accomplished, for example, via hashing) to one or more of its attribute certificates are permitted to be included within attribute certificate(s) owned by the invitee that result from successful execution of the invitee protocol. The invitee may be able to choose whether or not to include such references in the resultant attribute certificate(s). This "chaining" of attribute certificates can be used to supply a chain of endorsements as a component of a reputation scoring method enabled, in part, through inviter and invitee processing. The initiation of "communication lines" may be inspired by positive experience with pair-wise / group-wise blockchain transactions, where inviters may base choice of invites, in part, on current reputation of potential invitees. Invitees can check current reputation of inviters as a condition of acceptance. The existence of dedicated communication lines may be a prerequisite to entrusting others with properly handling sensitive data, and/or believing data (prior to precipitating user action or device reconfiguration/recalibration). Performance metrics of established communication lines affect reputation of participating users/devices.

V. A scalable hybrid transaction model

Consider, for example, an on-chain physician order to activate an IV apparatus, where the IV apparatus does not wait for and may remain oblivious of payment aspects. A secure, non-hacked IoT device will not report service fulfillment ahead of actually providing such service (e.g., life-saving IV drip). Payment can be via cryptocurrency or off-chain- reconciled monetary exchange. Reputation metrics, as incorporated and updated into TCerts, play a vital role in enabling a highly scalable and responsive concurrent- or post- service-delivery payment reconciliation model. Reputation of devices and of users is dependent upon perceived device robustness (which may change during the life-cycle of a given instance of a device), payment timeliness, and service performance timeliness, completeness and accuracy. Adoption of this model can reduce dependency on complex fully-automated, non- fully-vetted/understood "smart contract" code.

VI. Registering Users and Devices and Populating/Refreshing Attributes Database

A. *Registration*

User Agent (UA) and Registration Authority (RA) perform mutually authenticated Transport Layer Security (TLS) handshake protocol, where UA generates an elliptic curve Digital Signature Algorithm (ECDSA) key pair and corresponding self-signed certificate Cert_{UA}. Subsequent communications between UA and RA are TLS-protected.

RA extracts Cert_{UA} from TLS handshake, and issues signed authentication request to UA:

AuthnRequest =

$$<\text{AuthnRequest(RA\_ID, Nonce}_{\text{RA}}\text{, Timestamp}_{\text{RA}}\text{, Cert}_{\text{UA}}) \parallel \text{Sign}_{\text{RA private key}} (<\text{AuthnRequest}>) \quad (1)$$

Timestamps here and elsewhere may be included for use during audit.

User/UA registers with / logs into Signature Service Provider (SSP), and UA passes <AuthnRequest>. SSP generates (ECDSA) Enrollment private key – Enrollment public key key-pair.

SSP provides to UA:
$\text{Sign}_{\text{Enrollment private key}}$ (<AuthnRequest>), and Enrollment public key.

SSP retains Enrollment private key as associated with User login credential(s).

NOTE: Here and subsequently, use of an SSP as an entity that is disjoint from UA is optional, and its presence or absence does not differentially affect the system flow.

UA communicates with (internal or external) Identity Provider IdP. Communications are TLS-protected under mutual authentication (with self-signed Cert_{UA} (same as within AuthnRequest) provided by UA). IdP extracts Cert_{UA} from TLS handshake, and incorporates it into the Assertion it issues.

UA provides to IdP:
AuthnRequest, Enrollment public key, and
$\text{Sign}_{\text{Enrollment private key}}$ (<AuthnRequest>).

User/UA provides proof of ownership of UserID and Affiliation, as required by IdP. Such proof may involve the use of pre-existing private key(s) held by the UA. For example, a pre-existing public key may be included within a certificate that establishes the private key owner's UserID and/or Affiliation, where such certificate was issued by an entity recognized by the IdP to have such authority. Such entity may be the IdP itself.

IdP issues signed assertion to UA:

Assertion =

$$<\text{Assertion(IdP\_ID, Nonce}_{\text{RA}}\text{, Timestamp}_{\text{IdP}}\text{, Cert}_{\text{UA}}\text{, UserID, Affiliation, AuthnRequest, Enrollment public key, Sign}_{\text{Enrollment private key}} (<\text{AuthnRequest}>))> \parallel \text{Sign}_{\text{IdP private key}} (<\text{Assertion}>) \quad (2)$$

UA provides Assertion to RA. RA makes Assertion available to Enrollment Certificate Authority (ECA) if it is properly formulated, where, in particular, signatures verify, Nonce_{RA} is current and Cert_{UA} field within Assertion and Cert_{UA} field within included AuthnRequest match Cert_{UA} extracted from current TLS handshake.

ECA generates Enrollment Certificate (ECert), as made available (e.g., via LDAP server or TLS communications) to the ECA, the Attribute Certificate Authority (ACA), Transaction Certificate Authority (TCA), and authorized Auditors upon presentation of UserID and Affiliation. ECA may make use of threshold cryptography for digital signature generation. If so, each signer unit may have independent secure access to RA-approved Assertions (so as not to have an indirect trust model, even if Assertions are not RA-signed).

