Security Concerns

2020/01/14
Discussion W3C DID WG
Samuel M. Smith Ph.D.
https://github.com/SmithSamuelM/Papers
sam@samuelsmith.org

Background References

Self-Certifying Identifiers:

Girault, M., “Self-certified public keys,” EUROCRYPT 1991: Advances in Cryptology, pp. 490-497, 1991

https://link.springer.com/content/pdf/10.1007%2F3-540-46416-6_42.pdf

Kaminsky, M. and Banks, E., “SFS-HTTP: Securing the Web with Self-Certifying URLs,” MIT, 1999

https://pdos.csail.mit.edu/~kaminsky/sfs-http.ps

Mazieres, D. and Kaashoek, M. F., “Escaping the Evils of Centralized Control with self-certifying pathnames,” MIT Laboratory for Computer Science, 2000

http://www.sigops.org/ew-history/1998/papers/mazieres.ps

Mazieres, D., “Self-certifying File System,” MIT Ph.D. Dissertation, 2000/06/01

https://pdos.csail.mit.edu/~ericp/doc/sfs-thesis.ps

Smith, S. M., “Key Event Receipt Infrastructure (KERI) Design and Build”, arXiv, 2019/07/03

https://arxiv.org/abs/1907.02143

Certificate Transparency:

Laurie, B., “Certificate Transparency: Public, verifieable, append-only logs,” ACMQueue, vol. Vol 12, Issue 9, 2014/09/08

https://queue.acm.org/detail.cfm?id=2668154

Google, “Certificate Transparency,”

http://www.certificate-transparency.org/home

Laurie, B. and Kasper, E., “Revocation Transparency,”

https://www.links.org/files/RevocationTransparency.pdf

Identifier Authority

Must First Establish Control Authority over an Identifier

URI

```
URI = scheme:[//authority]path[?query][#fragment]
authority = [userinfo@]host[:port]
```

DID

```
"did:" method-name ":" method-specific-id
```

Use Public/Private Key Pair (Certificate) to Verify Control Authority over an Identifier

DNS Certificate Authority Infrastructure

DID Method Authority Infrastructure

Identifier Authority Infrastructure

DNS Certificate Authority Infrastructure
- Multiple Certificate Providers
- Common Standard Protocol
- Trust Infrastructure of Each Provider
- Secure Code Delivery
  - of Common Standard Libraries
- Simple implementation

DID Method Authority Infrastructure
- Multiple Method Providers
- Method Specific Protocols
- Trust Infrastructure of Each Provider
  - Distributed Ledger Infrastructure
- Trust DID Resolver Infrastructure
  - Method Specific Code In Resolver
- Secure Code Delivery
  - of Multiple Method Specific Libraries

Complex implementation

Different Validation Vulnerability Spaces

Trust on intervening infrastructure
- Secure remote code execution
- Trusted execution environment
- Distributed consensus ledger
- Enduser may verify
  - OS code integrity
  - Application code integrity

Zero-trust on intervening infrastructure
- End-verifiable attestations (logs)
- Secure local code execution
- End user may verify via crypto signatures
  - Integrity and source-of-truth of attestations

Certificate Transparency Problem

“The solution the computer world has relied on for many years is to introduce into the system trusted third parties (CAs) that vouch for the binding between the domain name and the private key. The problem is that we've managed to bless several hundred of these supposedly trusted parties, any of which can vouch for any domain name. Every now and then, one of them gets it wrong, sometimes spectacularly.”

Pinning inadequate
Notaries inadequate
DNSSec inadequate
All require trust in 3rd party compute infrastructure that is inherently vulnerable

Certificate Transparency: (related EFF SSL Observatory)
Public end-verifiable append-only event log with consistency and inclusion proofs
End-verifiable duplicity detection = Ambient verifiability of duplicity
Event log is third party infrastructure but it is not trusted because it is verifiable.
Sparse Merkle Trees for revocation of certificates

Certificate Transparency Solution

Public end-verifiable append-only event log with consistency and inclusion proofs
End-verifiable duplicity detection = ambient verifiability of duplicity
Event log is third party infrastructure but it is not trusted because logs are verifiable.
Sparse Merkle trees for revocation of certificates
(related EFF SSL Observatory)

