

Key Event Receipt Infrastructure

KERI-2

A Secure Identifier Overlay for the Internet

Samuel M. Smith Ph.D.

<https://github.com/SmithSamuelM/Papers>

https://github.com/SmithSamuelM/Papers/blob/master/presentations/KERI2_Overview_AB_IIW_2020.pdf

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/KERI_WP_2.x.web.pdf

sam@prosapien.com

IIW Spring 2020 April 28

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

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

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

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., "Self-certifying File System," MIT Ph.D. Dissertation, 2000/06/01

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

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>

TCG, "Implicit Identity Based Device Attestation," Trusted Computing Group, vol. Version 1.0, 2018/03/05

<https://trustedcomputinggroup.org/wp-content/uploads/TCG-DICE-Arch-Implicit-Identity-Based-Device-Attestation-v1-rev93.pdf>

Smith, S. M., "Key Event Receipt Infrastructure (KERI) Design and Build", arXiv, 2019/07/03 revised 2020/04/23

<https://arxiv.org/abs/1907.02143>

Smith, S. M., "Key Event Receipt Infrastructure (KERI) Design", 2020/04/22

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/KERI_WP_2.x.web.pdf

Certificate Transparency:

Laurie, B., "Certificate Transparency: Public, verifiable, 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>

Human Basis-of-Trust “in person”

I can know you – therefore I can trust you



“on the internet”

I can't really know you – therefore I can't really trust you

Replace human *basis-of-trust* with cryptographic *root-of-trust*.

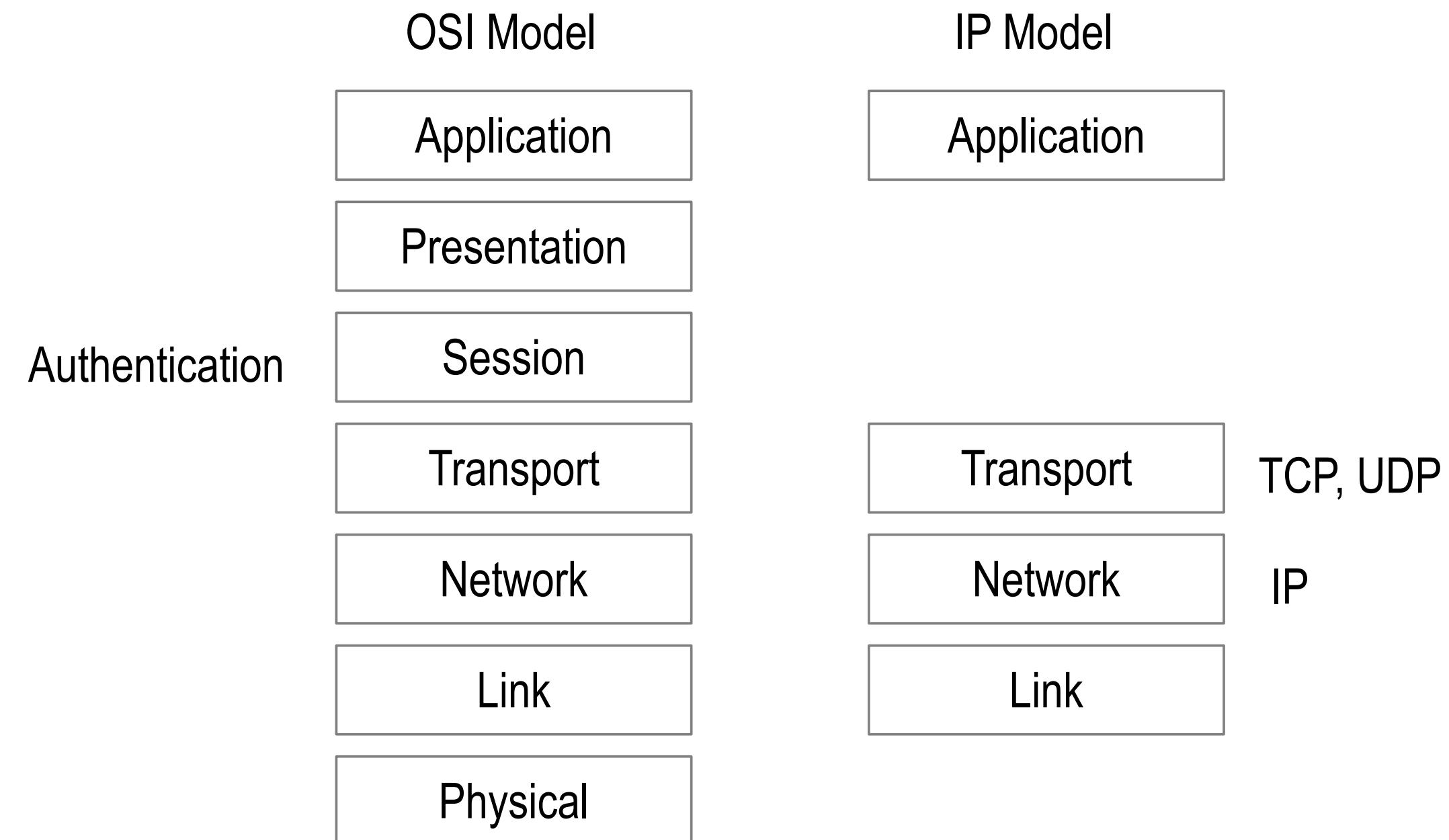
With verifiable digital signatures from asymmetric key crypto –
we may not trust in “**what**” was said, but we may trust in “**who**” said it.

We may verify that the **controller** of a private key, (the **who**), made a statement
but not the validity of the statement itself.

The root-of-trust is **consistent attribution** via verifiable integral non-repudiable statements

We may build trust over time in **what** was said via histories
of verifiably attributable (to **whom**) consistent statements i.e. **reputation**.

The Internet Protocol (IP) is *bro-ken* because it has no security layer.

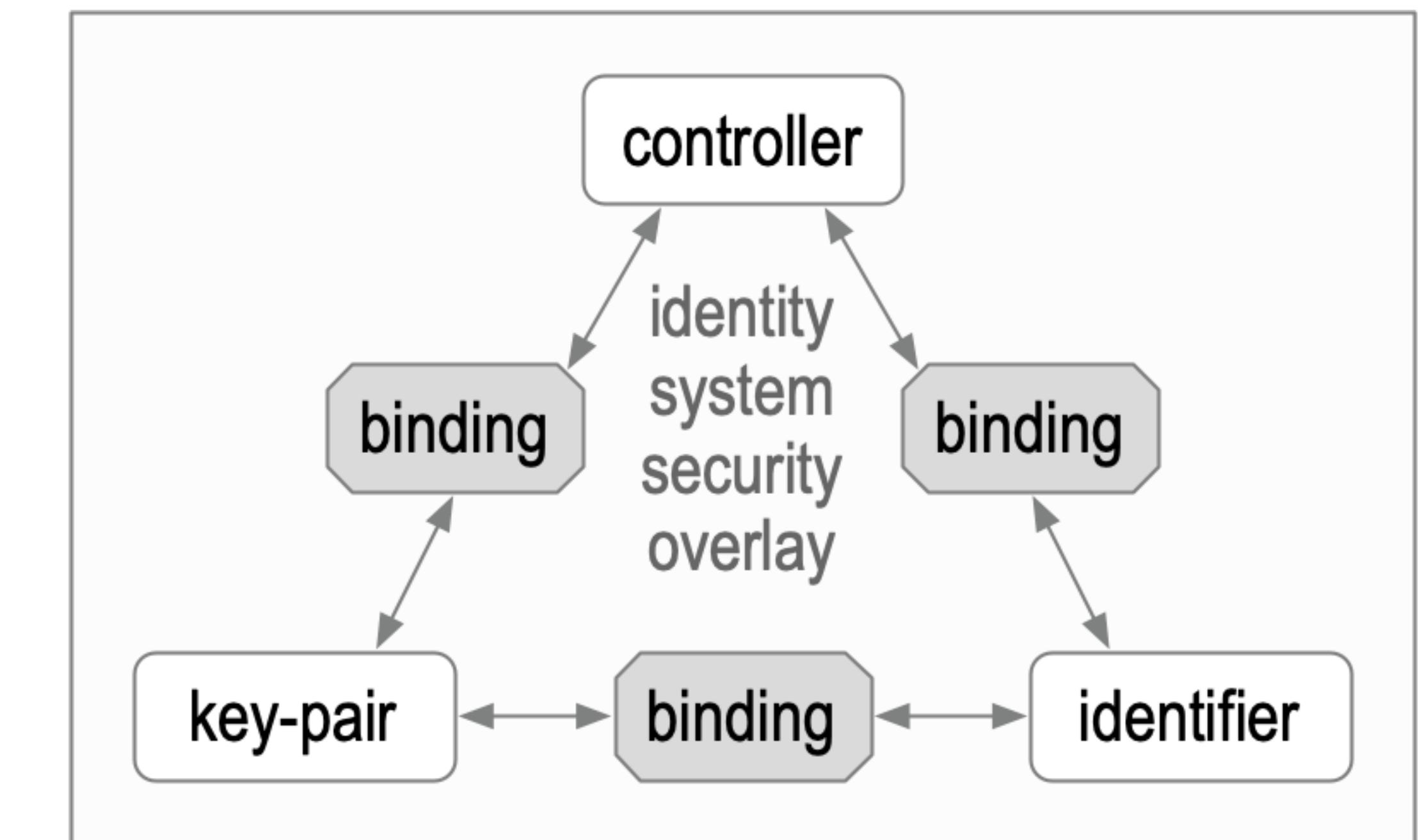
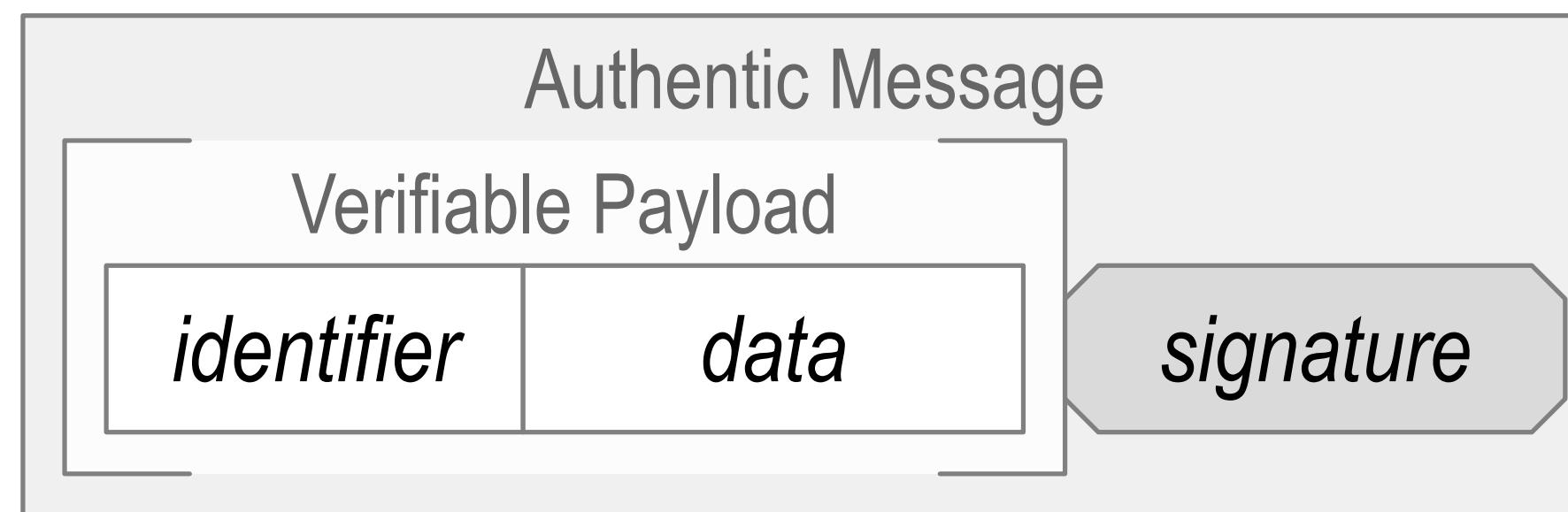
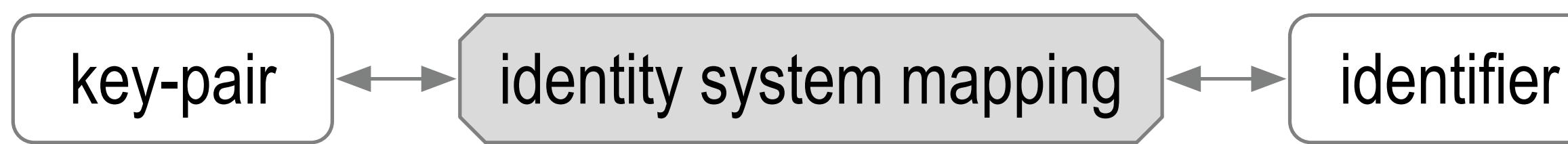


Instead ...

We use *bolt-on* identity system security overlays.
(DNS-CA ...)

Identity System Security Overlay

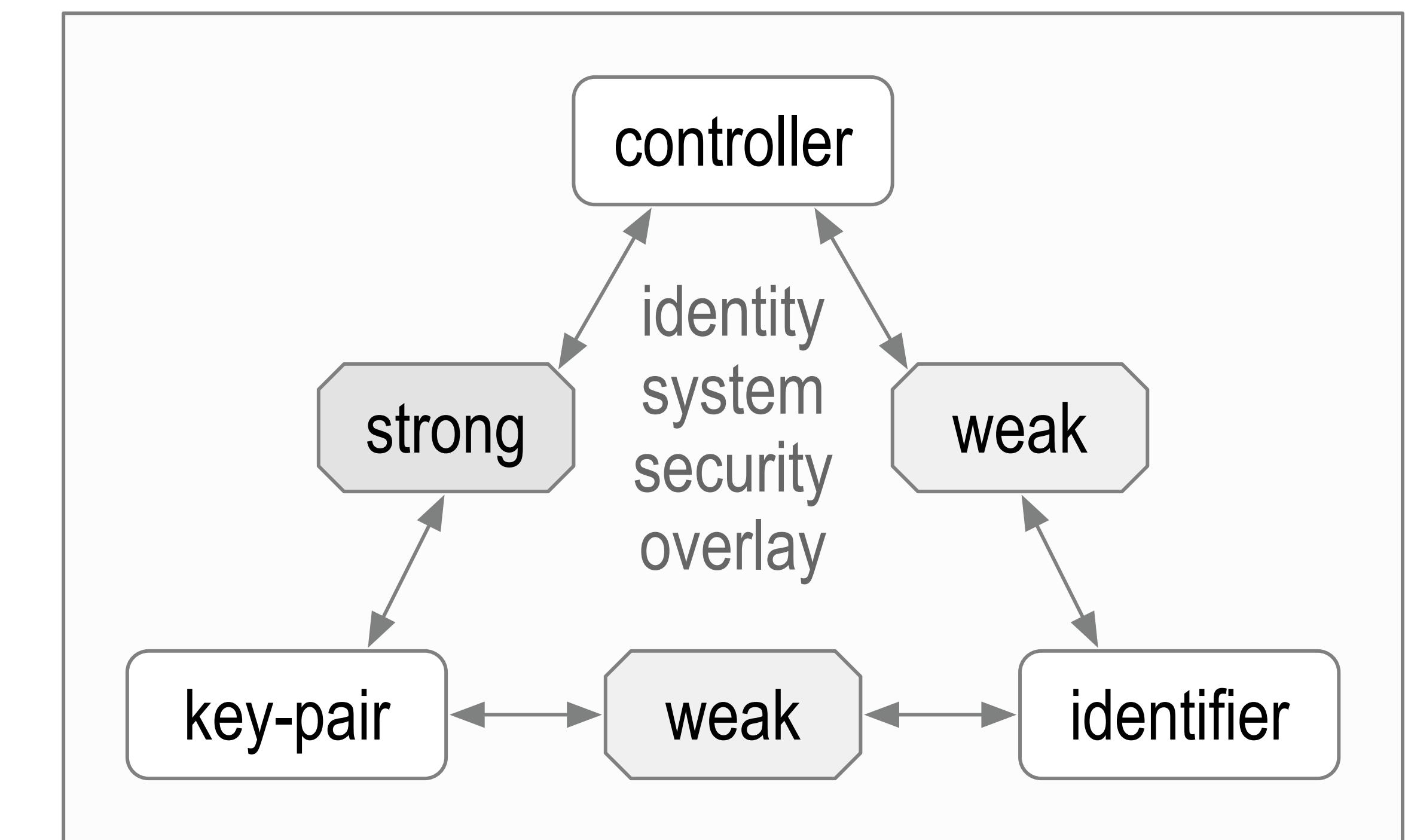
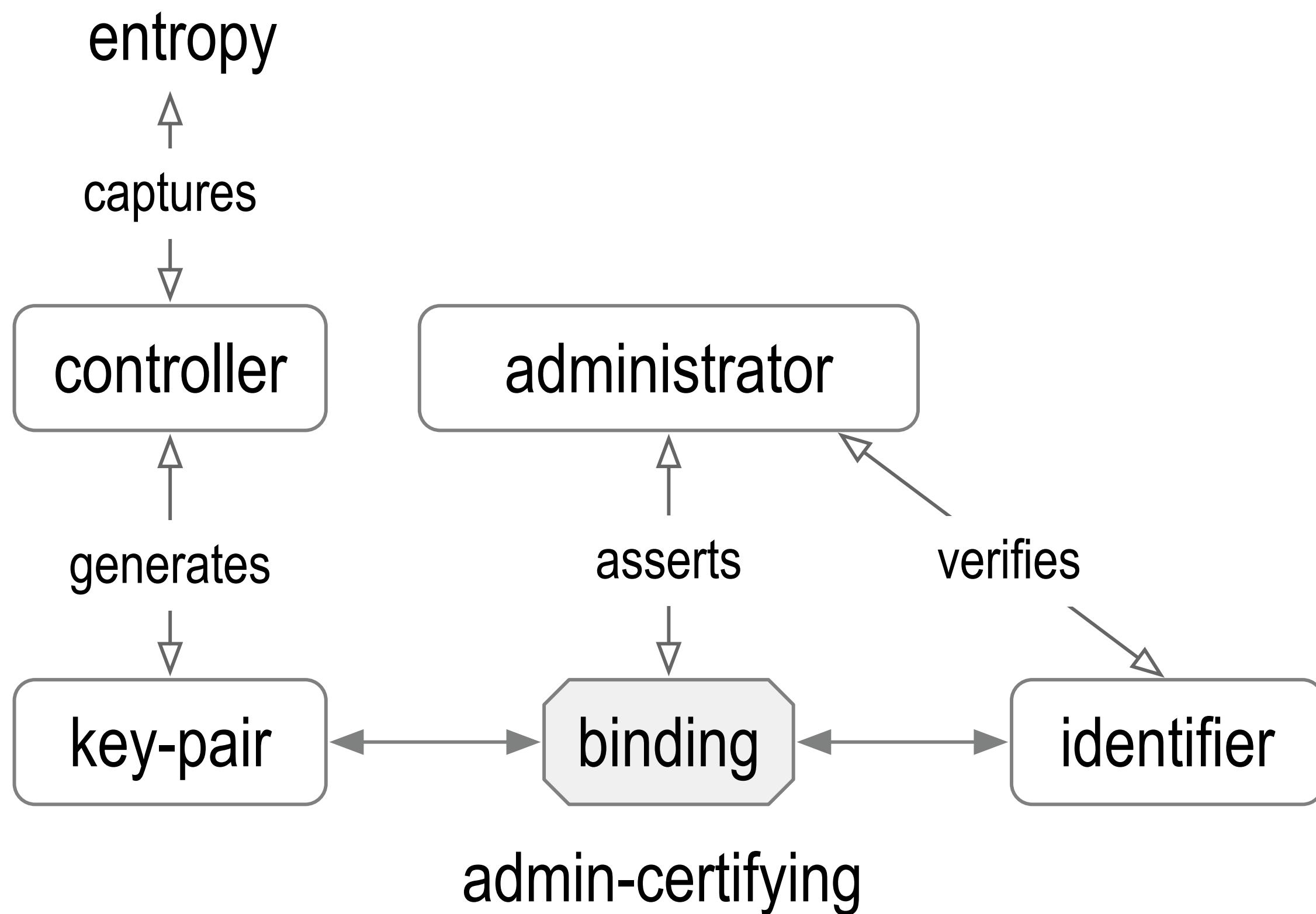
Establish authenticity of IP packet's message payload.