NOTE: A rogue ECA cannot undetectably substitute another Enrollment public key for which it knows the corresponding Enrollment private key, since the ECA cannot generate an acceptable Assertion without knowledge of an IdP private key.

B. *Attribute Certificate Authority database refresh*

User Agent (UA) and ACA perform mutually authenticated TLS handshake protocol, where UA generates an ECDSA key pair and corresponding self-signed certificate Cert_{UA}. Subsequent communications between UA and ACA are TLS-protected.

ACA extracts Cert_{UA} from TLS handshake, and issues signed authentication request to UA:

AuthnRequest =

$$<\text{AuthnRequest(ACA\_ID, Nonce}_{\text{ACA}}\text{, Timestamp}_{\text{ACA}}\text{, Cert}_{\text{UA}})> \parallel \text{Sign}_{\text{ACA private key}} (<\text{AuthnRequest}>) \quad (3)$$

The step below is iterated, as necessary, across multiple (internal or external) attribute authorities (AA), with disjoint or overlapping attribute(s): UA communicates with AA. Communications are TLS-protected under mutual authentication (with Cert_{UA} (same as within AuthnRequest) provided by UA).

UA provides AuthnRequest and UserID, where UserID should be comparable (if not identical) to UserID within the User's Enrollment Certificate.

User/UA provides proof of ownership of UserID and attribute(s), as required by AA. Such proof may involve the use of pre-existing private key(s) held by the UA. For example, a pre-existing public key may be included within a certificate that is referenced by an attribute/authorization certificate or SAML assertion, where such certificate(s)/assertions were issued by entity(ies) recognized by the AA to have the relevant authority. Such entities may overlap with the AA itself.

AA issues signed assertion:

$$\text{Assertion} = \langle \text{Assertion(AA\_ID, Nonce}_{\text{ACA}}\text{, Timestamp}_{\text{AA}}\text{, Cert}_{\text{UA}}\text{, UserID, attribute(s), attribute validity period(s))} \rangle \parallel \text{Sign}_{\text{AA private key}} (\langle \text{Assertion} \rangle) \quad (4)$$

UA logs into Signature Service Provider (SSP), and passes <Assertion>(s) (as well as UserID and Affiliation (from User registration) if not already associated at the SSP with the User's account).

If a given attribute has a currently valid Assertion within the ACA database, then UA can pass a "null assertion" (that identifies the attribute) to SSP in place of an AA-issued <Assertion>. Such null assertion is signed by SSP in place of signing an AA-issued <Assertion>.

SSP provides Sign$_{\text{Enrollment private key}}$ (<Assertion>)(s) and Sign$_{\text{Enrollment private key}}$ (null assertion)(s) to UA. UA provides AA- and SSP- signed assertion(s), SSP- signed null assertions, UserID and Affiliation to ACA, where UserID and Affiliation should be identical to such fields within a previously issued Enrollment Certificate (ECert). ACA randomly/pseudo-randomly generates and communicates Rand to UA if there is at least one freshly signed assertion (where Rand is retained by UA and used to refer to ACA database entry during TCert requests). UA is responsible for providing appropriate previously issued Rand value(s) to ACA as matched against SSP- signed null assertion(s), if any – so that ACA can append or point to these within the currently processed ACA database entry.

ACA verifies that AA- and SSP- signed assertion(s) and null assertions are properly formulated, where, in particular, signatures verify, Nonce$_{\text{ACA}}$ is current and Cert$_{\text{UA}}$ field within Assertion and Cert$_{\text{UA}}$ field within included AuthnRequest match Cert$_{\text{UA}}$ extracted from current TLS handshake.

ACA makes AA- and SSP- signed assertion(s) and SSP-signed null assertions, UserID and Affiliation available to TCA by appropriately updating ACA database (incorporating hash(Rand), and any previous hash(Rand) values as appropriate).

ACA database encryption option: After verifying signatures (using AA public key(s) and Enrollment public key (per ECert that corresponds to UserID and Affiliation), the ACA can encrypt using one-pass elliptic curve Diffie-Hellman (ECDH) key agreement with a locally generated ephemeral key pair and the current ECDH public key of the TCA (if such is securely available to the ACA via a certificate or other means). If the ephemeral is generated using a true random number generator (as opposed to wholly via a deterministic process), then the key agreement- based encryption will not be reversible by the ACA. The associated hash(Rand) values can remain in the database in the clear, so that the ACA can effectively handle null assertions.

NOTE: ACA is responsible for determining if UserID included within an Assertion is comparable (if not identical) to the UserID associated with the Enrollment public key (via an ECert).