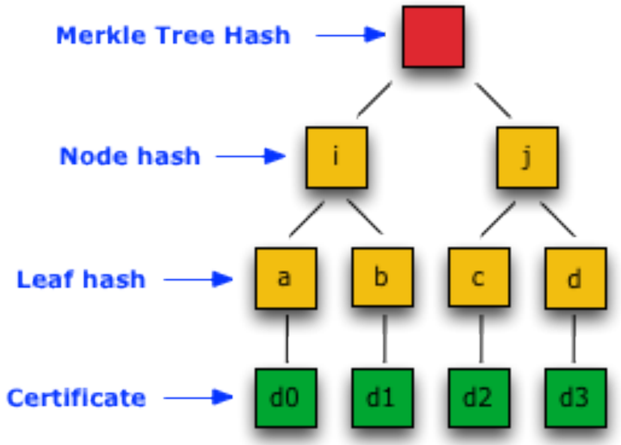


Figure 1

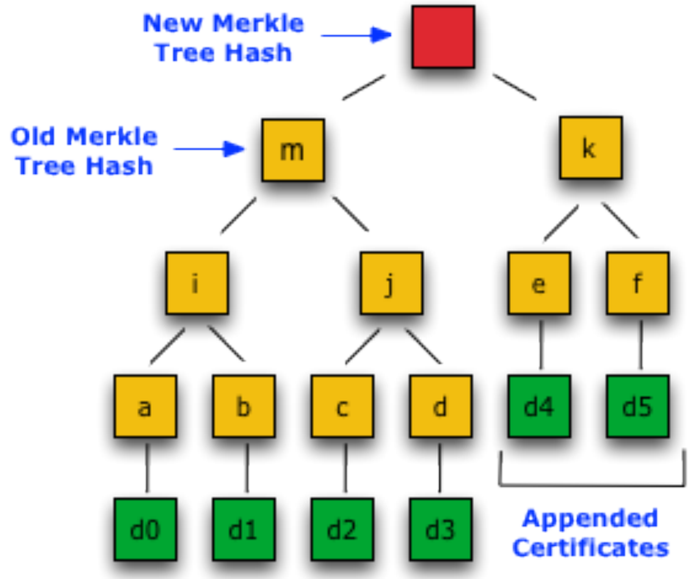


Figure 2

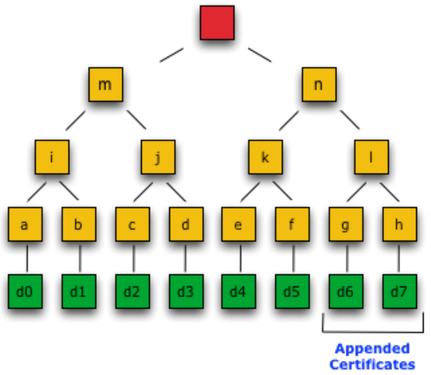


Figure 3

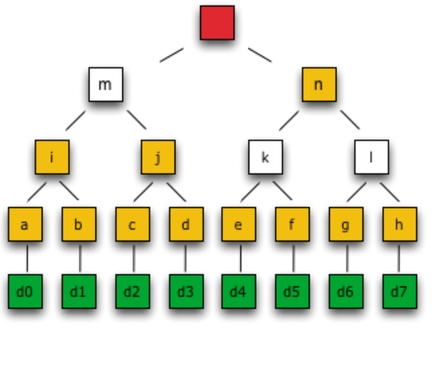


Figure 4

End-Verifiable Consistent Event Logs for DIDs

KERI & Sidetree

First seen wins
Ambient duplicity detection

Decentralized Control Authority

Asymmetric PKI: Public-Private Key Pairs for Digital Signatures

CSPRNG Cryptographic Strength Pseudo Random Number Generator

Collision Resistant Random Seed (Entropy) Available to Anyone

Seed -> Private Key -> One Way Function -> Public Key

Authority comes from collision resistance (entropy)

Controller of private key is the only one who can make verifiable signed attestations associated with the public key (identifier)

Inherently decentralized via singular sovereignty over random seed

Derived Self-Certifying Content Addressable Identifiers

`Content Hash:`

Seed -> Private Key -> One-way Function (sign scheme) -> Public Key -> +Content -> One-way Function (hash scheme) -> Derived Identifier

Seed -> Private Key -> One-way Function (sign scheme) -> Public Key -> [+Content -> One-way Function (hash scheme) -> Derived Identifier] …

Security Concepts

Roots-of-Trust (primary and secondary)

Sources-of-Truth

Loci-of-Control

“BEST” DECENTRALIZED SECURITY

End Verifiability

Ambient Verifiability = Anyone can verify, anywhere, at any time.

Ambient Duplicity Detection = Consistent Attribution

EXPLOIT MANAGEMENT

Control exploit always eventually exhibits as duplicitous behavior:

i.e. inconsistent control statements

Duplicity detection provides exploit detection and containment.

Duplicity detection via immutable logs of control statements is scalable and performant

End verifiability removes trust on intervening infra-structure

Control exploit potential mitigated via best practices for key management

Cryptographic Root-of-Trust

Trust *who said it* not *what was said*

Consistent attribution is the root-of-trust (integral non-repudiable statements)

Duplicity detection

I trust that controller of private key made a set of statements

Build trust in *what was said* via consistently attributable (who) histories of consistent statements (what). (i.e. reputation)

Loci-of-Control

Who controls What

Control over Identifier

Control over Operations on Identifier

Control over Support Infrastructure to establish and maintain control over identifier

Source-of-Truth

The Controller is the source-of-truth for control over the Identifier

Only the the controller my make verifiably authoritative control statements:

- Sequencing (of authoritative statements)
- Dependency (of authoritative statements)

Authoritative Statements:

- Transfer of control (non-revokable)
- Delegation of control (revokable)
- Authorizations:
  - Additional roots-of-trust and sources-of-truth