Identifier Issuance

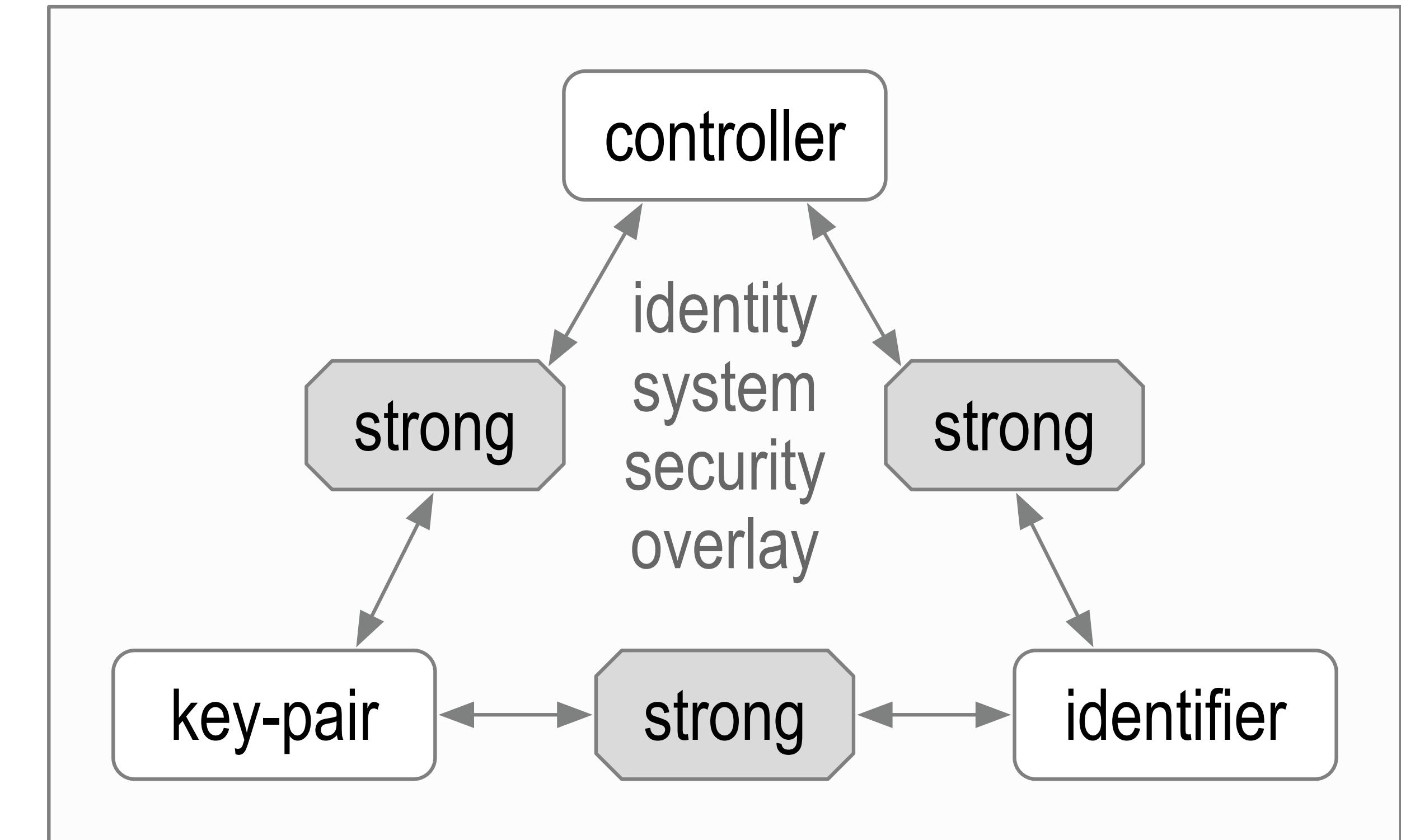
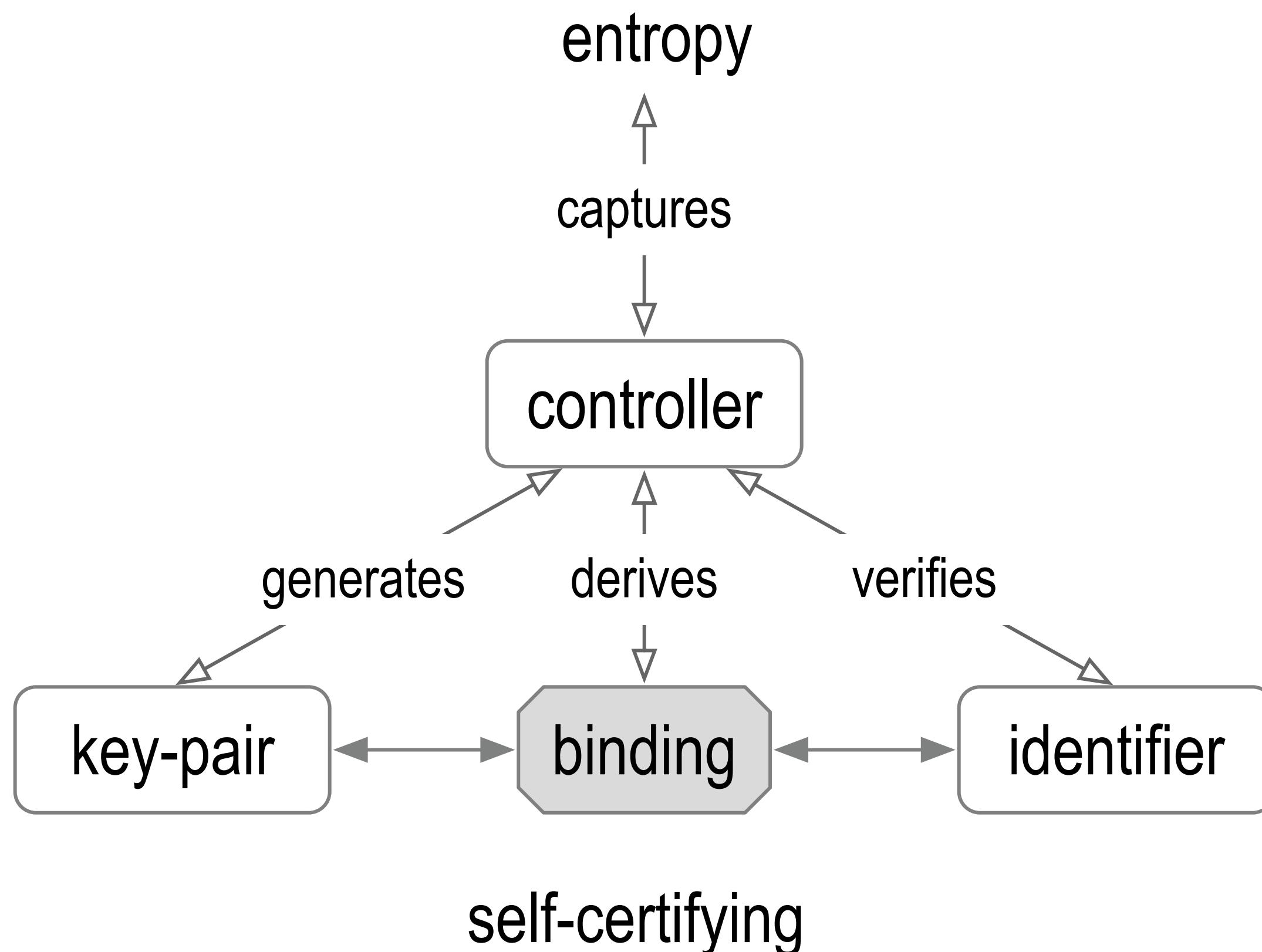
The overlay's security is contingent on the mapping's security.

Administrative Identifier Issuance and Binding



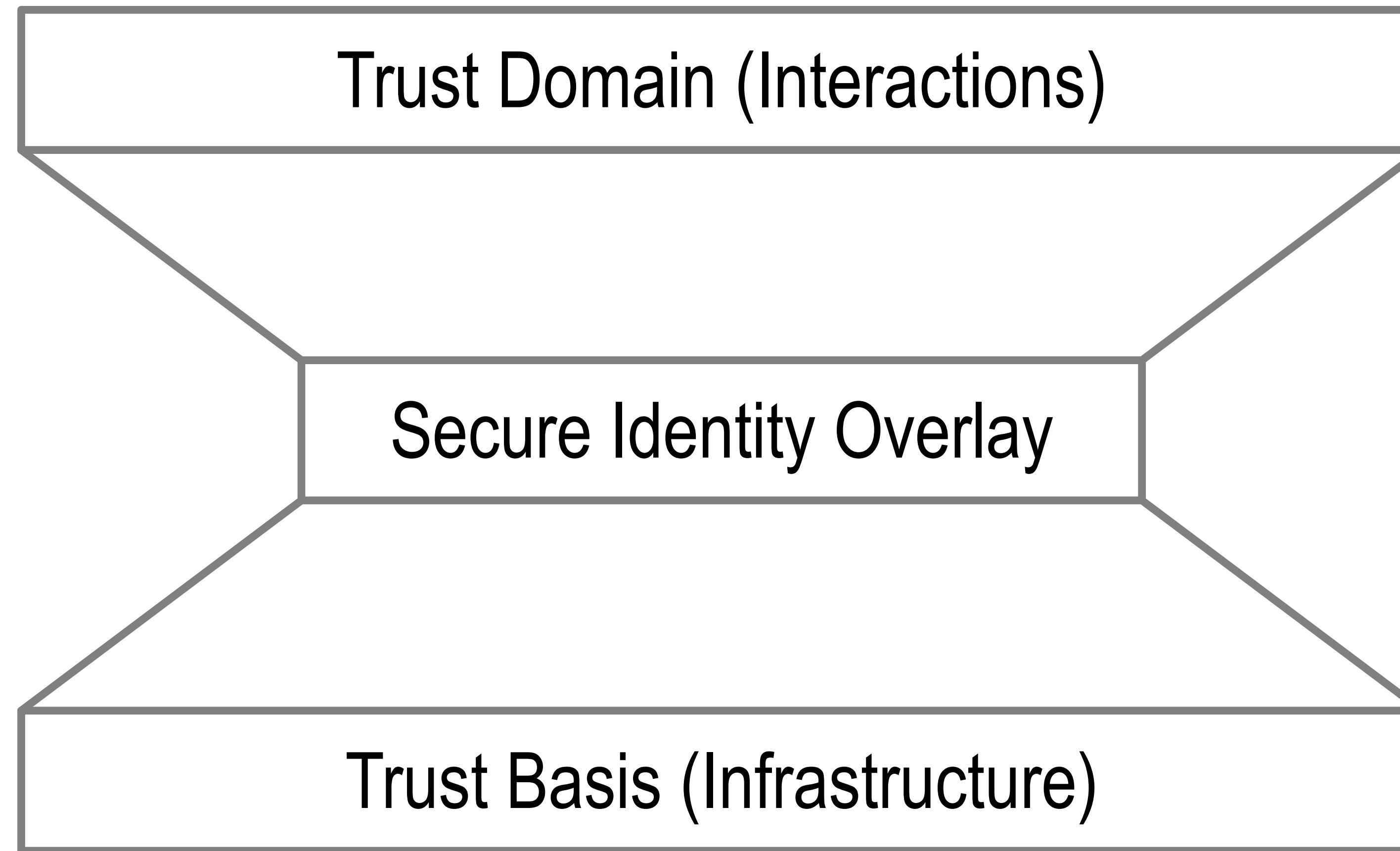
Admin-Certifying Identifier Issuance

Self-Certifying Identifier Issuance and Binding

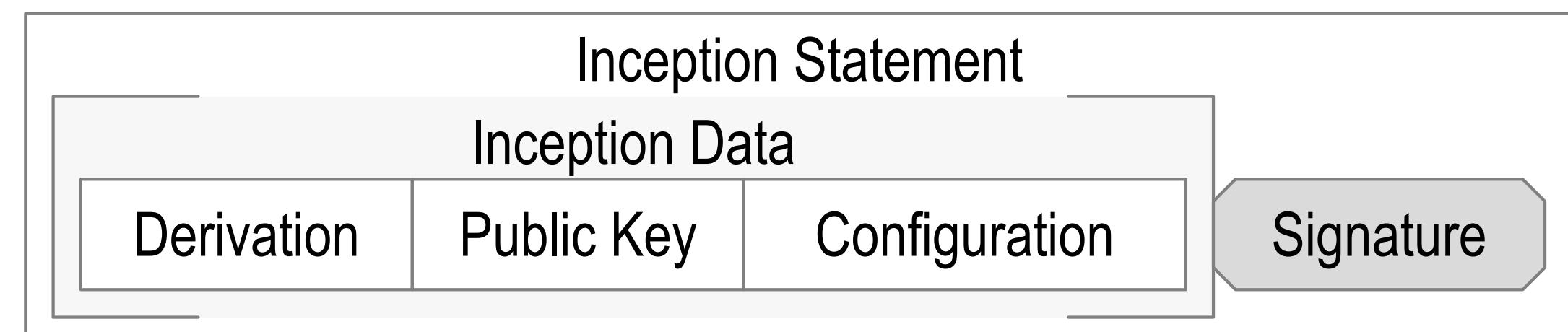
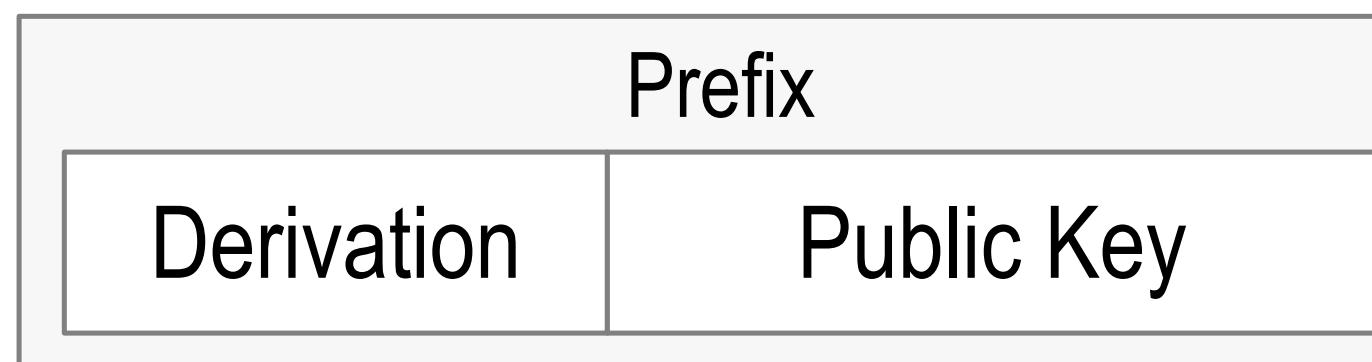
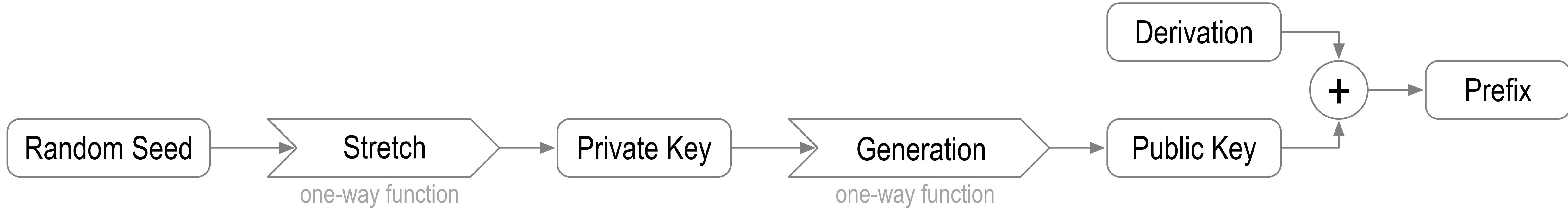