VII. Asynchronous Inviter Protocol- and Invitee Protocol- Cryptographic Messaging and Processing

A. Introduction to device provisioning and Inviter-Invitee protocols and usage

Before embarking on details of an example embodiment of inviter and invitee message construction and processing, we illustrate at a higher level the principles involved in Fig. 1 and Fig. 2. Digital Identity Token (DIT) is introduced in Fig. 1.

B. Inviter protocol: designed to be "Invitee" phishing-resistant

Inviter's client sends to Mediating Service Provider:

- Pointer Value(s) or nicknames or other means for service provider to locate particular relevant attribute certificates.

- DIT_part_1

DIT_part_1 is comprised of DIT_ID, New Pointer Value, Invitee Email Address (and/or other contact information for intended invitee that is usable by service provider for invitee that is not necessarily registered with the service provider), and the encryption for access by the service provider's AA of: [Rand_1, Rand_2, Rand_3, and a digital signature (generated using an inviter's signature generation private key) over DIT_ID, Private_Identifier_1 and New Pointer Value]. New Pointer Value is the Pointer Value that will be used for the attribute certificate to be generated by the service provider's AA if the invitee protocol runs successfully.

$$\text{Private\_Identifier\_1} = \text{hash(answer} \parallel \text{Rand\_1)} \quad (5)$$

$$\text{Private\_Identifier\_2} = \text{hash(answer} \parallel \text{Rand\_2),} \quad (6)$$

where New Pointer Value = hash(Private_Identifier_2).

$$\text{KEK} = \text{hash (answer} \parallel \text{Rand\_3),} \quad (7)$$

where KEK is a key encryption key.

Inviter's client sends to Mediating Service Provider:

- Data_part_1 (optional)

Data_part_1 is comprised of a digital signature (generated using an inviter's signature generation private key) over DIT_ID and Confidential Data and Rand_4, and the encryption for access by the service provider's AA of: [Confidential Data and Rand_4]; Rand_4 can be used to hide low-entropy Confidential Data against guessing; Confidential Data may include sensitive data concerning the intended invitee's association with the inviter and/or with the inviter's institution.

- DIT_part_2, where the DIT_part_2_data component of DIT_part_2 includes "security question" if a (security question, answer) pair is used.

- Data_part_2 (optional) DIT_part_2 and Data_part_2 are releasable to the purported invitee's client or browser upon successful login.

- DIT_part_3 (optional)

- Data_part_3 (optional)

DIT_part_3 and Data_part_3 are releasable to the purported invitee's client or browser if that service provider subscriber has passed Test_1.

Inviter's client sends to Mediating Service Provider:

- DIT_part_4 (optional)
- Data_part_4 (optional)

DIT_part_4 and Data_part_4 are releasable to the purported invitee's client or browser if that service provider subscriber has passed Test_2.

For 2 ≤ i ≤ 4, DIT_part_i is comprised of DIT_ID, DIT_part_i_data, hash(Data_part_i), and DIT_part_i_signature, where DIT_part_i_signature is a digital signature generated over DIT_ID, DIT_part_i_data and hash(Data_part_i).

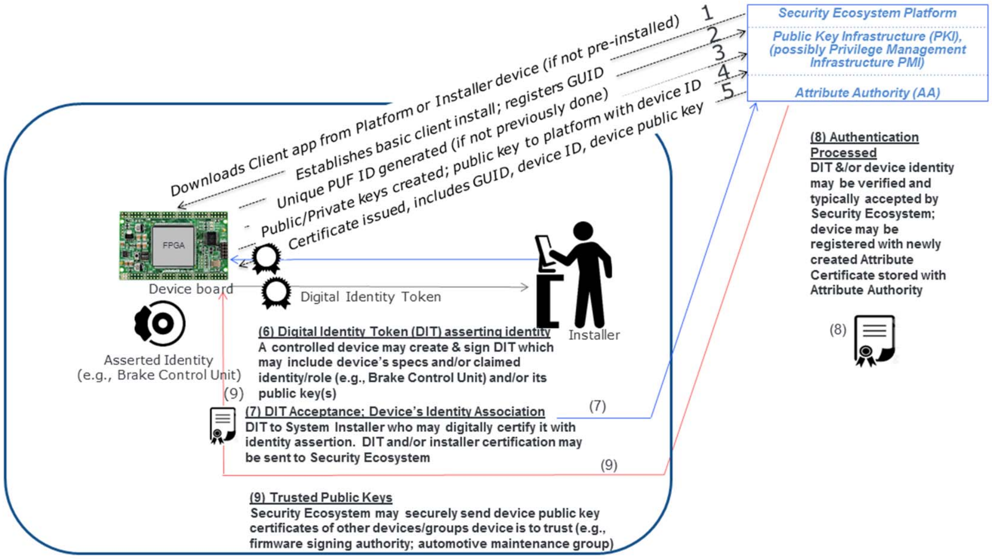
Security Ecosystem Platform
Public Key Infrastructure (PKI), (possibly Privilege Management Infrastructure PMI)
Attribute Authority (AA)
1 Downloads Client app from Platform or Installer device (if not pre-installed)
2 Establishes basic client install; registers GUID
3 Unique PUF ID generated (if not previously done)
4 Public/Private keys created; public key to platform with device ID
5 Certificate issued, includes GUID, device ID, device public key
FPGA
Device board
Digital Identity Token
Installer
Asserted Identity (e.g., Brake Control Unit)
(6) Digital Identity Token (DIT) asserting identity
A controlled device may create & sign DIT which may include device's specs and/or claimed identity/role (e.g., Brake Control Unit) and/or its public key(s)
(7) DIT Acceptance; Device's Identity Association
DIT to System Installer who may digitally certify it with identity assertion. DIT and/or installer certification may be sent to Security Ecosystem
(7)
(9)
(9)
(9) Trusted Public Keys
Security Ecosystem may securely send device public key certificates of other devices/groups device is to trust (e.g., firmware signing authority; automotive maintenance group)
(8) Authentication Processed
DIT &/or device identity may be verified and typically accepted by Security Ecosystem; device may be registered with newly created Attribute Certificate stored with Attribute Authority
(8)