Autonomic Identifier

Unified Sovereignty (control) over

- Root-of-trust
- Source-of-truth
- Loci-of-control

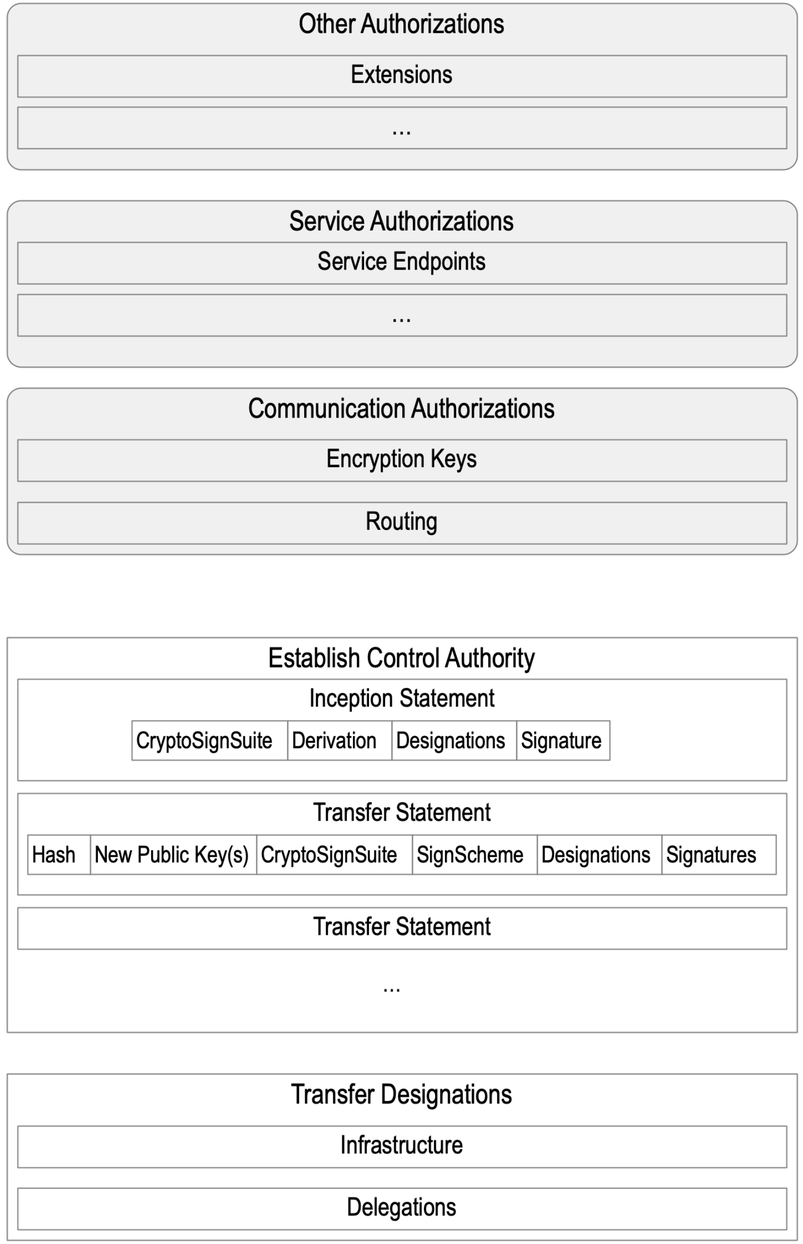
Identifier Features

Ephemeral (Non-Rotatable) vs. Persistent (Rotatable)

Public vs Private

Offline vs Online Establishment

Control Establishment Hierarchy



Methods for Making Other Authorizations
Given Control Authority Established

Methods for Making Common Authorizations
Given Control Authority Established

Methods for Establishing Control Authority

Background

AUTONOMIC IDENTIFIER (AID)
AND
AUTONOMIC NAMESPACE (AN)

auto nomos self rule (autonomic, autonomous)

Self-Governing, Self-Sovereign, Self-Controlling

Self-Certifying,Self-Authorizing,

Self-Managing, Self-Administering

Cryptographically Verifiable Identifier Derivation

Prefixed with AID

Related Works

Smith, S. M., “Key Event Receipt Infrastructure (KERI) Design and Build”, arXiv, 2019/07/03

https://arxiv.org/abs/1907.02143

Smith, S. M., “Open Reputation Framework,” vol. Version 1.2, 2015/05/13

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/open-reputation-low-level-whitepaper.pdf

Smith, S. M. and Khovratovich, D., “Identity System Essentials,” 2016/03/29

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/Identity-System-Essentials.pdf

Smith, S. M., “Decentralized Autonomic Data (DAD) and the three R’s of Key Management,” Rebooting the Web of Trust RWOT 6, Spring 2018

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/DecentralizedAutonomicData.pdf

Conway, S., Hughes, A., Ma, M. et al., “A DID for Everything,” Rebooting the Web of Trust RWOT 7, 2018/09/26

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/A_DID_for_everything.pdf

Shae, M., Smith, S. M. and Stocker, C., “Decentralized Identity as a Meta-platform: How Cooperation Beats Aggregation,” Rebooting the Web of Trust, vol. RWOT 9, 2019/11/19

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/CooperationBeatsAggregation.pdf

Self-Certifying Identifier/Namespace
Decentralized Cryptographic Root-of-Trust

Use public key in identifier or as prefix in namespace

Signed statements that include self-certifying identifier

Self-Certifying namespace

End Verifiable to Primary Root-of-Trust

Other roots of trust may add to but not replace self-certification

All decentralized infrastructure has self-certifying identifiers as primary root-of-trust

Decentralized key management is therefore essential to protecting infrastructure

Enables provenanced chain of transformations with verifiable control over transformations.

Authoritative control via signing not encryption keys.

Human Basis of Trust

I know you

or

I know of you

therefore

I trust you

On the internet I can't really know who you are.
therefore
I can't trust you

KEY MANAGEMENT

Rotation = Revoke and Replace = Full Transfer of Control

Reproduction = Creation, Derivation

Recovery

BEST PRACTICE

One-use = One-time, One-place, One-way

Key Rotation = Transfer of Control

Key rotation is useful when the controller of a self-certifying identifier needs to maintain persistent control over that identifier indefinitely despite exploits of the private key(s).

Otherwise in the event of exploit, the controller could just abandon the exploited identifier and create a new identifier with a new (public, private) key pair.

Non-rotatable Authoritative Signing Key(s) (trivial)

Rotatable Authoritative Signing Key(s) (difficult)

More Key Rotation

After rotation the public key associated with the identifier is not changed, but merely the authoritative private key(s) is changed.

Otherwise the identifier loses its value as an identifier.

After rotation the original private key has been revoked and replaced with the new private key(s) specified in the rotation operation.

The initial rotation operation must be signed at the very least with the original private key.

The latest rotation operation provides the current authoritative key pair.

After rotation, the new public key is appended to the identifier's key rotation history.

In order to verify an attestation affiliated with an identifier, the verifier must lookup and validate the key rotation history for that identifier to determine if the attestation signature was made with the authoritative key(s) at the time of signing.

Pre-rotation

Inception -> Rotation -> Rotation -> …

Inception event:

1) Establish original self-certifying key

2) Commit to next set of key(s)

Each rotation event:

1) Execute rotation operation to new key(s)

2) Commit to next set of key(s)

By default Rotation signed by both previous (current) and new set of key(s)

(Joint Rotation Approval)

Compromise of an unexposed set of keys required for exploit.