Self-Certifying Identifier Issuance

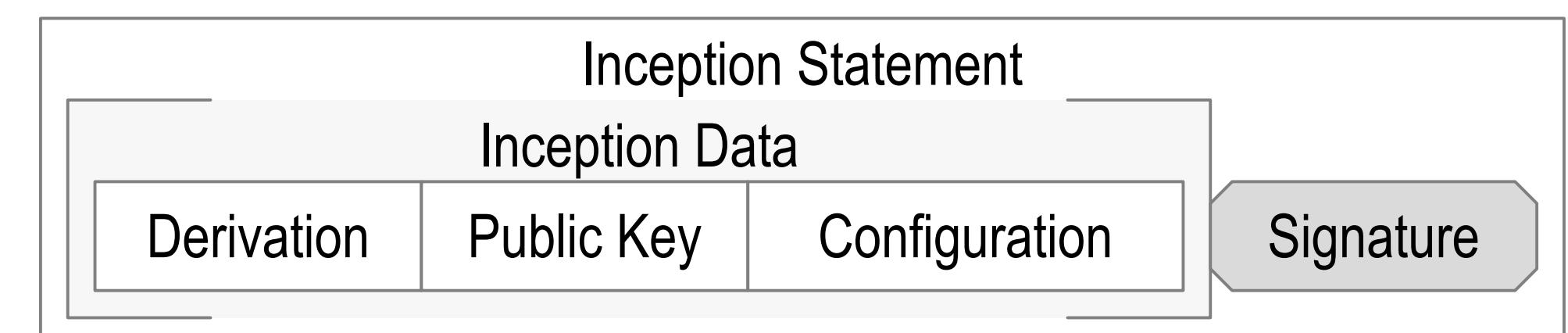
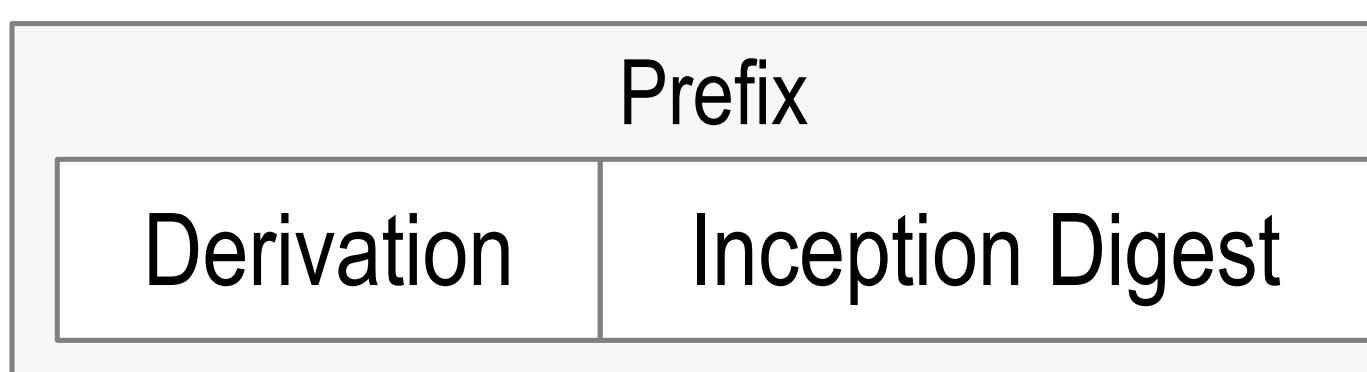
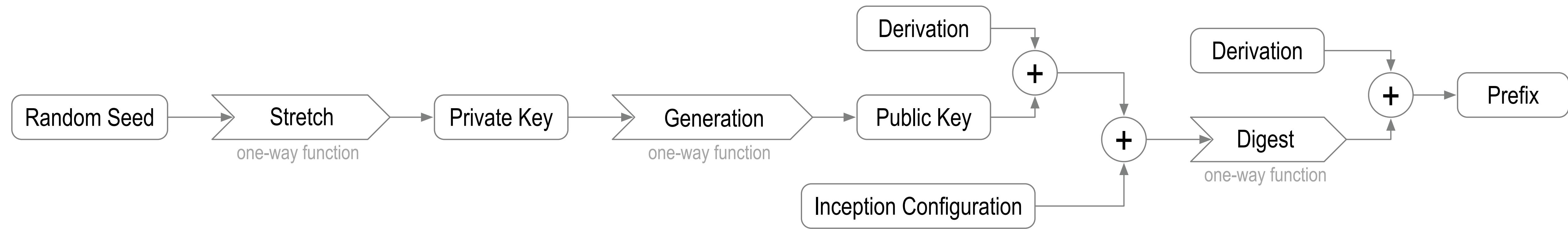
Identity System Security Overlay



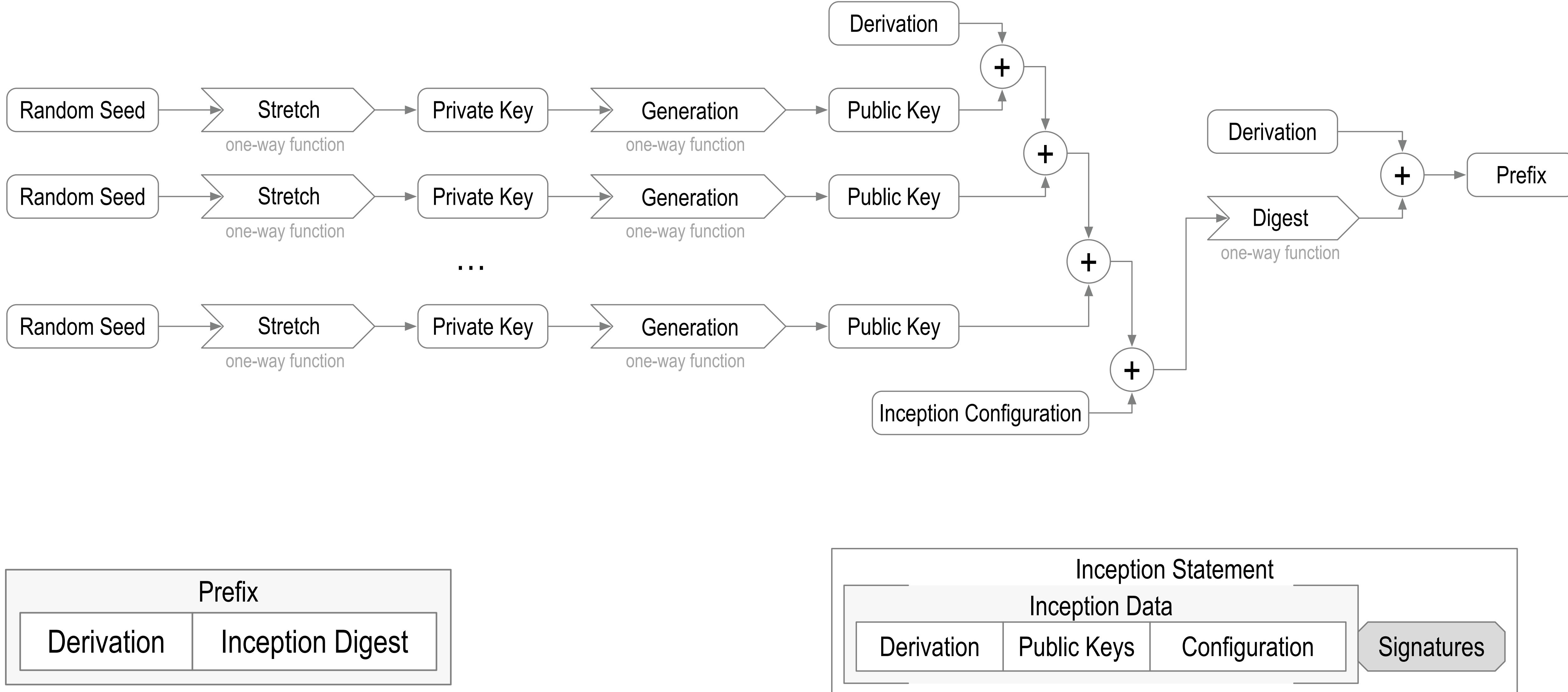
Basic



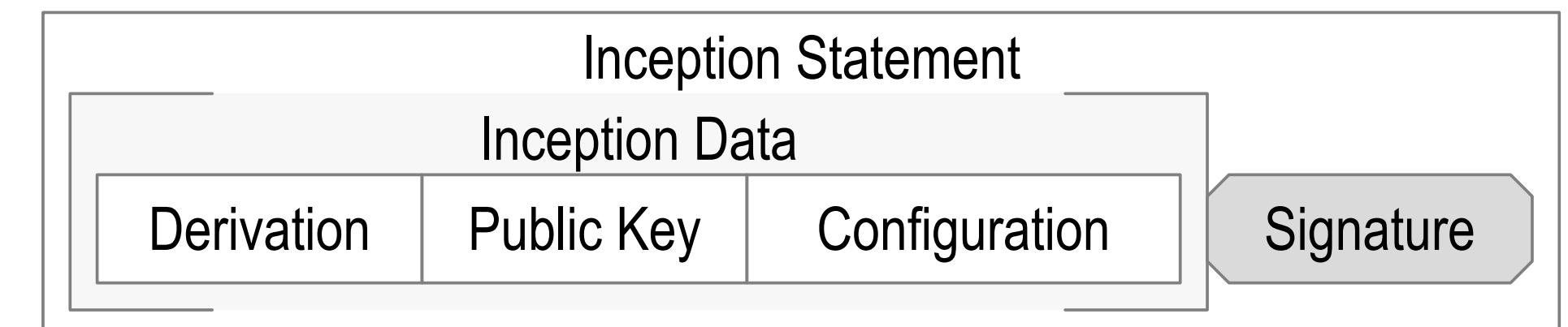
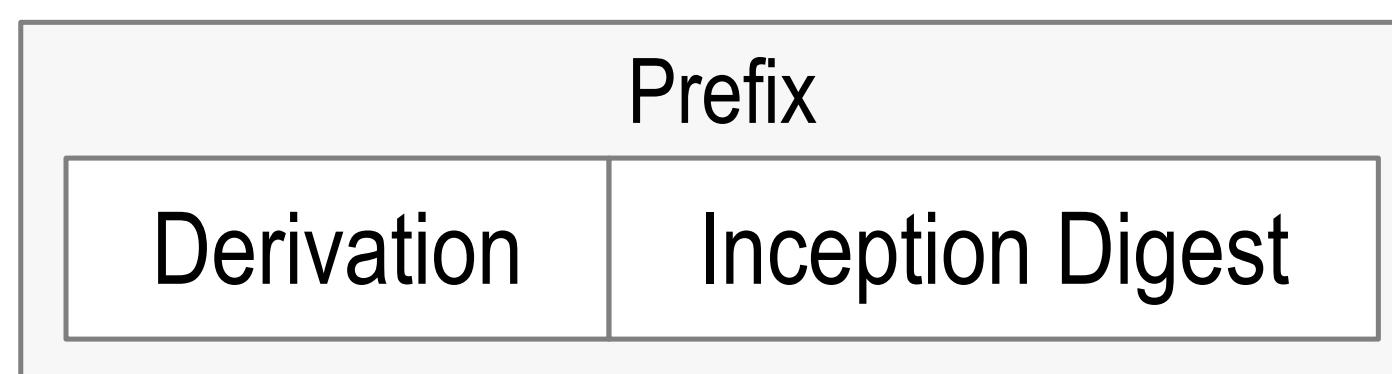
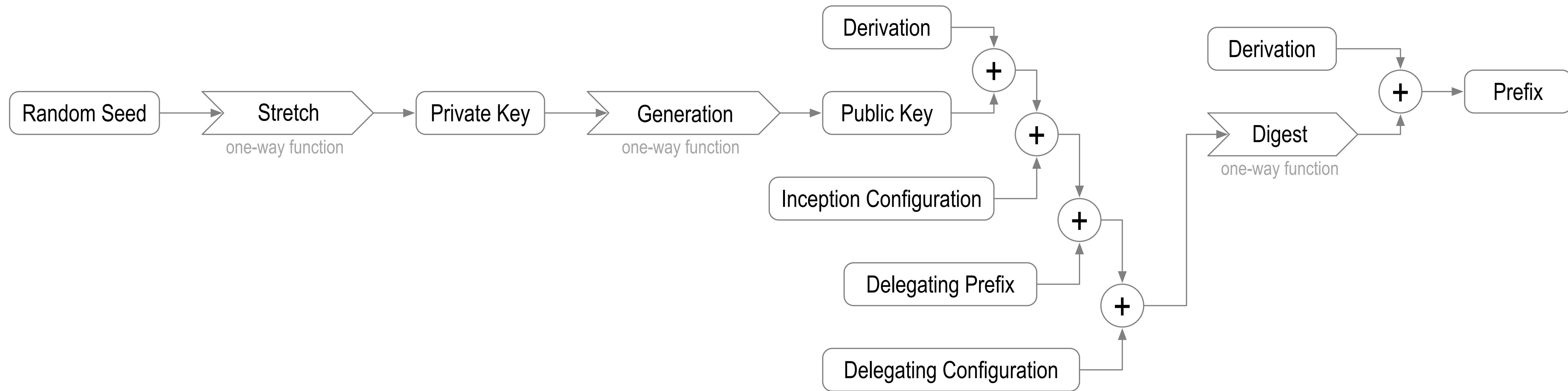
Self-Addressing