Fig. 1. Provisioning an IoT device in an automotive use case example

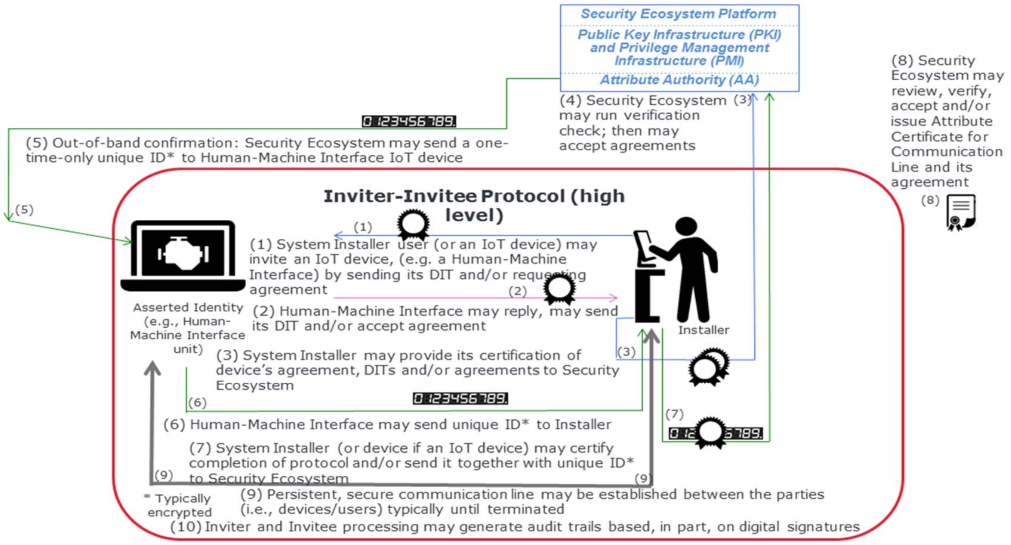
Security Ecosystem Platform
Public Key Infrastructure (PKI) and Privilege Management Infrastructure (PMI)
Attribute Authority (AA)
0123456789.
(5) Out-of-band confirmation: Security Ecosystem may send a one-time-only unique ID* to Human-Machine Interface IoT device
(4) Security Ecosystem may run verification check; then may accept agreements
(3)
(8) Security Ecosystem may review, verify, accept and/or issue Attribute Certificate for Communication Line and its agreement
(8)
(5)
Inviter-Invitee Protocol (high level)
(1)
(1) System Installer user (or an IoT device) may invite an IoT device, (e.g. a Human-Machine Interface) by sending its DIT and/or requesting agreement
(2)
(2) Human-Machine Interface may reply, may send its DIT and/or accept agreement
Asserted Identity (e.g., Human-Machine Interface unit)
Installer
(3) System Installer may provide its certification of device's agreement, DITs and/or agreements to Security Ecosystem
(3)
0123456789.
(6)
(6) Human-Machine Interface may send unique ID* to Installer
(7)
0123456789.
(7) System Installer (or device if an IoT device) may certify completion of protocol and/or send it together with unique ID* to Security Ecosystem
(9)
(9)
* Typically encrypted
(9) Persistent, secure communication line may be established between the parties (i.e., devices/users) typically until terminated
(10) Inviter and Invitee processing may generate audit trails based, in part, on digital signatures

Fig. 2. Inviting of an IoT device in an automotive use case example

C. *Invitee protocol*

Part of the processing may involve the generation of one or more digital signatures attributable to the invitee, particularly if non-repudiation relative to acceptance of one or more Digital Agreements is involved.