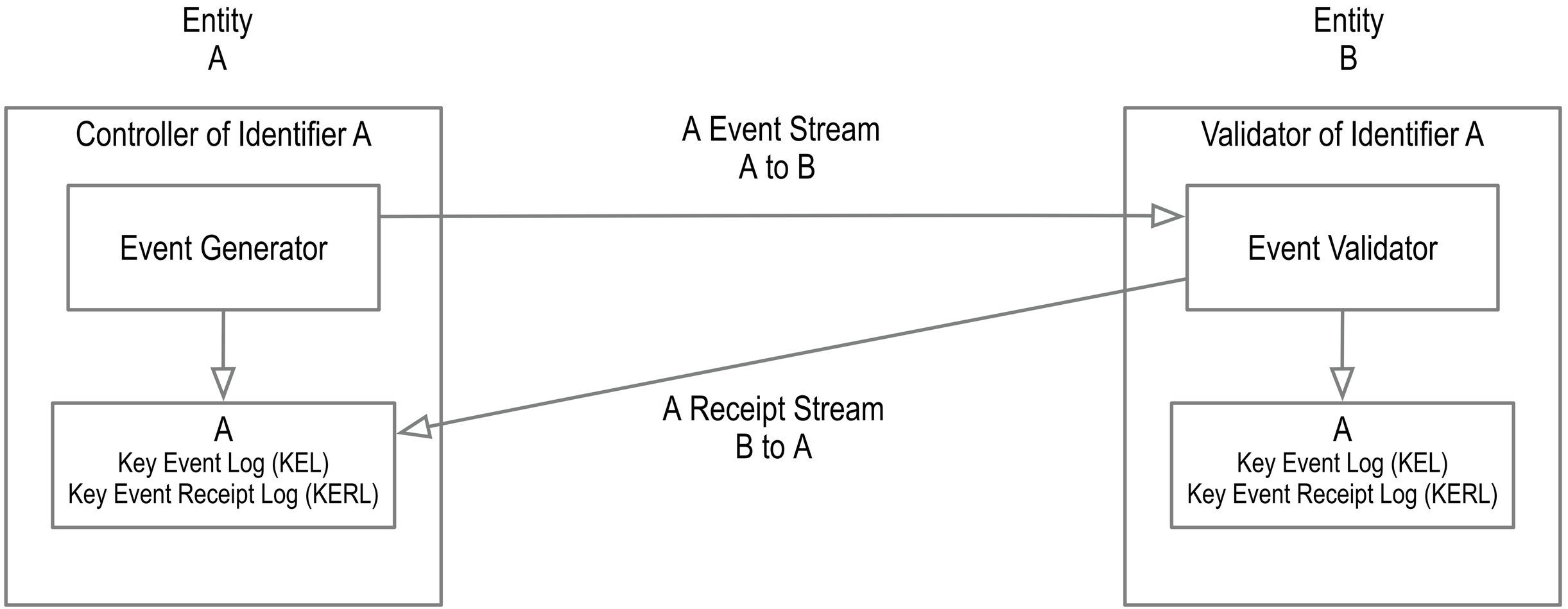
(with joint approval compromise of two sets of keys required, one unexposed)

Two Use Cases

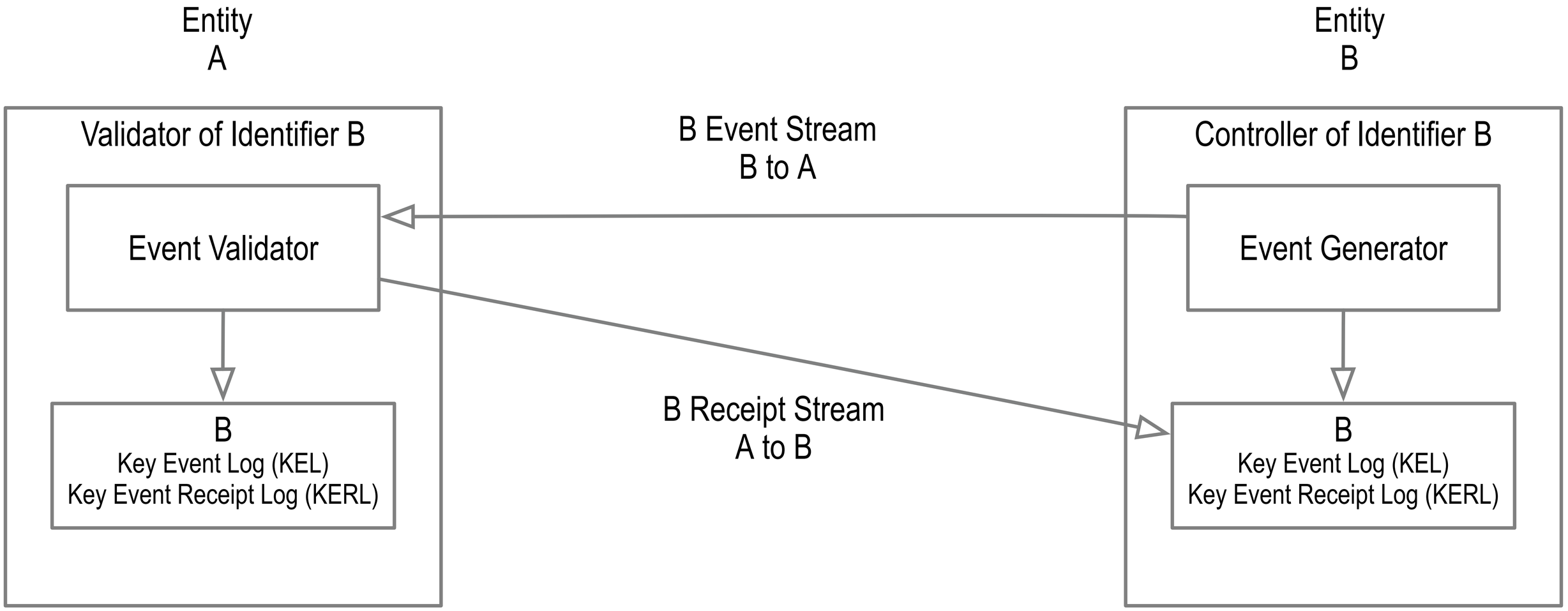
Online haltable establishment (one-to-one, pair-wise)

Offline non-haltable establishment (one-to-many, any-wise, N-wise)