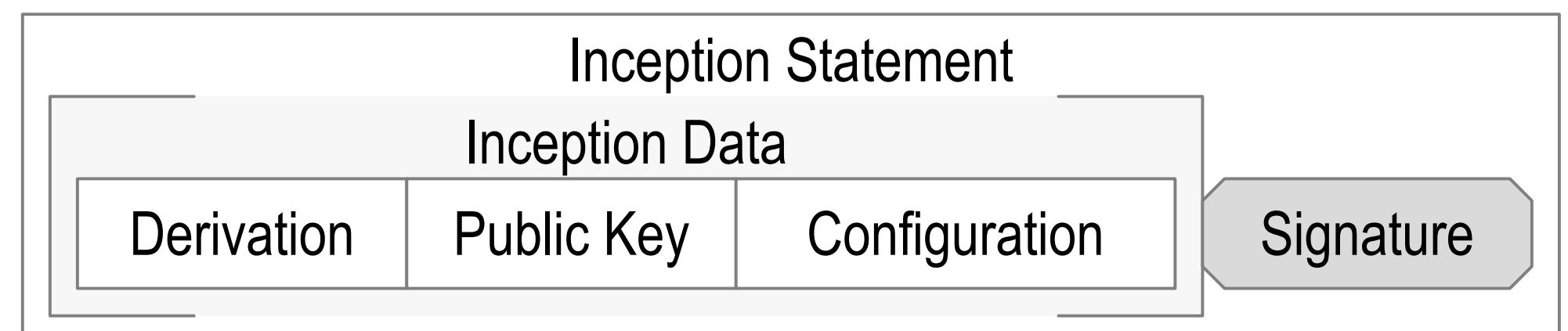
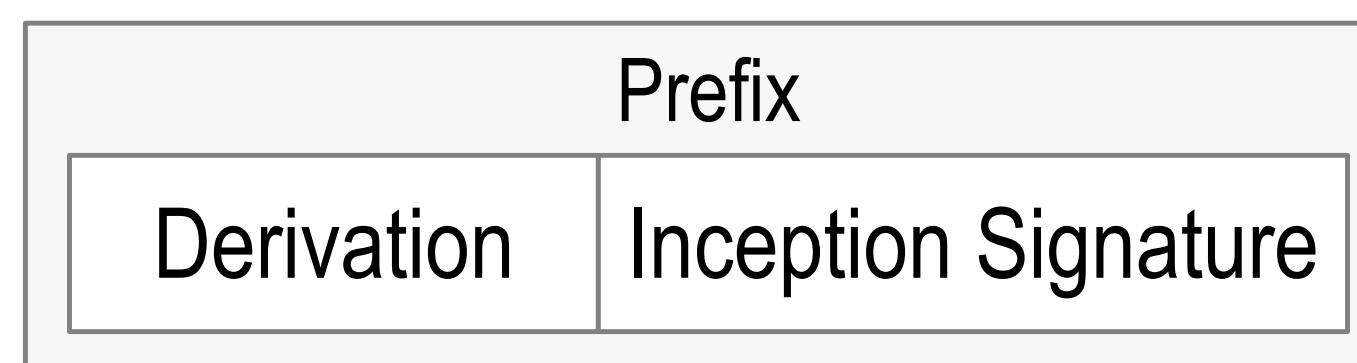
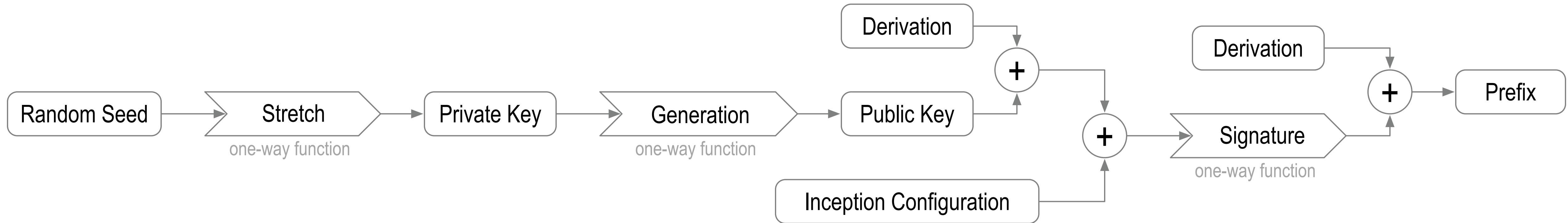
Multi-Sig Self-Addressing



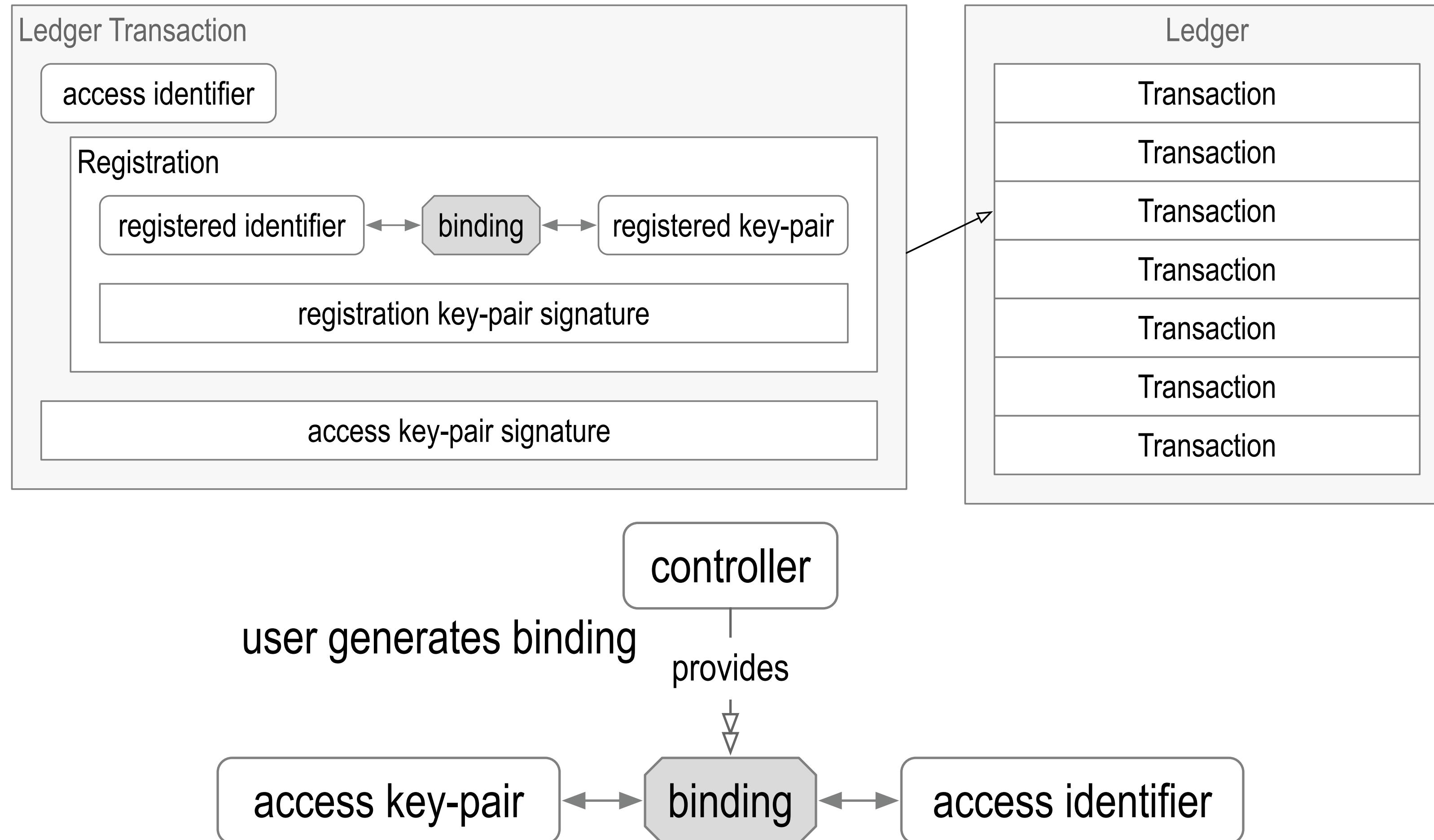
Delegated Self-Addressing



Self-Signing



Ledger Registration



Access identifier may have self-certifying primary root-of-trust
but registered identifier does not, even if its format appears self-certifying.

Autonomic Identifier (AID) and Namespace (AN)

auto nomos = self rule

autonomic = self-governing, self-controlling, etc.

An *autonomic* namespace is
self-certifying and hence *self-administrating*.

ANs are *portable* = truly self-sovereign.