Service provider transmits to invitee:

- DIT_part_2
- Data_part_2

Invitee transmits to service provider:

- Nonce_Invitee_1 encrypted for access by AA (where Nonce_Invitee_1 is generated by invitee's client or browser or peripheral device)

Service provider transmits to invitee:

- [Nonce_Invitee_1, Rand_1, and Nonce_AA_1] encrypted for access by invitee (where Nonce_AA_1 is generated by AA).

If Nonce_Invitee_1 is correct, invitee transmits to service provider:

- [Nonce_AA_1, Private_Identifier_1 = hash(answer || Rand_1), and Nonce_Invitee_2] encrypted for access by AA, where "security question" is found within DIT_part_2

If Nonce_AA_1 is correct and the previously received digital signature over DIT_ID, Private_Identifier_1 and New Pointer Value (from DIT_part_1) verifies correctly using Private_Identifier_1 from invitee, then service provider transmits to invitee:

- [Nonce_Invitee_2, Rand_2, and Nonce_AA_2] encrypted for access by invitee

AA may (preferably securely) now or later archive [DIT_ID, Private_Identifier_1, New Pointer Value, and digital signature over DIT_ID, Private_Identifier_1 and New Pointer Value] as part of the information relating to the invite that may be subject to possible audit.

Service provider may also transmit to invitee:

- [DIT_part_3 and Data_part_3 (if available), since the invitee has passed/satisfied Test_1

If Nonce_Invitee_2 is correct, invitee transmits to service provider:

- [Nonce_AA_2, Private_Identifier_2 = hash(answer || Rand_2), and Nonce_Invitee_3] encrypted for access by AA

If Nonce_AA_2 is correct and if hash(Private_Identifier_2) computed over Private_Identifier_2 from invitee matches New Pointer Value from DIT_part_1 in invite, then service provider transmits to invitee:

- [Nonce_Invitee_3, Rand_3, and New Pointer Value] encrypted for access by invitee.

AA may now or later generate an attribute certificate to be owned by the invitee, and which includes Private_Identifier_2 as one of its fields. New Pointer Value, as known to the invitee, as well as by construction by the inviter (and its delegates, if any) will point to such new attribute certificate that represents successful completion of the Inviter-Invitee protocol processing. New Pointer Value does not necessarily need to be transmitted to the invitee as shown above, if there exist other means (such as assigned nickname) by which the attribute certificate can later be located from the invitee's data accessible to the service provider.

Service provider may also transmit to invitee:

- [DIT_part_4 and Data_part_4 (if available), since the invitee has passed/satisfied Test_2.

If Nonce_Invitee_3 is correct, then Rand_3 may be used at invitee's client or browser and/or peripheral device to derive KEK as hash (answer || Rand_3), where KEK is a key encryption key used originally by inviter. That KEK may potentially be used by the invitee immediately if the service provider has made available to the invitee data that has been encrypted using a key derivable via the KEK. We will demonstrate in the next section that retention of such KEKs is not required when combining use of Inviter-Invitee communication lines with permissioned blockchains.

VIII. Leveraging Inviter-Invitee Results on Blockchain

The goal here is to enable "pre-validation" that eases submitted transaction Validation phase requirements for IoT-friendly speed up.

Client logs in to Mediating Service Provider to associate Enrollment Certificate with the Client's profile. Mediating Service Provider checks for static identity/identifier match, and can require proof of possession of Enrollment Private Key.

Client logs in to Mediating Service Provider to request AA-issued specialized attribute certificates that each reference Enrollment Certificate of requester and include one or more pairs of hash(Enrollment Certificate) and associated Pointer Value. The Pointer Value may be for an attribute certificate for a communication line that indicates requester is eligible to transmit to the owner of that Enrollment Certificate. This eligibility may result from an Inviter-Invitee protocol run, where the Client was an inviter or invitee or a specified delegate. This may have been a direct invitation or one that leveraged chained attribute certificates. The AA can require the Client to prove it has legitimate access rights to the Pointer Value(s).

If Mediating Service Provider's AA is recognized by Subordinate TCA, Client can request a batch of key agreement TCerts owned by the entity that owns one or more of the attributes that the requesting Client identifies (as learned by the Client through querying the Mediating Service Provider via the Pointer Values). Here a Subordinate TCA is subordinate to Primary TCA that generates TemplateTCerts based on the ACA database population- and refresh- model that was discussed in section VI. Subordinate TCA matches the indicated hash(Enrollment Certificate) of the AA-issued specialized attribute certificate against one or more TemplateTCerts that include hash(Enrollment Certificate) as an index. The request can potentially be approved either way, but the resultant key agreement TCerts can include a flag visible to key agreement TCert owner that indicates whether or not communication line exists (and indicates that the key agreement TCert is not "reflexive," i.e., that it is not owned by its requester). Key agreement TCerts are not successfully reusable, as enforced via Validation.