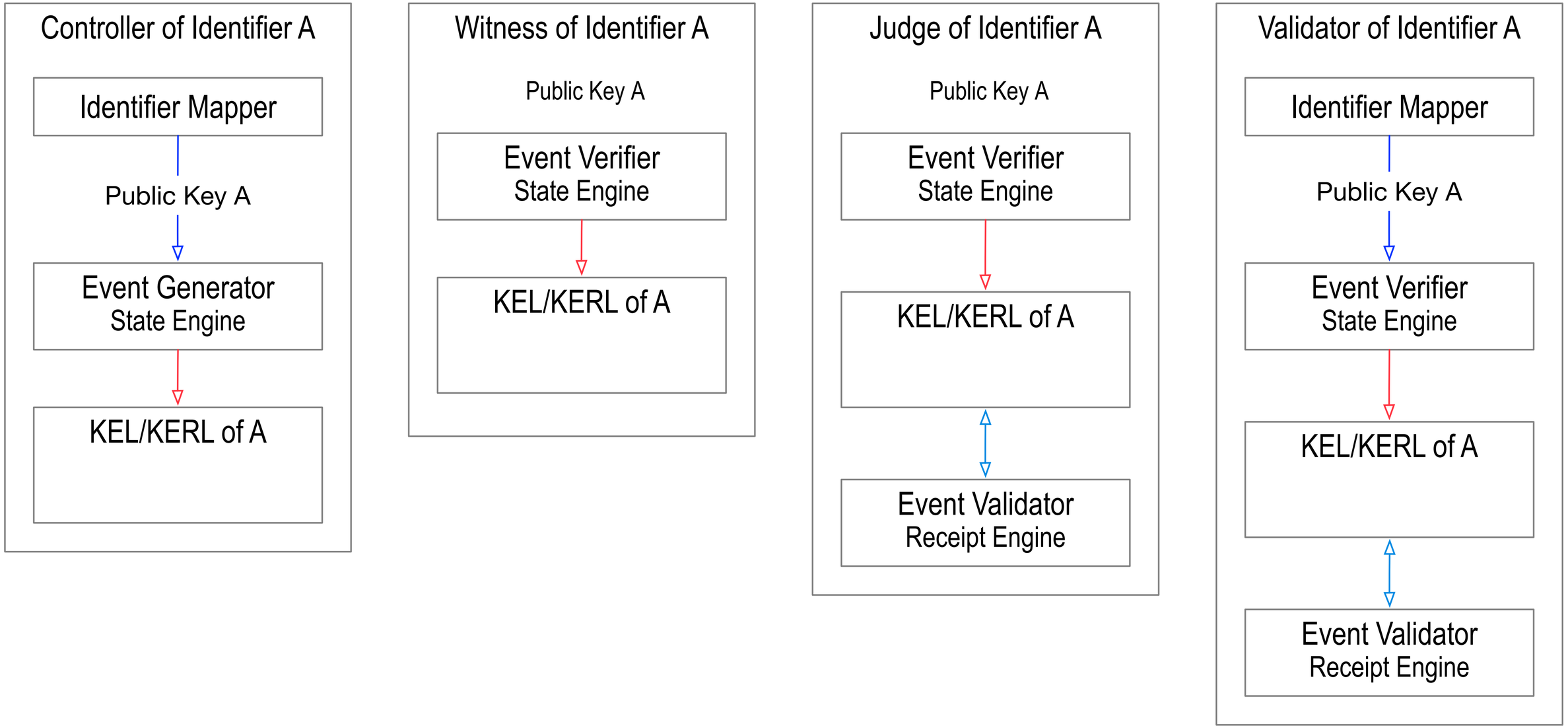
Online or One-to-one or Pair-wise: A to B



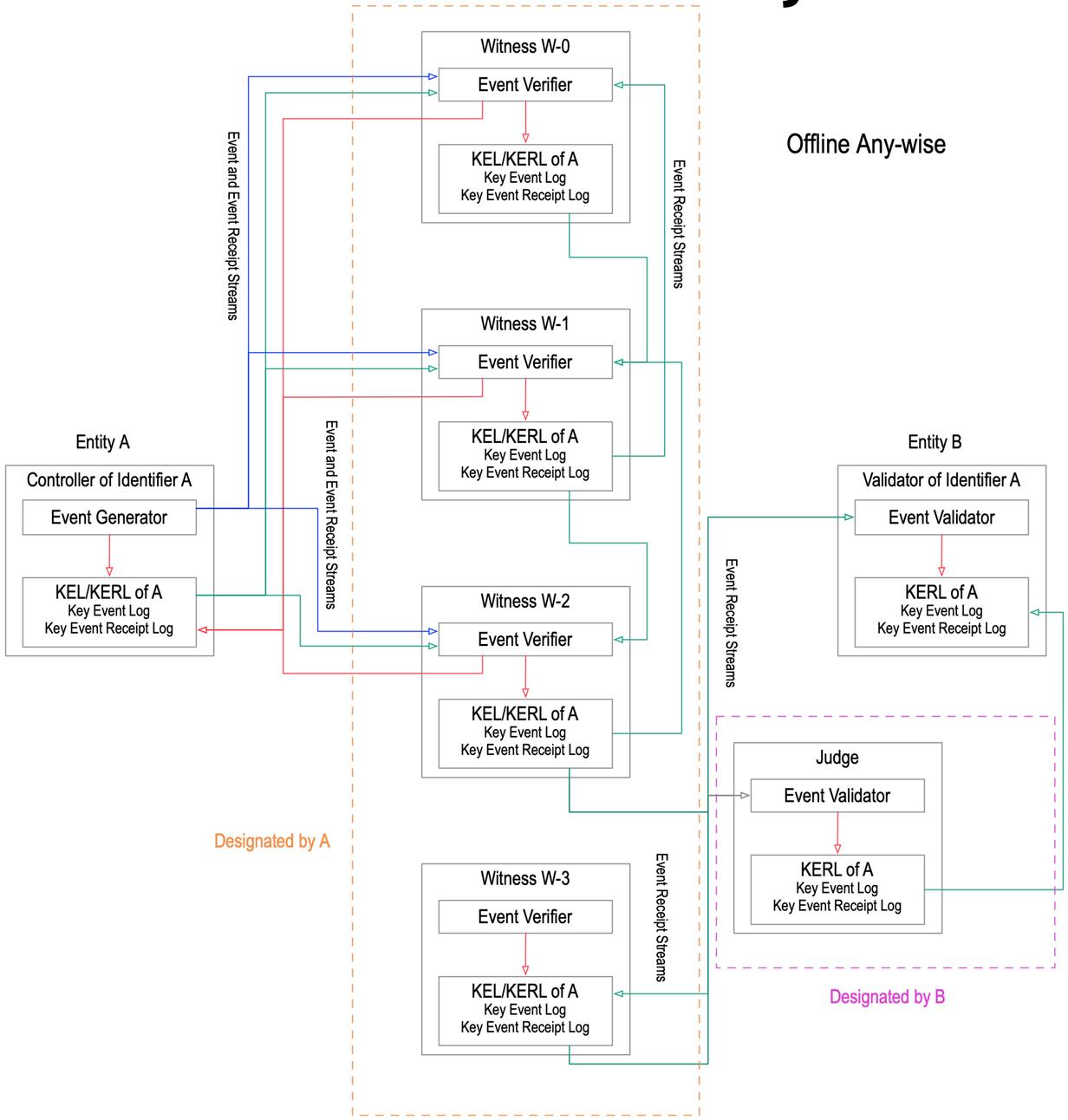
Online or One-to-one or Pair-wise: B to A



Offline Components



Offline or One-to-many or Any-wise



Offline Trustworthy Service

Service provided by *N* designated *witnesses* and at least one *judge*.