autonomic prefix = self-cert + UUID + URL = universal identifier

Autonomic Identity System

why, how – *who* controls *what, when, and how?*

Root-of-Trust

cryptographic autonomic identifier = *why, how*

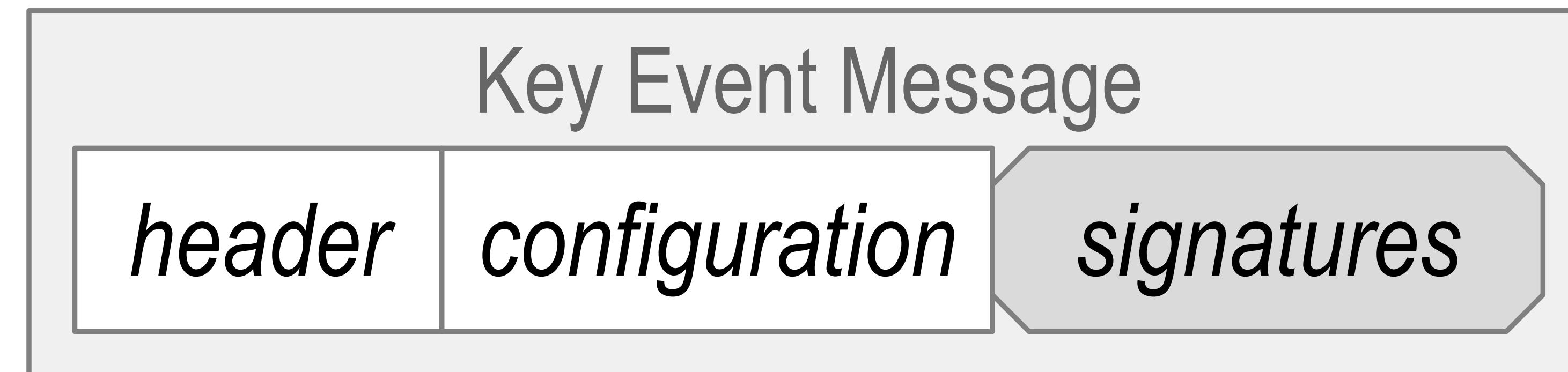
Source-of-Truth

controller of the private key = *who*

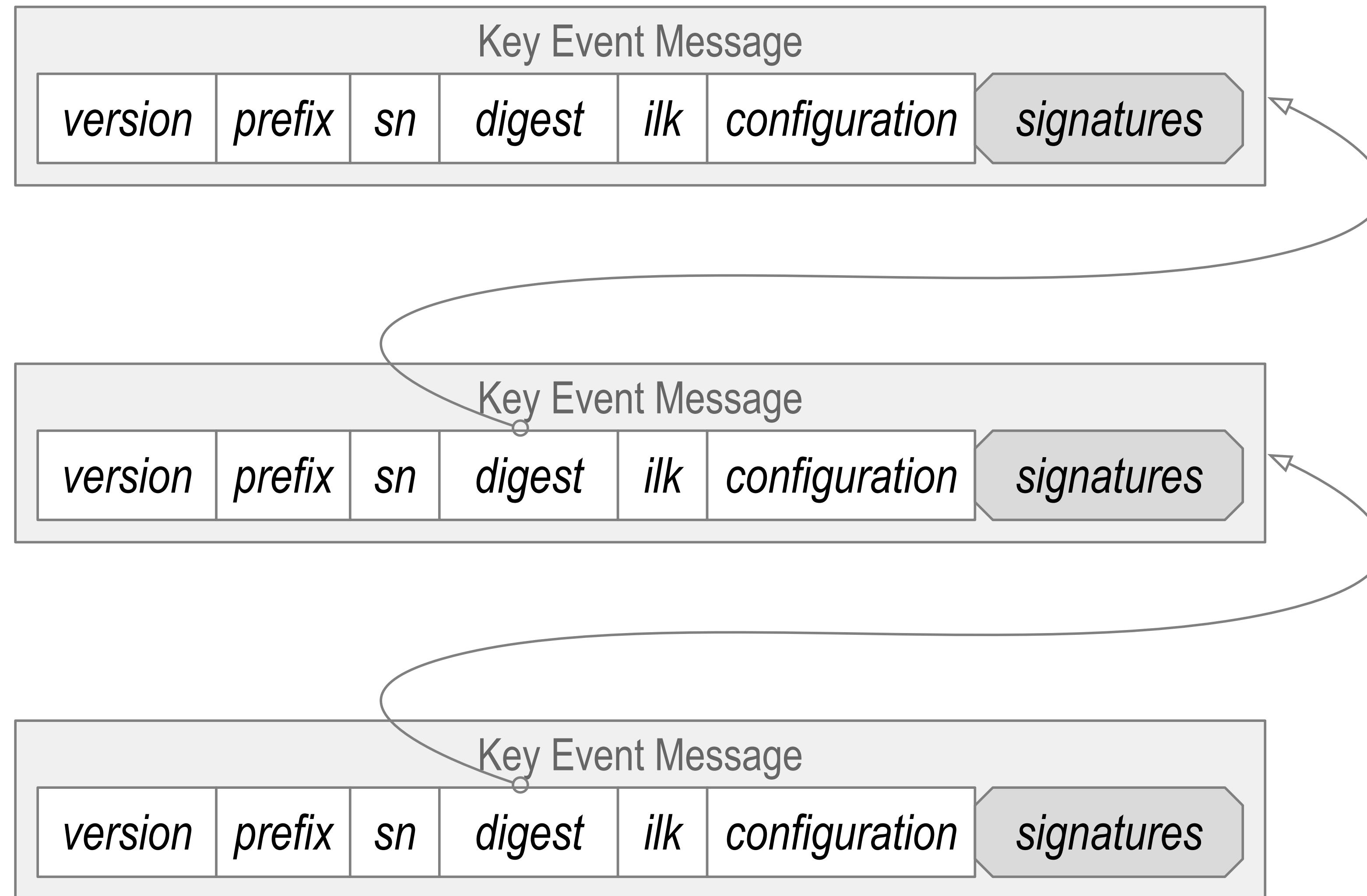
Loci-of-Control

authoritative operation = *what, when, how*

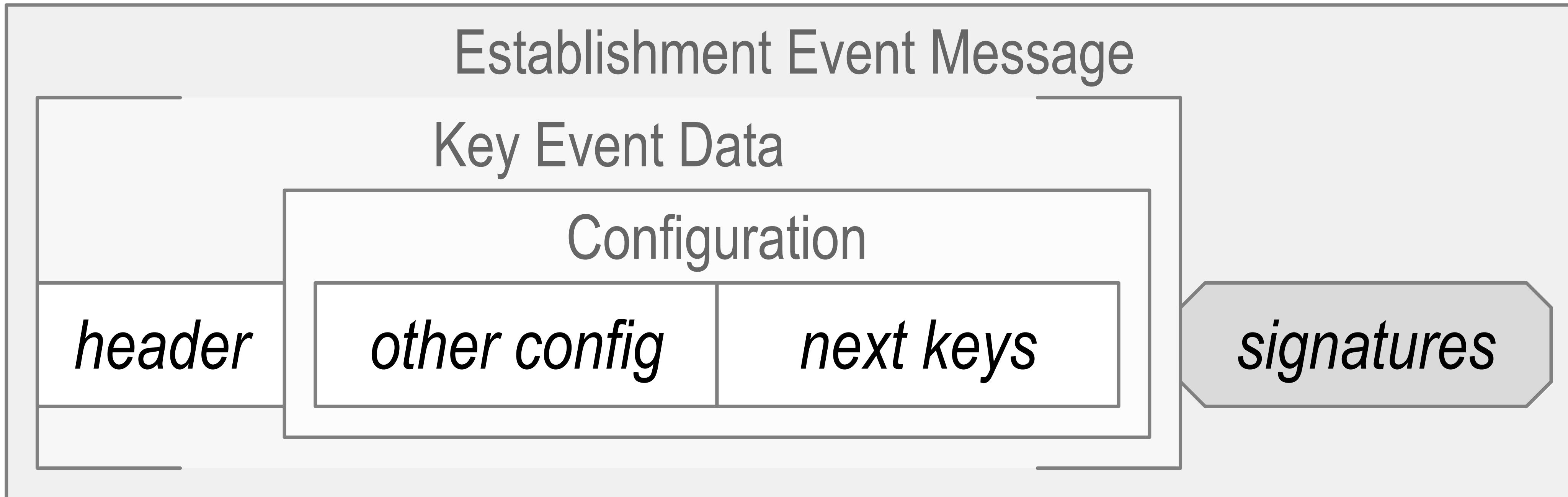
Key Event Message



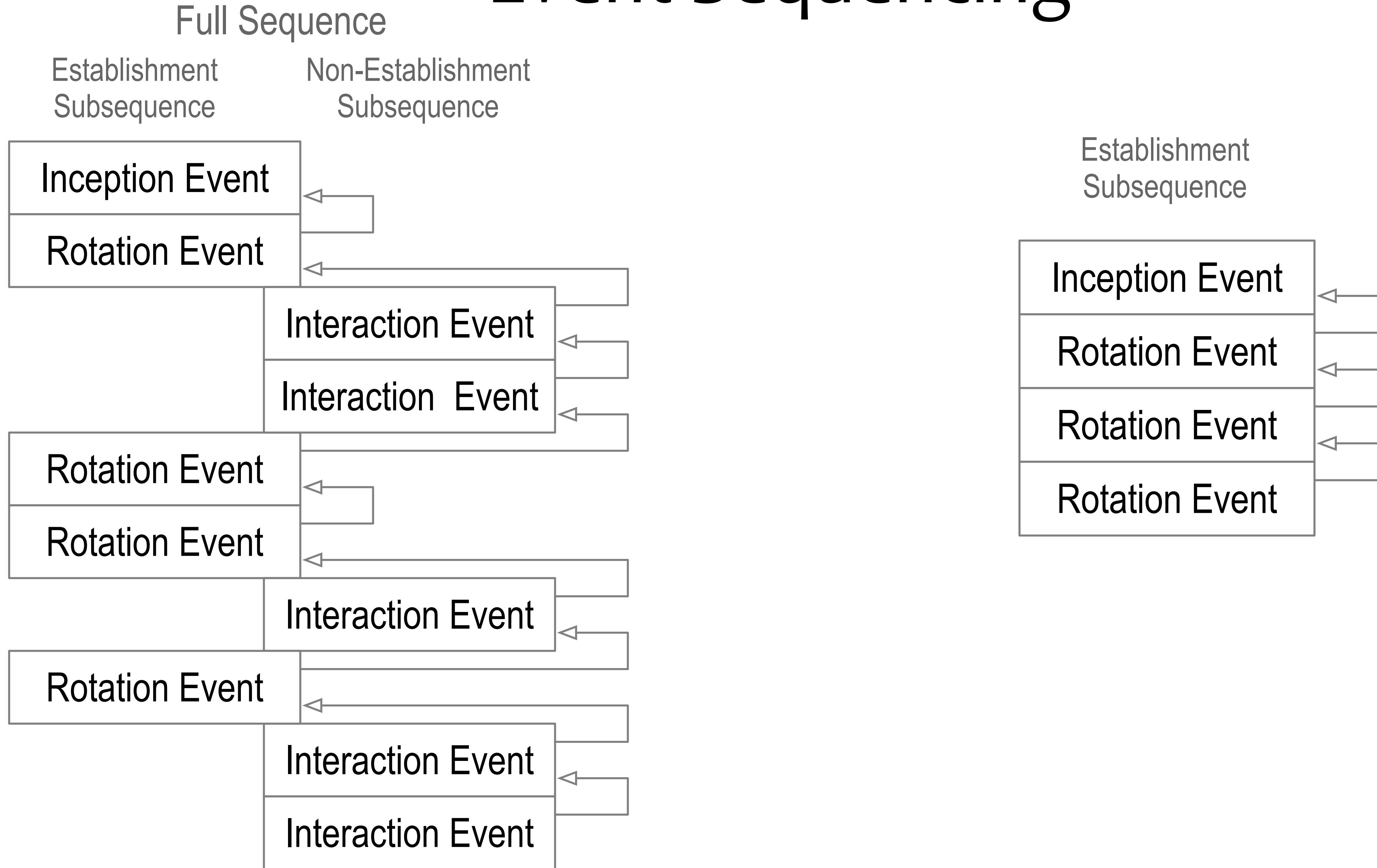
Event Digest Chaining



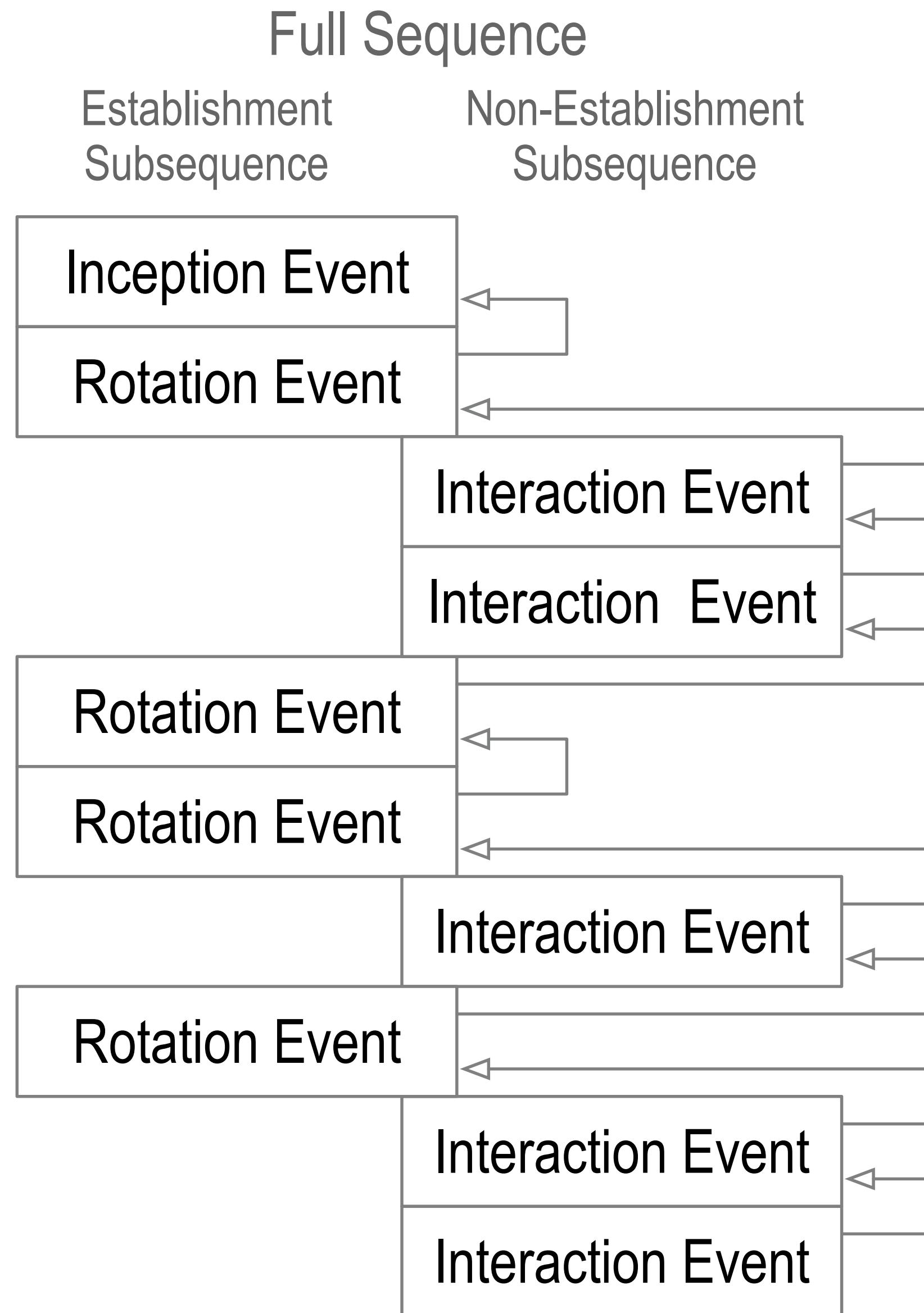
Establishment Event



Event Sequencing



Inconsistency and Duplication



Inconsistency vs. Duplication

inconsistency: lacking agreement, as two or more things in relation to each other

duplicity: acting in two different ways to different people concerning the same matter

Internal vs. External Inconsistency

Internally inconsistent log = not verifiable.

Log verification from self-certifying root-of-trust
protects against internal inconsistency.

Externally inconsistent log with a purported
copy of log but both verifiable = duplicitous.

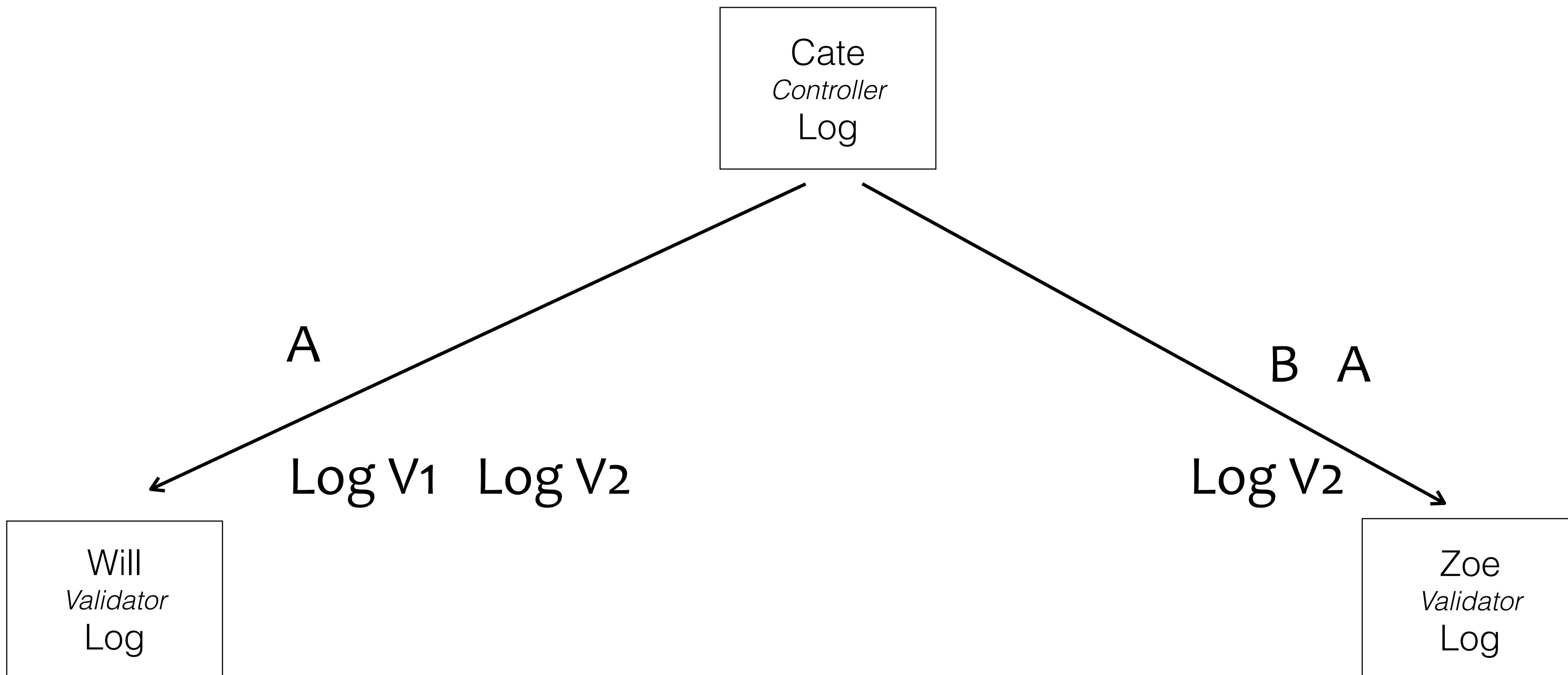
Duplicity detection protects against
external inconsistency.

Cate promises to provide a consistent pair-wise log.

Local Consistency Guarantee

Duplicity Game

How may Cate be *duplicitious* and not get caught?



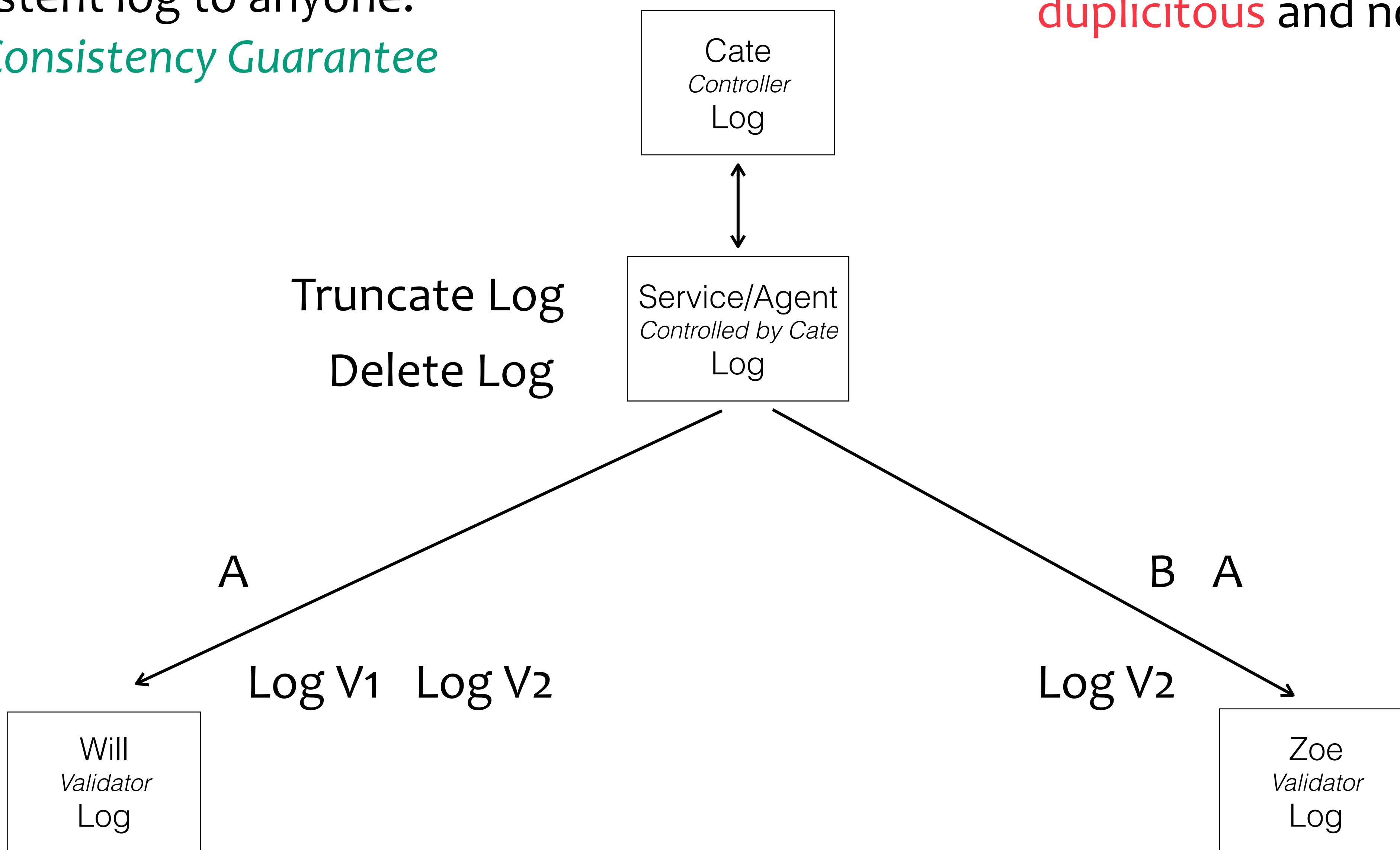
private (one-to-one) interactions

Service promises to provide a consistent log to anyone.

Local Consistency Guarantee

Duplicity Game

How may Cate/Service/Agent be **duplicitous** and not get caught?



highly available, private (one-to-one) interactions

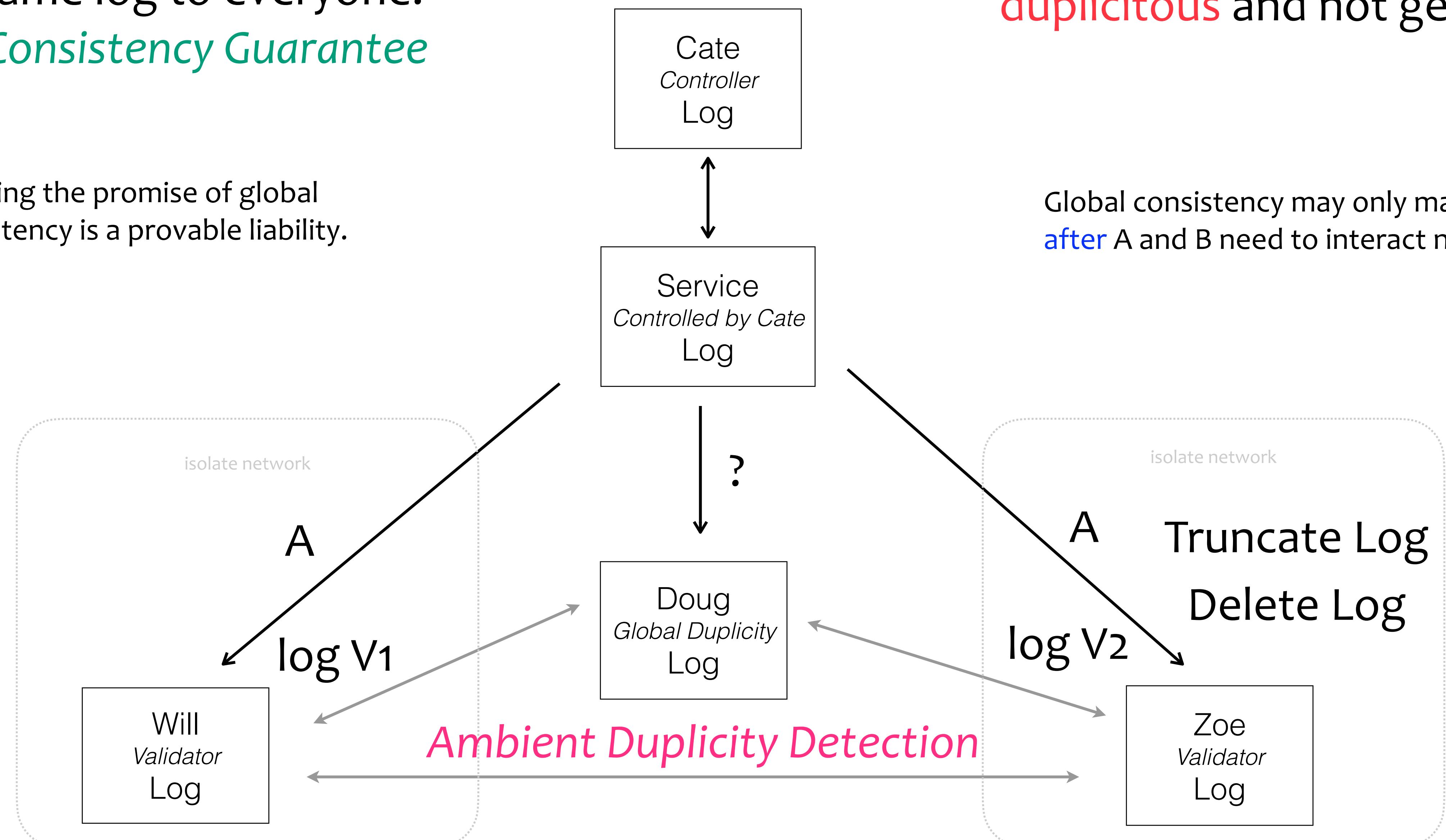
Service promises to provide exact same log to everyone.
Global Consistency Guarantee

Duplicity Game

How may Cate and/or service be **duplicitous** and not get caught?