As an anti-phishing measure, Subordinate TCA does not selectively release sensitive attributes of issued TCerts -- even if known by non-reflexive key agreement TCert requester – unless, at a minimum, flag is set because of requester-demonstrated knowledge of static or pseudonymous identity/identifier. For example: Account number is revealed by that account holder within a blockchain transaction responsive to a previous blockchain "flagged" transaction (where that previous-transaction creator may have selectively released one or more relevant attributes of the signature TCert). Account numbers may be subaccount numbers (to dilute account activity leakage). Common ownership of subaccounts is not revealed even if recipients from different subaccounts collude.

IX. SUPPLY CHAIN PROVENANCE: TRANSFERRING REPRESENTATIONS OF DEVICE OWNERSHIP

Example flow is comprised of Transactions A, B and C:

TXN A: Device Manufacturer → Distributor

TXN B: Distributor → Consumer i

TXN C: Consumer i → Consumer j.

Below, { } represents plaintext, and {{ }} represents ciphertext. The use of authenticated encryption assures that released payload decryption keys are legitimate, where ciphertext payloads are immutable. TCerts are self-contained, in that an internal integrity check would detect falsification of selectively released attribute keys.

1. Device Creation (TXN A): payload ⊃ {{Device Serial Number(s)}}; metadata ⊃ {Device Manufacturer signature TCert} with {{"selectively released" attribute(s) key(s)}} + {Device Manufacturer-acquired Distributor- owned key agreement TCert} with {{Distributor attribute key}}

2. First Sale (TXN B): payload ⊃ {{specific Device Serial Number and decryption key for (full or partial) payload of TXN A}}; metadata ⊃ {Distributor signature TCert} with {{attribute(s) key(s)}} + {Distributor-acquired Consumer i-owned key agreement TCert with pseudonym attribute key}

3. eBay (TXN C): payload ⊃ {{decryption key for (full or partial) payload of TXN B}}; metadata ⊃ {Consumer i signature TCert with pseudonym attribute key} (where pseudonym matches TXN B) + {Consumer i- acquired Consumer j- owned key agreement TCert with pseudonym attribute key}

X. AN M2M USE CASE

This example use case is applicable to ad hoc colonies of devices that can organize for fulfillment of a task, such as traffic flow coordination during emergency situations. A call for device participation is announced via one or more blockchain transactions. Such call may specify acceptance criteria, such as minimum/threshold attribute rating scores. Responses to the call by qualified devices are incorporated into the blockchain as transactions. Recall that devices can use factory-provisioned certificates, PUFs and/or other means in order to prove attributes to ACA via AA-issued assertions. Actual fulfillment of tasks by devices can be done off-chain after being set up on-chain. The signature TCerts used to respond to calls for device participation can be safely reused in this case, i.e., for TLS mutual authentication. Upon completion of off-chain tasks, devices can rate one another's performance on-chain by referencing devices' TCerts. This does not require that devices necessarily observe one another's static identities. The ratings can be encrypted for access only by an Analytics Processor (AP), e.g., by using an individual- or group- key agreement public key extracted from a key agreement TCert that has an AP attribute or from a standard public key certificate. An authorized AP can cluster individual ratings according to deviceID (using the same passive TCert attributes- readout technique as that employed by authorized auditors). Then the AP, acting similarly to an external AA, can issue assertions – which, in this case, assert device possession of cumulative rating attributes, where such ratings may quantify the reputation aspects of other attributes possessed by the device or of the overall device. This M2M scenario is extensible to H2M, whereby an AP can detect whether or not there is a match between a device that is present at an establishment or during online service use and a device that later submits a rating/vote of such establishment or service. To accomplish this, the AP clusters TCerts according to the owning users, even where a given such user utilizes multiple devices. Sybil attacks that involve over-voting by devices are thus thwarted through detection, as are attacks in which users submit ratings for establishments/services at which they lacked presence.

Devices can incorporate "self"- key agreement TCerts into M2M response transactions relative to call-for-task-participation transactions. They can do this in such a way as to selectively release a pseudonym or other unique- or group-attribute in order to prove key agreement TCert ownership matches that of the signature TCert (at the same device- or same group of devices- level). This enables devices to then successfully communicate with one another confidentially on the blockchain (instead of or in addition to communicating with one another off-chain, as mutually authenticated using TLS ECDHE-ECDSA based on the signature TCerts). ECDHE-ECDSA denotes the use within the Transport Layer Security handshake protocol of the elliptic curve Digital Signature Algorithm to sign and verify ephemeral public keys, whereby these elliptic curve Diffie-Hellman ephemeral key pairs are combined (where each entity uses its own ephemeral private key and the ephemeral public key of the other entity) in order to result in a shared secret for key agreement.