Service provides a *correct* or at least a *complete* KERL to any validator that requests it.

Service availability means at least one *correct* or *complete* KERL is extant

Live duplicity detection with one honest witness.

Dead duplicity detection with one complete KERL.

Key state validation:

- does not need total ordered distributed consensus (costly)
- does benefit from distributed consistency (scalable)

Complete or Correct

Witnesses only signs and stores first received and verified version of an event at a given location in full event sequence.

Witnesses provide an immutable KERL of all receipts from any witnesses for the version of each event in its log.

M of N Designated Witnesses

Complete if KERL has M+1 receipts for event

Correct if M is a majority of N

Exploit Recovery

Pre-rotation protects against live exploit

Recover by rotation to new authoritative key(s)

Witnessed event log protects against dead exploit

Single honest witness enables duplicity detection

Historical receipts prove accepted history

Variants

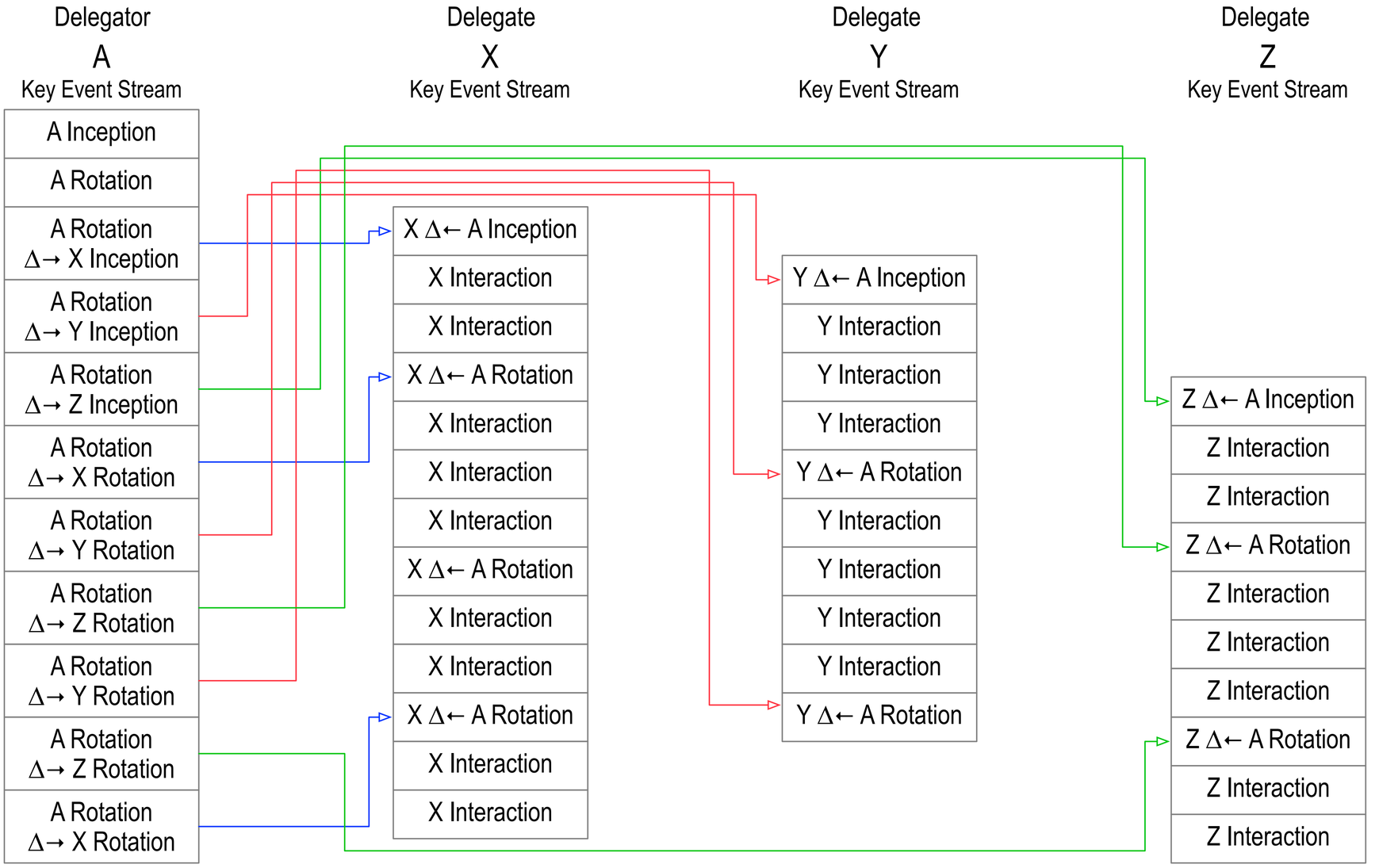
Single Key Sequence for Rotation and Signing

Single Interleaved (Doubled) Key Sequence for Rotation and Signing

Single Key Sequence for Rotation with multiple Delegated Key Sequence(s) for Signing

Dual Key Sequence for Separate Rotation and Signing

Delegation



Δ→ X : Delegation to X

Δ← A : Delegation from A

KEY Event Based Provenance of Identifiers

KERI events enable cryptographic Proof-of-Authority or Proof-of-Provenance for every identifier.