Breaking the promise of global consistency is a provable liability.

Global consistency may only matter **after** A and B need to interact not before.



global consistent, highly available, and public (one-to-any) interactions

KEY Event Based Provenance of Identifiers

KERI enables cryptographic *proof-of-control-authority (provenance)* for each identifier.

A *proof* is in the form of an identifier's *key event receipt log (KERL)*.

KERLs are *End Verifiable*:

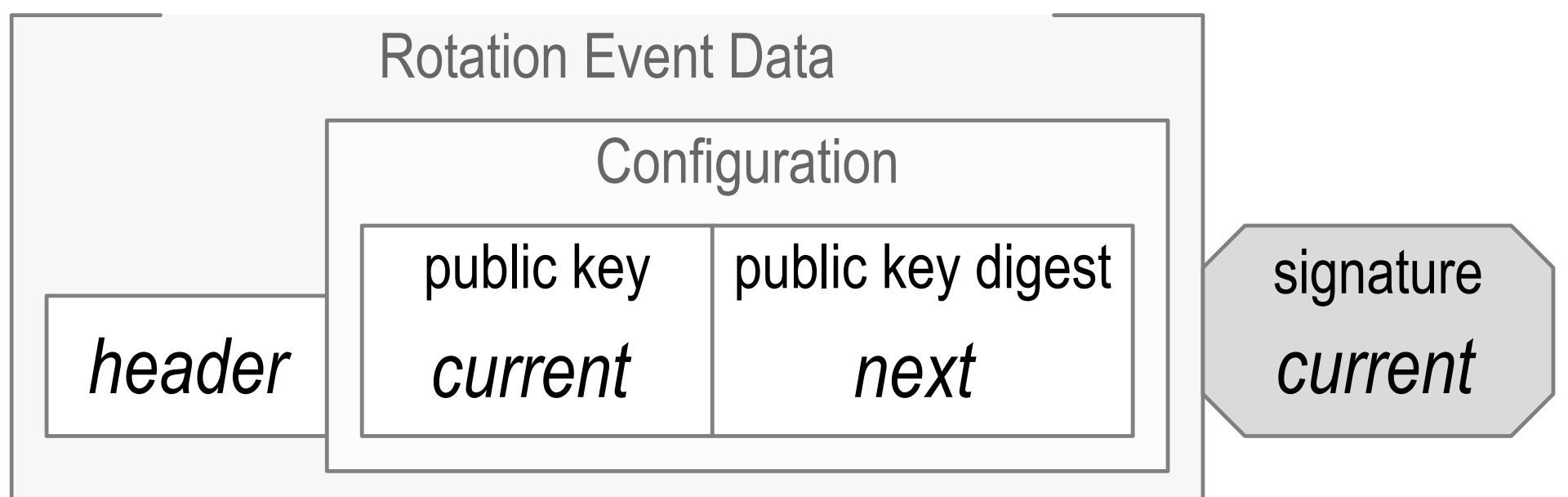
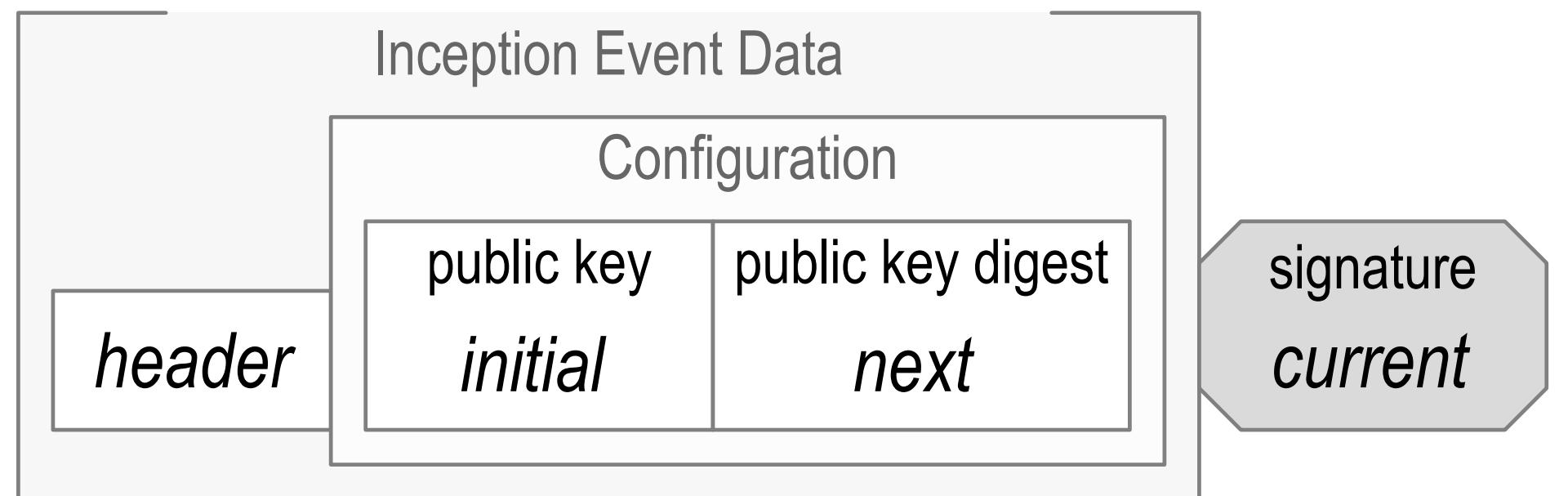
End user alone may verify. Zero trust in intervening infrastructure.

KERLs may be *Ambient Verifiable*:

Anyone may verify *anylog, anywhere, at anytime*.

KERI = self-cert root-of-trust + certificate transparency + KA²CE + recoverable + post-quantum.

Pre-Rotation



Inception			
SN	initial	next digest	current
0	C^0	\underline{C}^1	C^0

Rotation			
SN	current	next digest	current
1	C^1	\underline{C}^2	C^1

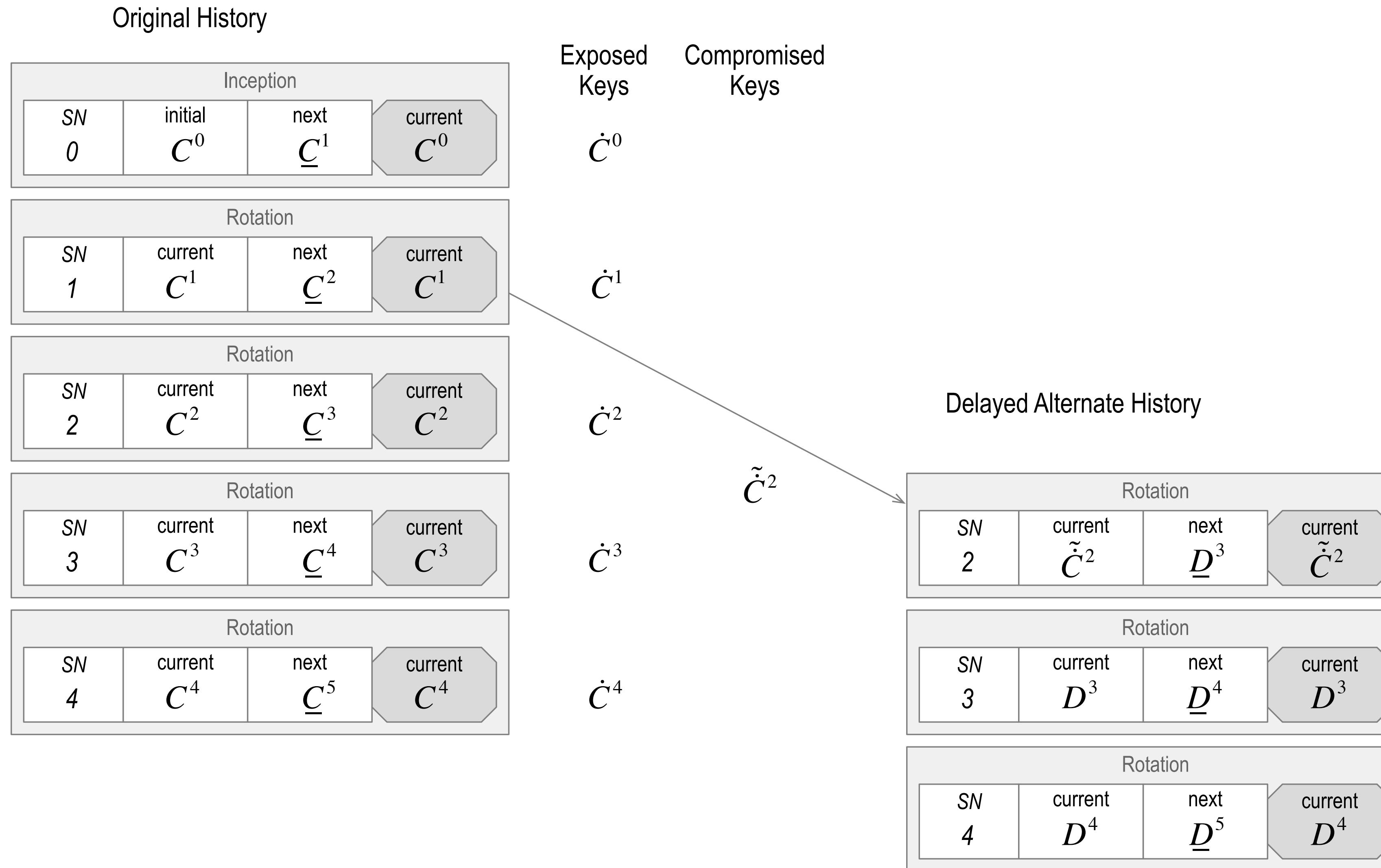
Rotation			
SN	current	next digest	current
2	C^2	\underline{C}^3	C^2