These M2M ad hoc task activities can be used as a source of setting up new invites (including automation of question-answer setup without involving a third party in distribution of secret answers). Such invites can include a "long-term" group ID, and such group ID can be reused across multiple invites from different inviters and potentially across devices that were members of distinct (possibly overlapping) ad hoc groups. Such group IDs can be incorporated into attribute certificates that are generated as a result of successful Inviter-Invitee protocol runs.

Then such group IDs can consequently be passed along into specialized attribute certificates that reference Enrollment Certificates. In that case, such group IDs can be incorporated as attributes into signature TCerts by Subordinate TCAs. Fig. 3 depicts a group of IoT devices within a vehicle, at the exclusion of taking action on sensor or control inputs from other devices that don't have communication lines set up. Fig. 4 represents the devices of a vehicle securely communicating with one another and interfacing to an installer. Fig. 5 illustrates that a required aspect of successfully setting up a communication line at participating endpoint devices may be the local incorporation of pair-wise (or group-wise) programmed rules and business logic that govern the ensuing communications. Devices can potentially be factory- or over-the-air- provisioned with scoping parameters that are used towards assessing whether or not an invite is acceptable and/or when inviting other devices, relative to the proffering of Digital Agreements. Whether humans and/or devices comprise the endpoints of such agreements, hacked devices or cheating users may be held accountable for violating a Digital Agreement. Violations may be result in penalization via reduced reputation scores. With respect to effectively managing the boundary between the vehicle and the external world, there is an issue of appropriately gauging context, such as to allow system override under suitable circumstances. For example, the default behavior may be that parts of a specific vehicle and the current driver/passengers of that vehicle are recognized by one another as being within a pre-authorized/white-listed group (as set up via Inviter-Invitee-orchestrated communication lines or other means), while actions of / effects upon "foreign" vehicle parts and/or drivers/passengers/pedestrians are considered only under appropriate circumstances. For example, should braking of a vehicle three cars in front have an effect (perhaps when corroborated by at least partially independent actions of other vehicles), while braking of a vehicle a half-mile away should be disregarded? Should intrusion of a body part of a driver/passenger of the vehicle disable the window from being powered closed, while allowing a driver/passenger to force closed a window against someone outside the vehicle? In a case of conflicting inputs, should there be a hierarchy/weighting system, so that, for example, a heavily- hardware-secured roadside infrastructure unit is taken more seriously than an easily modifiable / questionably robust "foreign" vehicle component (at least in the absence of independent corroboration)? Such weighting may be based, in part, on the assignment of reputation scores.

XI. Conclusion

We have concentrated in this paper on development of a reputation system that is compatible with permissioned blockchains, and indicated how such a system can drive the efficiency and scalability of the blockchain to the point that critical infrastructure highly time-sensitive IoT operations can occur off-chain in real time. In particular, the fact that the disposition of payment for services (and the quality of performance of rendered services) is reflected in updated reputation scores allows services to be performed without waiting for confirmation or reconciliation of payments.

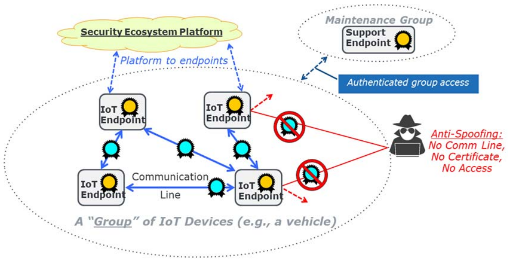

Fig. 3. Grouping of devices

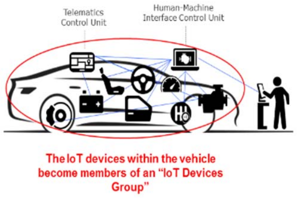
Telematics Control Unit
Human-Machine Interface Control Unit
The IoT devices within the vehicle become members of an "IoT Devices Group"

Fig. 4. Device-group secure communications and interfaces

The blockchain transactions that set up services that may be performed off-chain, and that are used to report on results of performed services, can be efficiently validated and incorporated into blocks via permissioned, accountable consensus algorithms (such as PBFT [11], [12]), while actual payments can be handled through more unwieldy mechanisms (such as cryptocurrency based on proof-of-work mining consensus).

Although we relied on key management that securely and efficiently addresses passive authorized auditability simultaneously with selective release of proof-of-possession of attributes, and reused such auditability mechanism for analytics processing that is necessary for reputation score updating, we did not recapitulate here these structures that were previously detailed in [2] and [3]. Fig. 6 depicts a pictorial synopsis of such key management.

References

[1] https://en.bitcoin.it/wiki/Multisignature

[2] D. Kravitz, J. Cooper, "Securing User Identity and Transactions Symbiotically: IoT Meets Blockchain," IEEE Global Internet of Things Summit (GIoTS) 2017, June 2017.

[3] D. Kravitz, US Patent Application 20170147808, "Tokens for Multi-tenant Transaction Database Identity, Attribute and Reputation Management"

[4] D. Fiore, R. Gennaro, "Making the Diffie-Hellman Protocol Identity-based," Proceedings of CT-RSA 2010, LNCS Vol. 5985, Springer, 2010, pp. 165-178.