Proof is copy of key event receipt log (KERL)

KEY (Identifier) Role Support

Controlling Key (Identifier) Roles via Authoritative Delegation:

- Signing Keys
- Rotatable (optional)
- Fully verifiable to root-of-trust using KERI

Non-Controlling Key (Identifier) Roles via Authoritative Attestation:

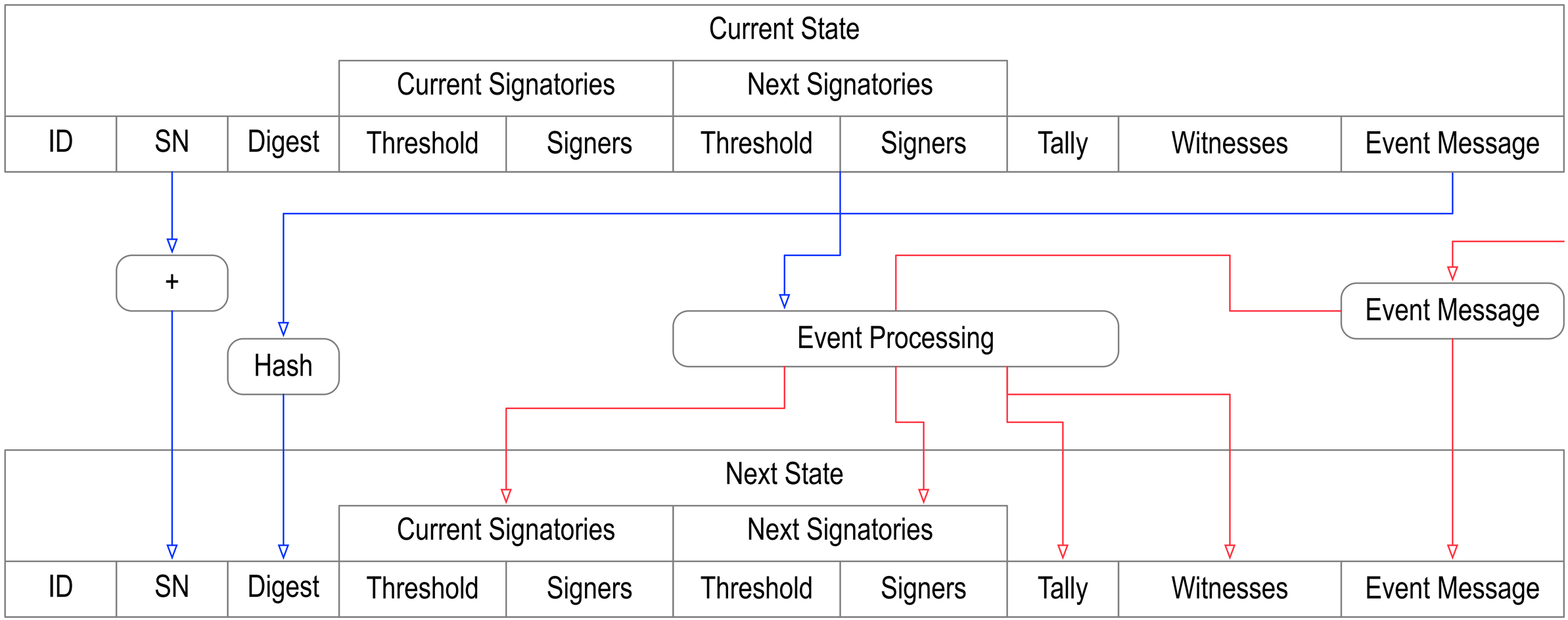
- Encryption, Sign-cryption, etc
- Always Non-Rotatable
- Fully verifiable to authoritative controlling key(s) at point of attestation
- Granular role establishment via events

Generalized Discovery Support

Non-Controlling Identifier Discovery Roles via Authoritative Attestation:

- Service Endpoints, Documentation, Etc.
- Always Non-Rotatable
- Fully verifiable to authoritative controlling key(s) at point of attestation
- Granular role establishment via events

State Verifier Engine



Generic Inception

$$\varepsilon_0 = \left\langle C, t_0, \texttt{icp}, C^0, K_1, \widehat{C}_1, M_0, \widehat{W}_0 \right\rangle \sigma_{C^0}$$

$$\widehat{C}_1 = \left[C^1, \ldots, C^{L_1} \right]_1$$

$$\widehat{W}_0 = \left[W_0, \ldots, W_{N-1} \right]_0$$

Generic Rotation

$$\varepsilon_k = \left\langle C, t_k, \eta\left(\varepsilon_{k-1}\right), \texttt{rot}, K_{l+1}, \widehat{C}_{l+1}, M_l, \widehat{X}_l, \widehat{Y}_l, \{data\}, \widehat{e}_l, \widehat{s}_l \right\rangle \widehat{\sigma}_l$$

$$\widehat{C}_{l+1} = \left[C^{r_{l+1}}, \ldots, C^{r_{l+1}+L_{l+1}-1} \right]_{l+1}$$