Rotation			
SN	current	next digest	current
3	C^3	\underline{C}^4	C^3

Rotation			
SN	current	next digest	current
4	C^4	\underline{C}^5	C^4

Digest of *next* key(s) makes pre-rotation post-quantum secure

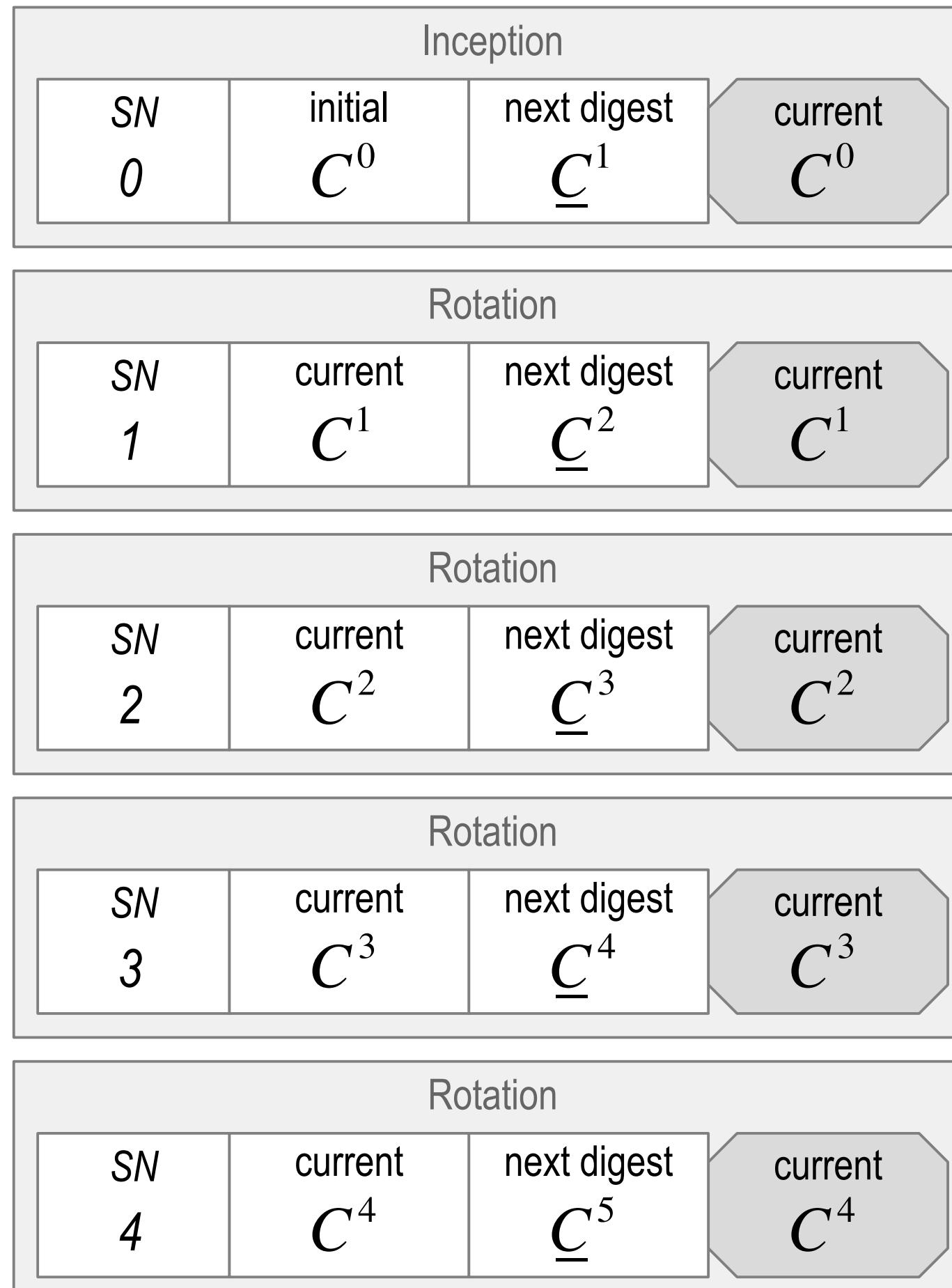
Dead Exploit



Any copy of original history protects against successful *dead exploit*

Live Exploit

Original History



Exposed Keys
Compromised Keys

\dot{C}^0

\dot{C}^1

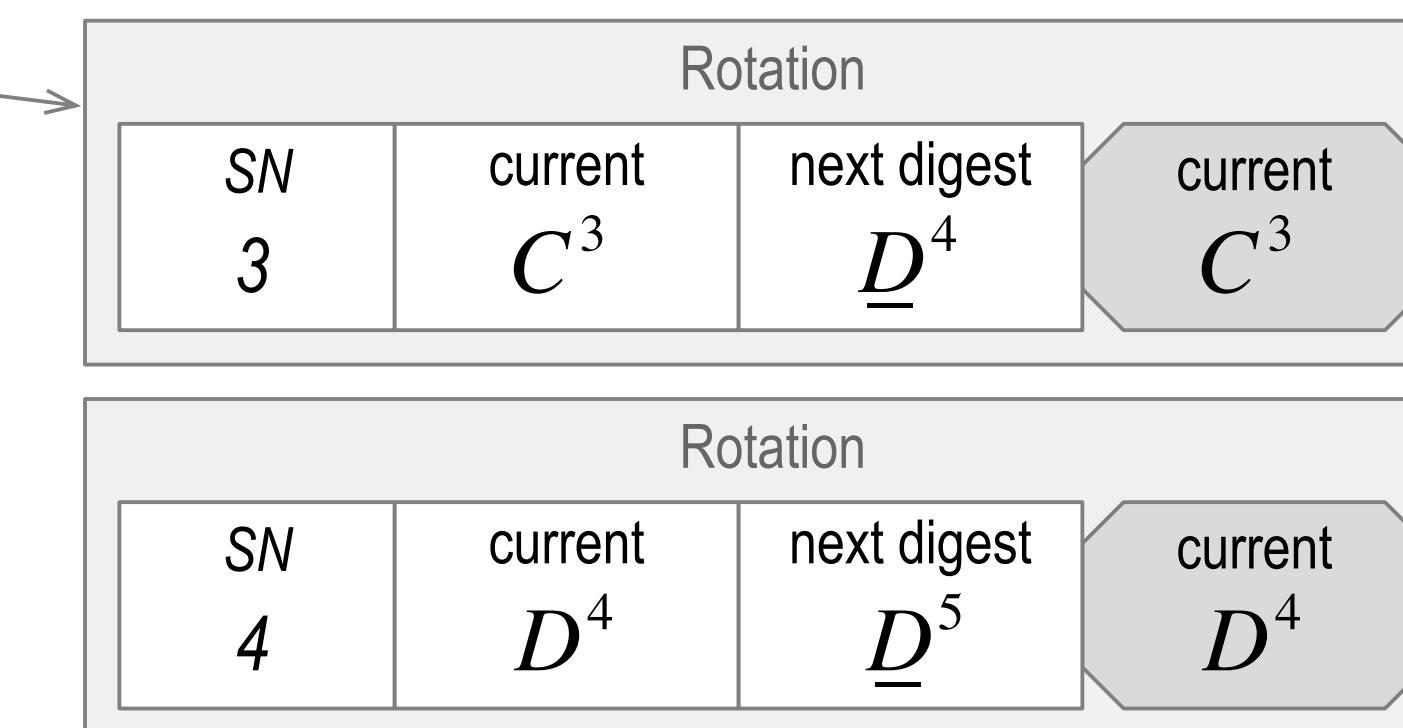
\dot{C}^2

\tilde{C}^3

\dot{C}^3

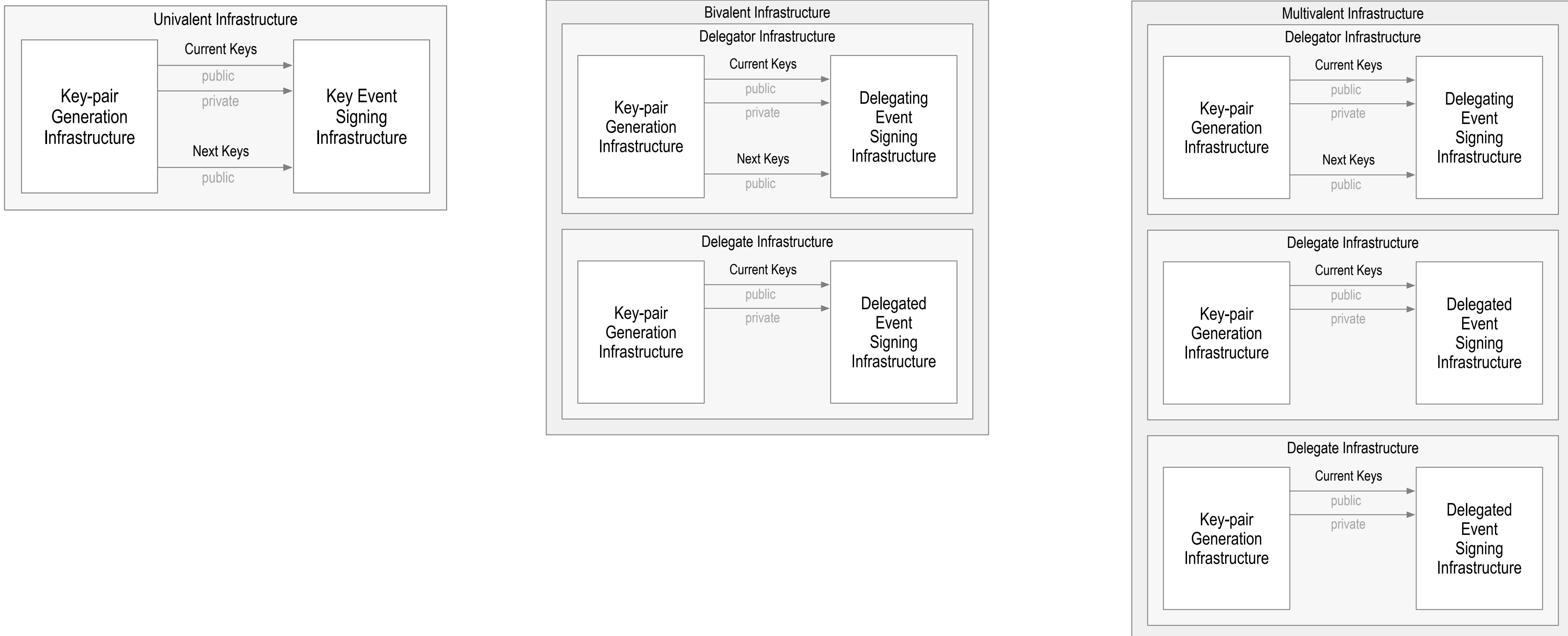
\dot{C}^4

Preemptive Alternate History

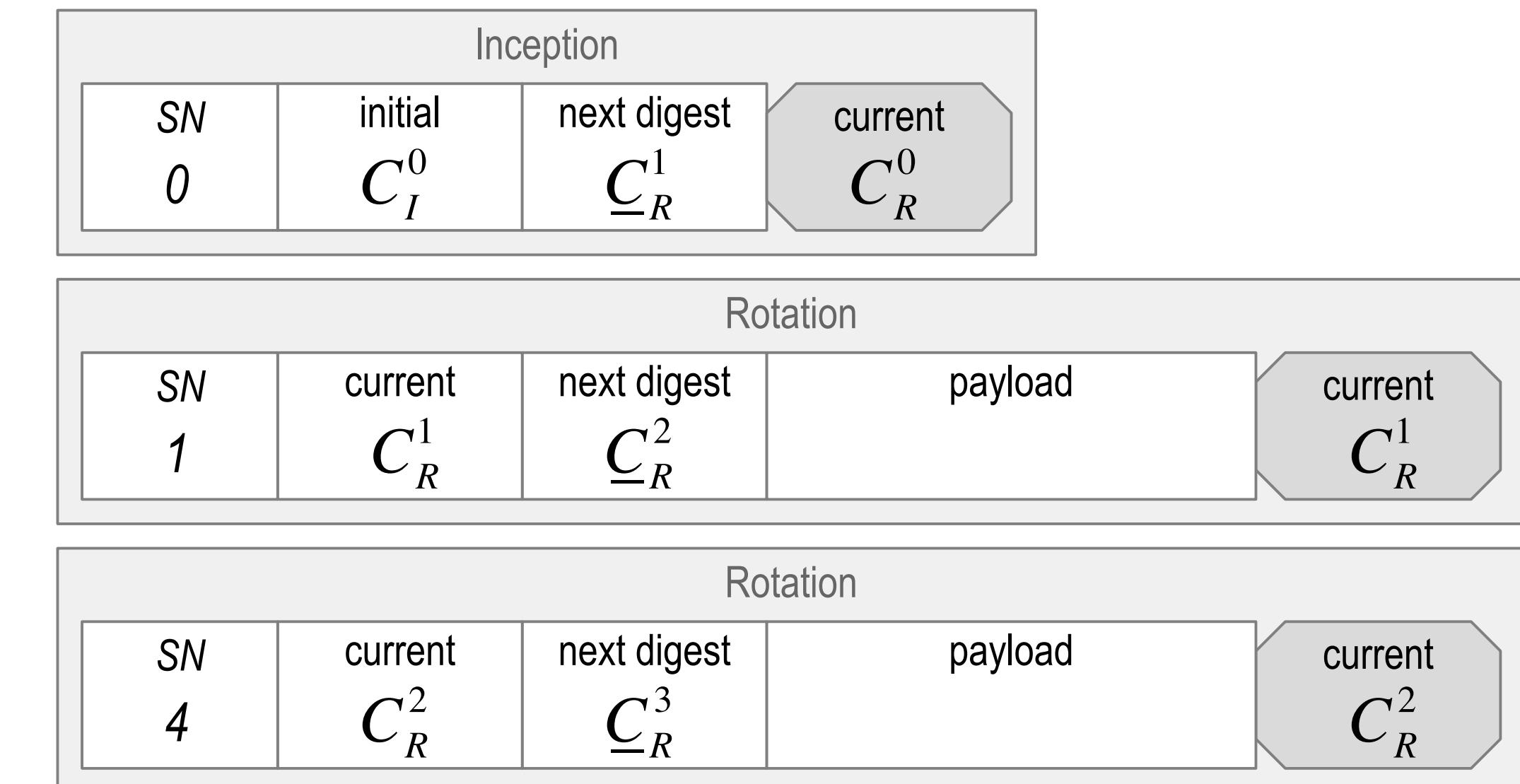
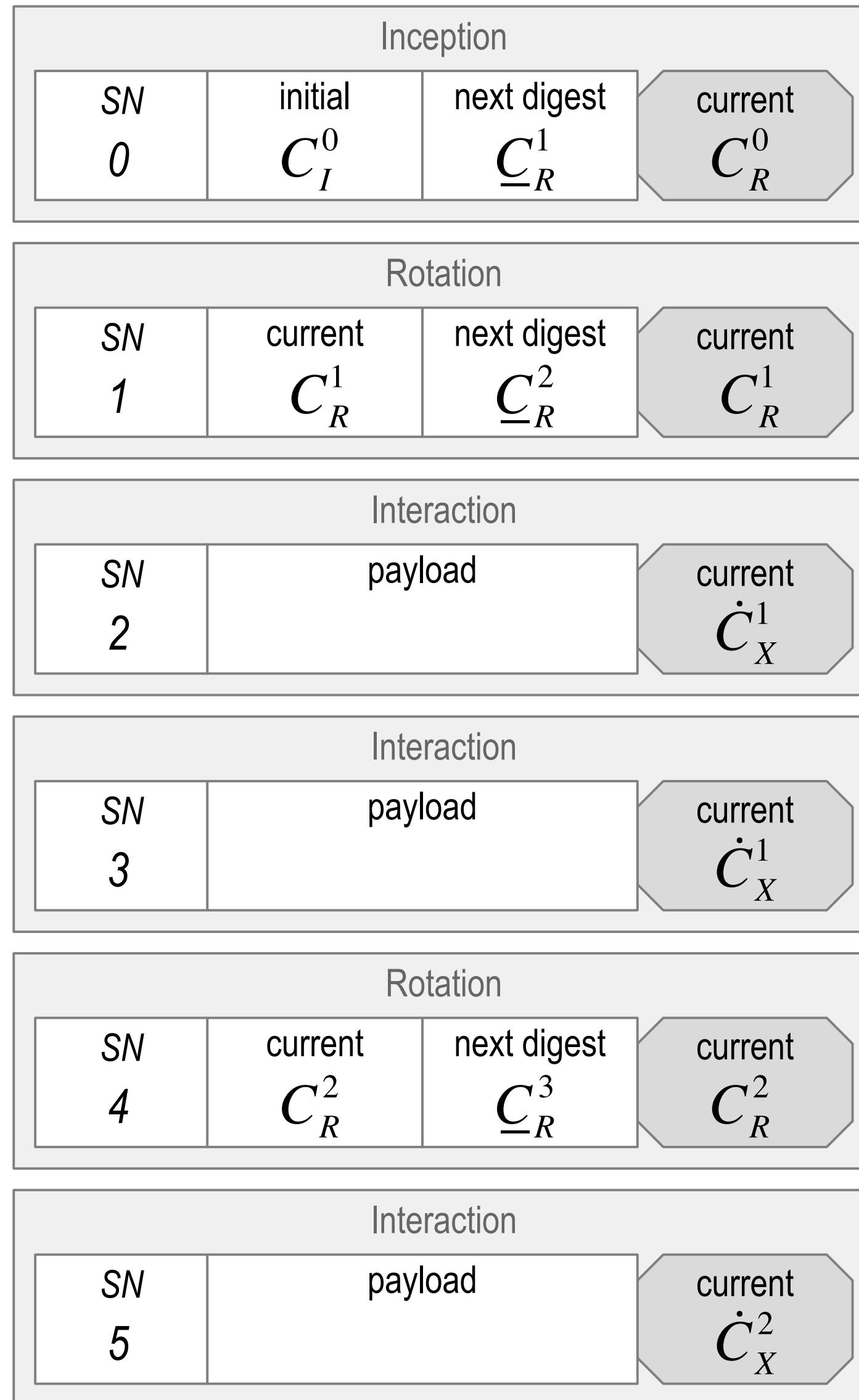


Difficulty of inverting next key(s) protects against successful *live* exploit

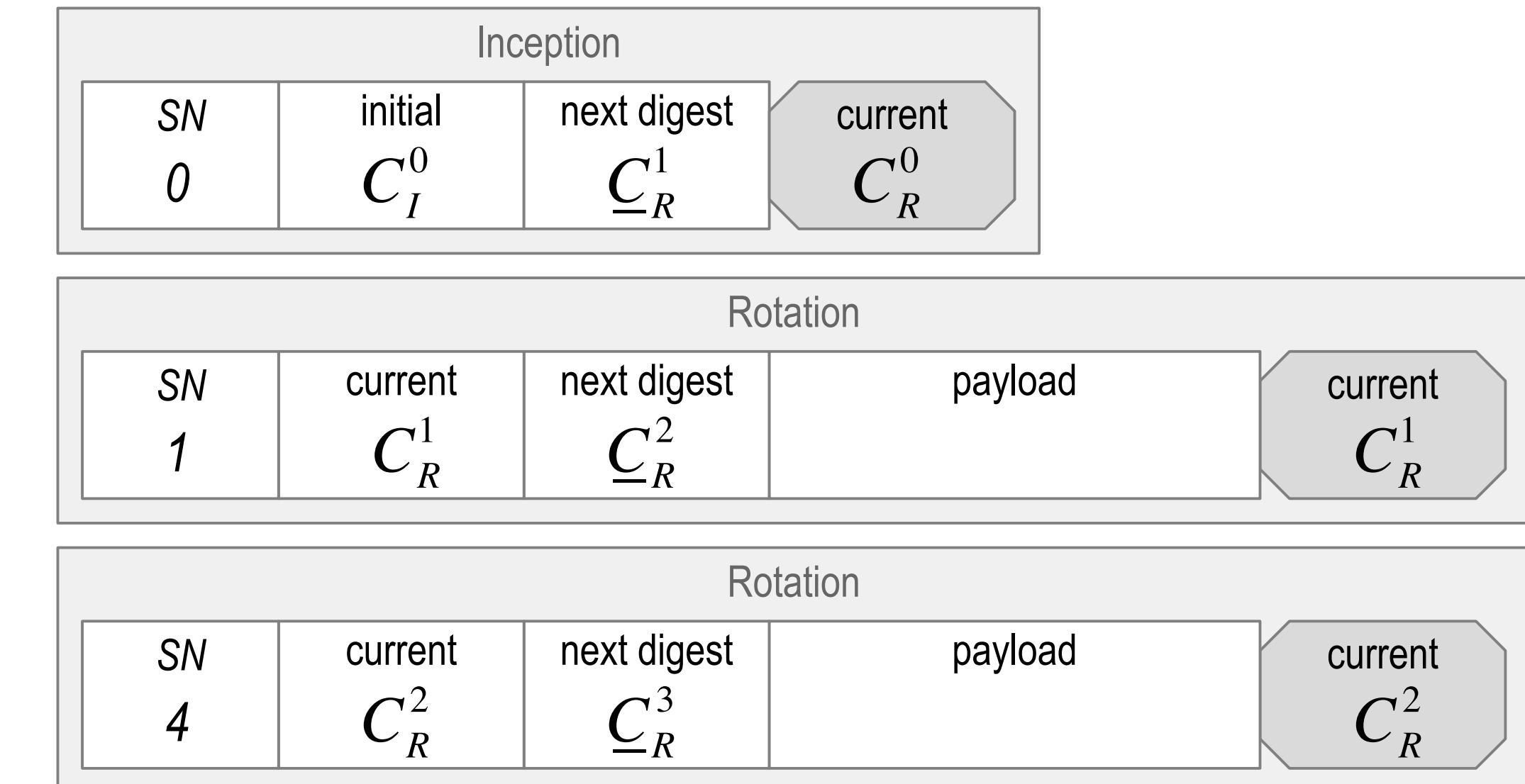
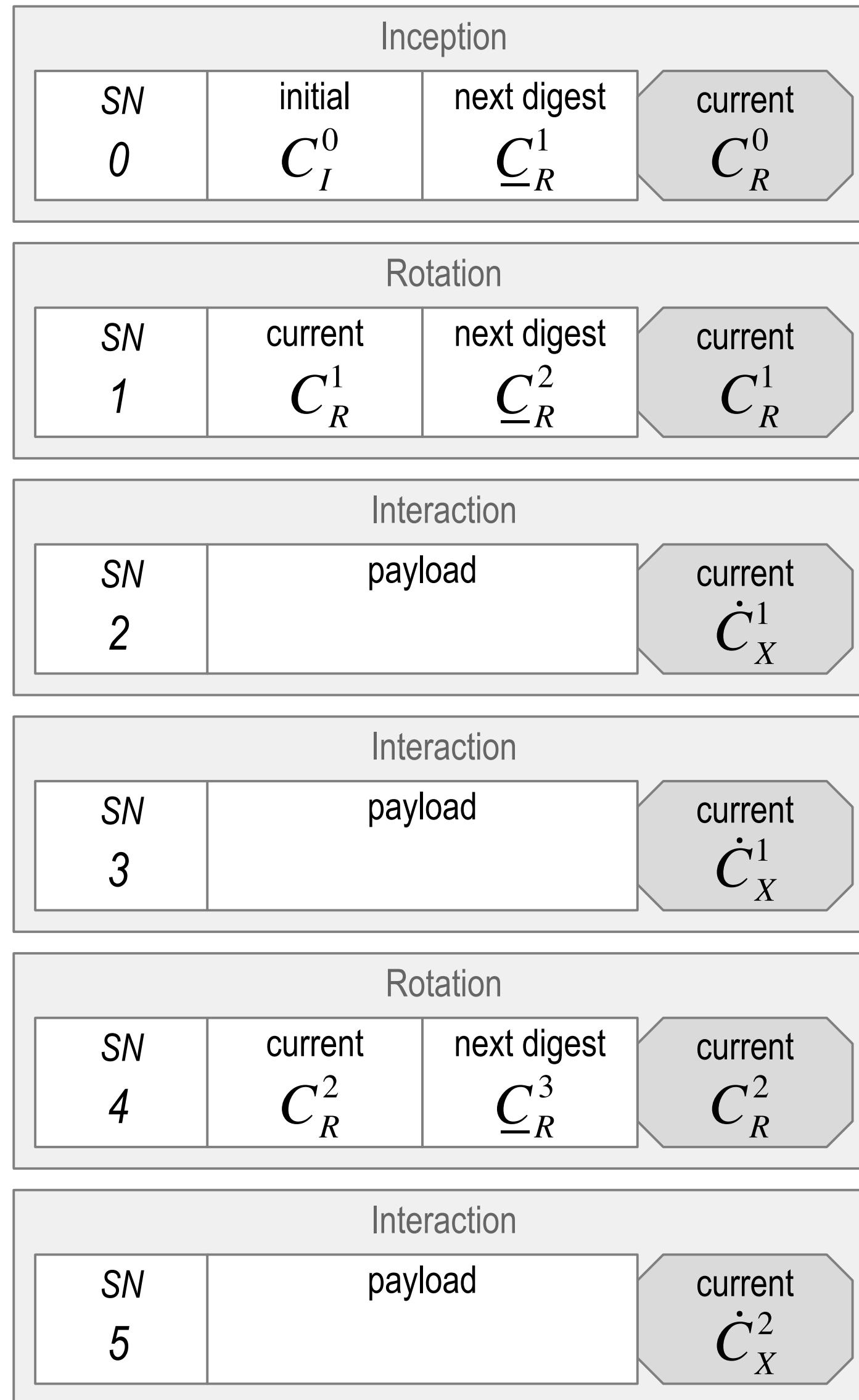
Key Infrastructure Valence