[5] D. Kravitz, D. Graham, J. Boudett, R. Dietz, US Patent 9,270,663, US Patent 9,356,916, US Patent 9,445,978, US Patent 9,578,035, US Patent 9,716,595.

[6] http://rijndael.ece.vt.edu/puf/background.html

[7] Y. Lindell, "Fast Secure Two-Party ECDSA Signing," Crypto 2017, LNCS Vol. 10402, Springer, 2017, pp. 613-644.

[8] P. Wong, keynote abstract, http://wfiot2016.ieee-wf-iot.org/program/

[9] https://cryptostreet.co/civic-the-identity-verification-solution/

[10] D. Kravitz, D. Graham, J. Boudett, R. Dietz, US Patent 9,445,978, "System and method to enable PKI- and PMI- based distributed locking of content and distributed unlocking of protected content and/or scoring of users and/or scoring of end-entity access means--added"

[11] M. Castro, B. Liskov, "Practical Byzantine Fault Tolerance and Proactive Recovery," ACM Transactions on Computer Systems, Vol. 20, No. 4, 2002, pp. 398-461

[12] https://github.com/hyperledger-archives/fabric/wiki/Consensus

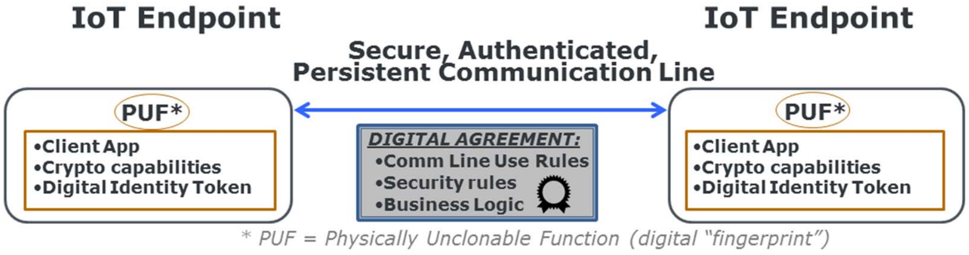

Fig. 5. Communication line reflecting acceptance of digital agreement

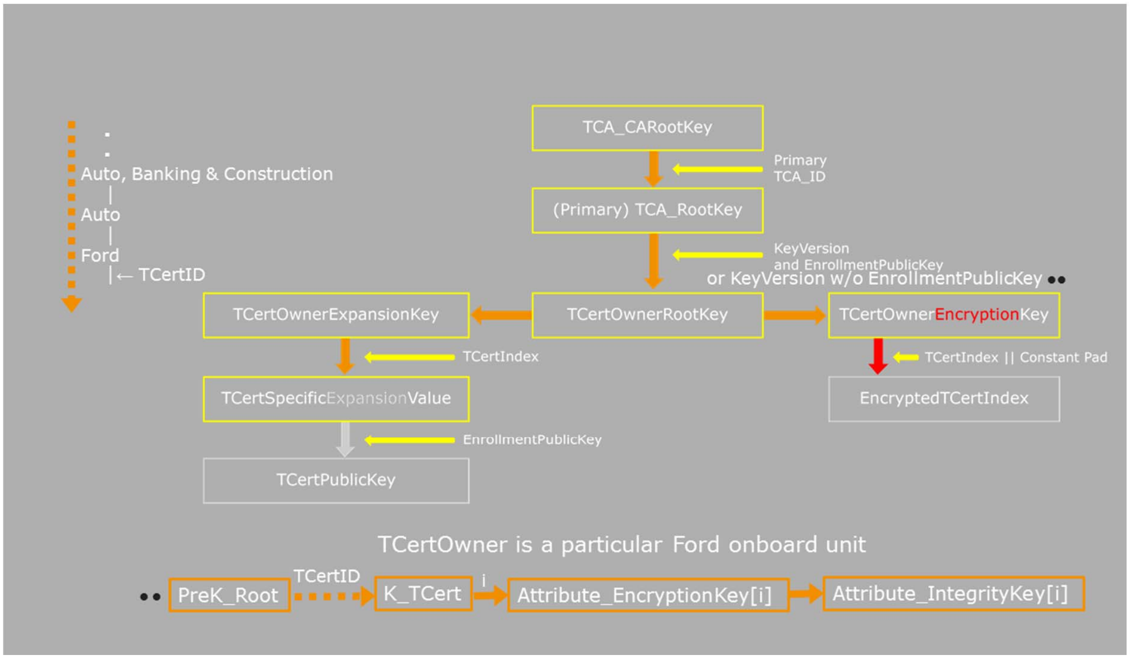

Fig. 6. Depiction of cryptographic key management of auditable Transaction Certificate- based permissioned blockchain