$$\widehat{X}_l = \left[X_0, \ldots, X_{O_l-1} \right]_l$$

$$\widehat{Y}_l = \left[Y_0, \ldots, Y_{P_l-1} \right]_l$$

$$\widehat{e}_l = \left[e_0, \ldots, e_{E_l-1} \right]_l$$

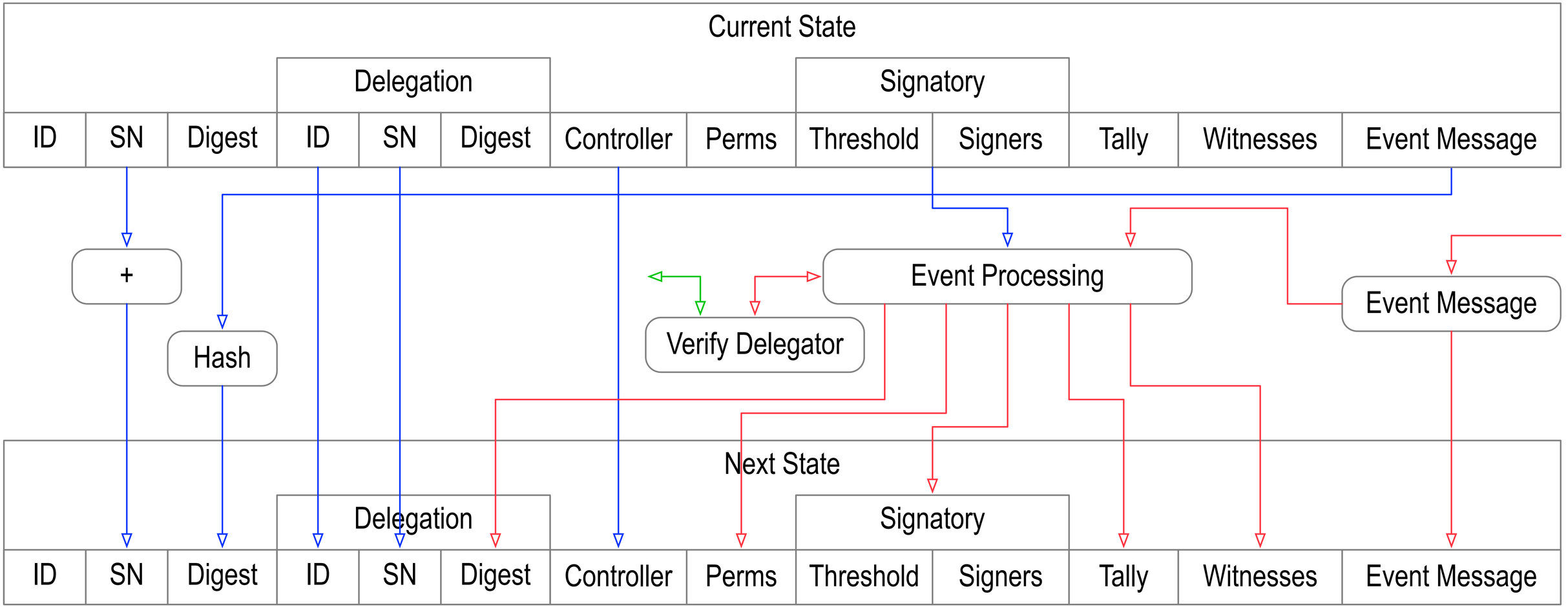
$$\widehat{s}_l = \left[s_0, \ldots, s_{S_l-1} \right]_l$$

$$\widehat{\sigma}_l = \sigma_{C^{e0+\eta-1}} \ldots \sigma_{C^{eE_l-1+\eta-1}} \sigma_{C^{s0+\eta}} \ldots \sigma_{C^{sS_l-1+\eta}}$$

Interaction Event

$$\varepsilon_k = C_{\varepsilon_k} = \left\langle C, t_k, \eta\left(\varepsilon_{k-1}\right), \texttt{itc}, \{data\}, \widehat{s}_k \right\rangle \widehat{\sigma}_k$$

Delegated State Verifier Engine



Delegating Inception

$$\varepsilon_k = \left\langle C, t_k, \eta\left(\varepsilon_{k-1}\right), \texttt{rot}, K_{l+1}, \widehat{C}_{l+1}, M_l, \widehat{X}_l, \widehat{Y}_l, \widehat{\Delta}_0^D, \widehat{e}_l, \widehat{s}_l \right\rangle \widehat{\sigma}_l$$

$$\widehat{\Delta}_0^D = \left\{ D, \texttt{dip}, D^0, perms, K_0^D, \widehat{D}_0^D, M_0^D, \widehat{W}_0^D \right\}$$

$$\varepsilon_0 = D_{\varepsilon_0} = \left\langle D, t_0, \texttt{dip}, \widehat{\Delta}_k^C, D^0, perms, K_0^D, \widehat{D}_0^D, M_0^D, \widehat{W}_0^D \right\rangle \sigma_{D^0}$$

$$\widehat{\Delta}_k^C = \left\{ C, t_k^C, \eta\left(\varepsilon_k^C\right) \right\}$$

Delegating Rotation

$$\varepsilon_k = \left\langle C, t_k, \eta\left(\varepsilon_{k-1}\right), \texttt{rot}, K_{l+1}, \widehat{C}_{l+1}, M_l, \widehat{X}_l, \widehat{Y}_l, \widehat{\Delta}_e^D, \widehat{e}_l, \widehat{s}_l \right\rangle \widehat{\sigma}_l$$

$$\widehat{\Delta}_e^D = \left\{ D, \texttt{drt}, perms, K_e^D, \widehat{D}_e^D, M_e^D, \widehat{X}_e^D, \widehat{Y}_e^D \right\}$$

$$\varepsilon_d = D_{\varepsilon_d} = \left\langle D, t_d, \eta\left(\varepsilon_{d-1}\right), \texttt{drt}, \widehat{\Delta}_k^C, perms, K_e^D, \widehat{D}_e^D, M_e^D, \widehat{X}_e^D, \widehat{Y}_e^D \right\rangle \sigma_{D^0}$$

$$\widehat{\Delta}_k^C = \left\{ C, t_k^C, \eta\left(\varepsilon_k^C\right) \right\}$$

Delegated Interaction

$$\varepsilon_d = D_{\varepsilon_d} = \left\langle D, t_d, \eta\left(\varepsilon_{d-1}\right), \texttt{itc}, data, \widehat{s}_d^{\,D} \right\rangle \widehat{\sigma}_d$$

KERI Nomenclature

self-certifying identifier: includes public key

digital signature: unique non-repudiable (cypher suite known)

digest: collision resistant hash of content

signed digest: commitment to content

controller: controlling entity of identifier

message: serialized data structure

event: actionable message

key event: key management operation

More KERI Nomenclature

inception event: unique self-signed event that creates identifier and controlling key(s)

rotation event: self-signed uniquely ordered event from a sequence that changes the set of controlling keys

verifier: cryptographically verifies signature(s) on an event message.

witness: entity that may receive, verify, and store key events for an identifier. Each witness controls its own identifier used to sign key event messages, *controller* is a special case of witness.

receipt: event message or reference with one or more witness signatures

Even More KERI Nomenclature

key event log: ordered record of all self-signed key event messages

key event receipt log: ordered record of all key event receipts for a given set of witnesses

validator: determines current authoritative key set for identifier from at least one key event (receipt) log.

judge: determines current authoritative key set for identifier from the key event receipt logs from a set of witnesses.

pre-rotation: commitment to next rotated key set in previous rotation or inception event

BACKGROUND