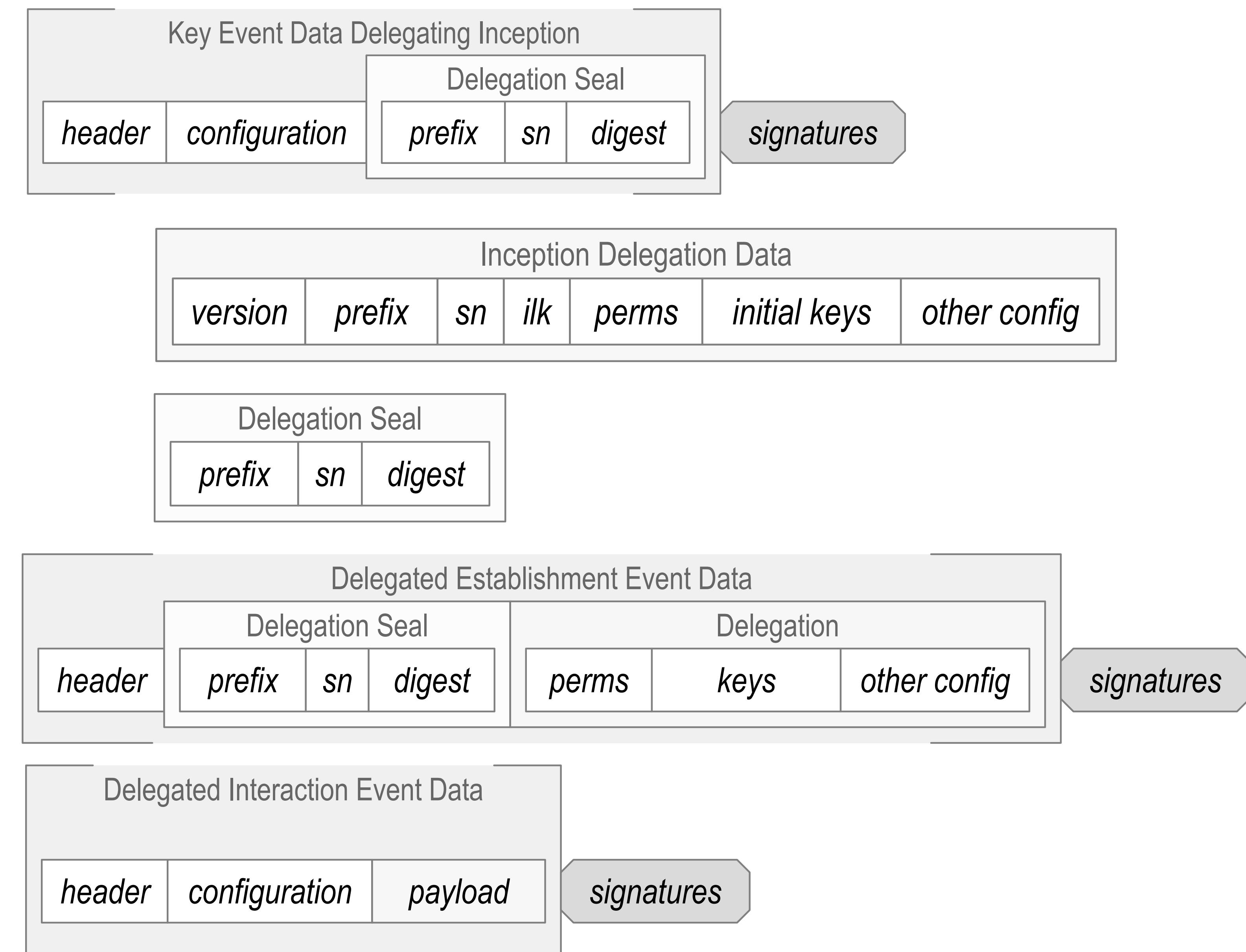
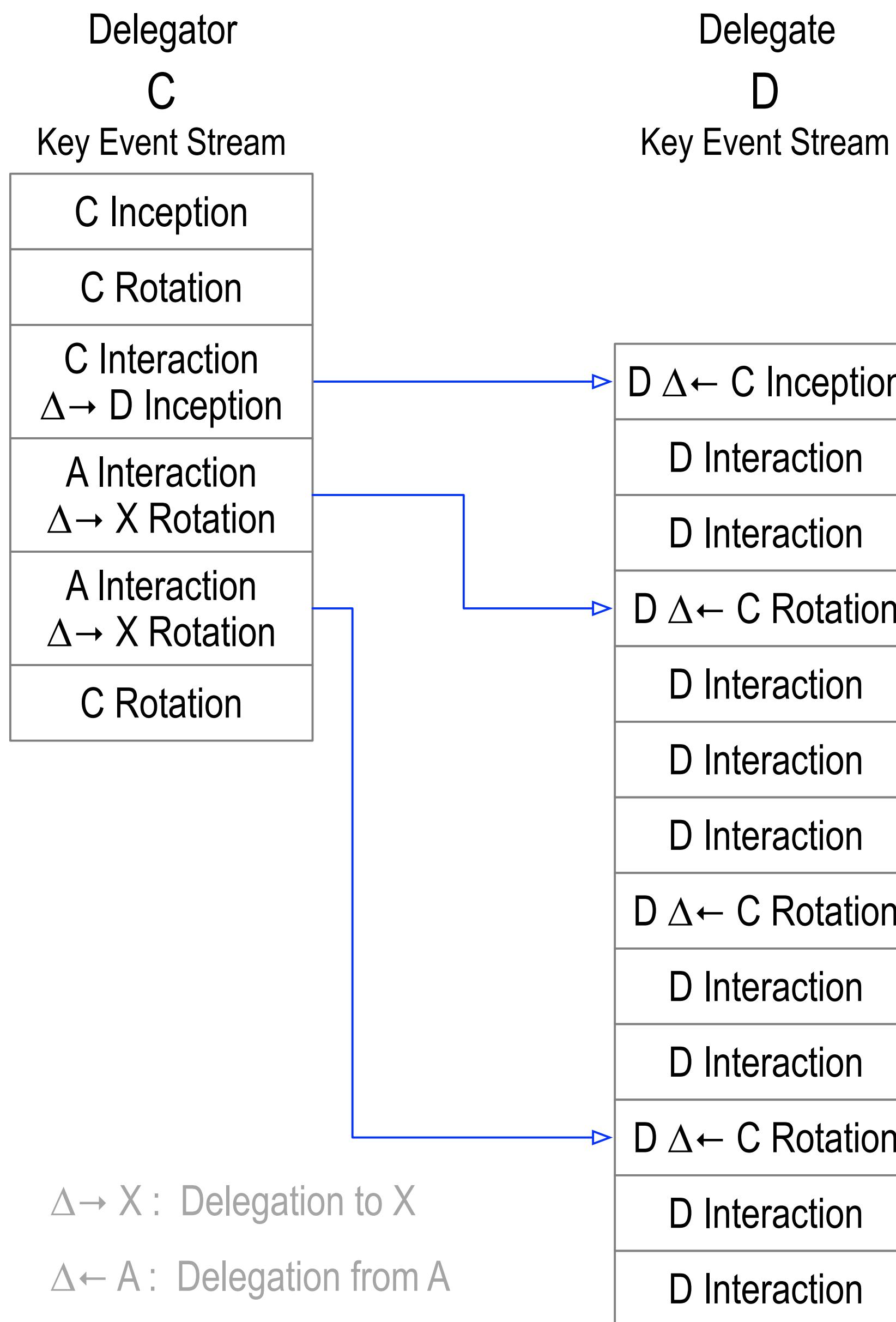
Repurposed Keys



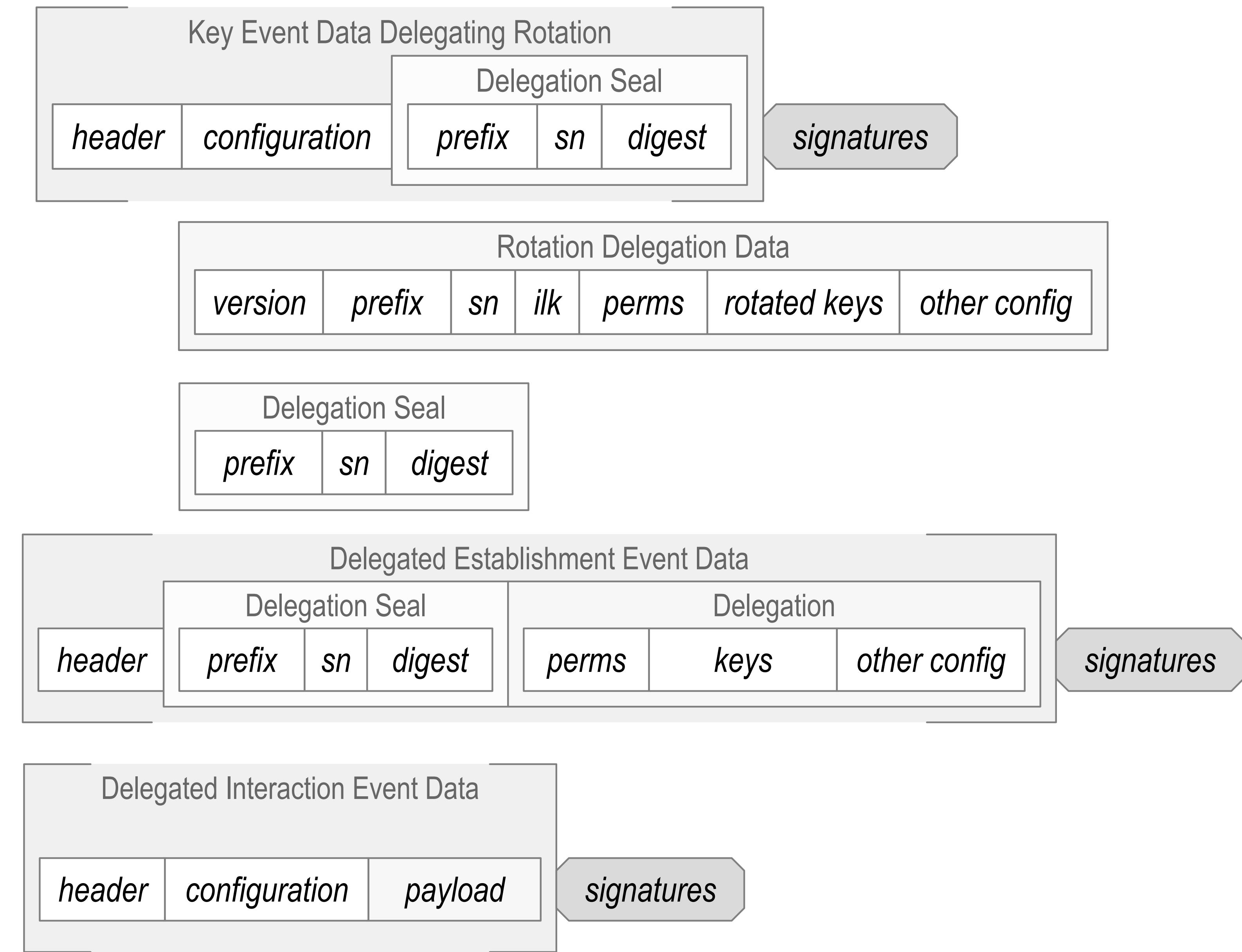
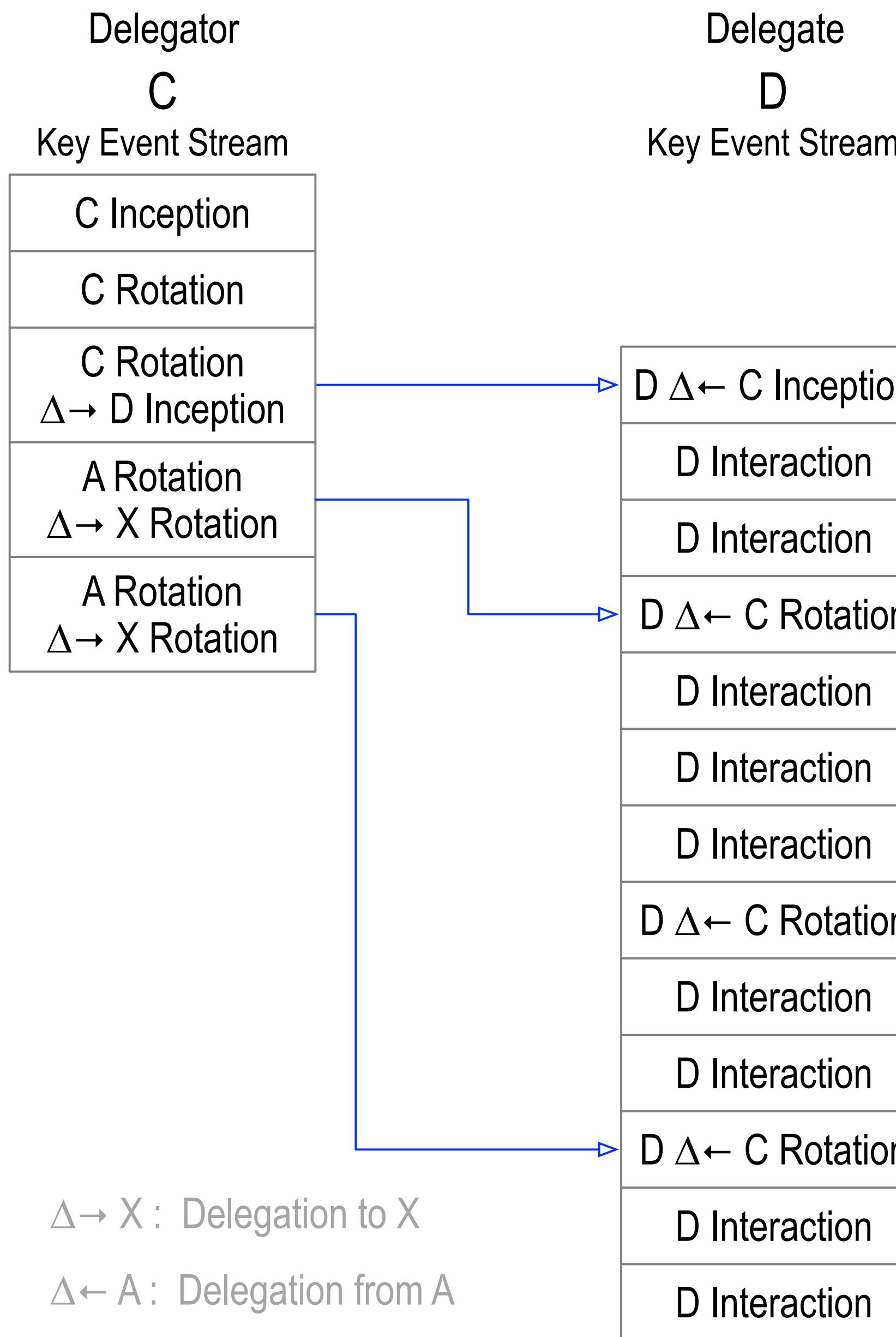
Repurposed Keys



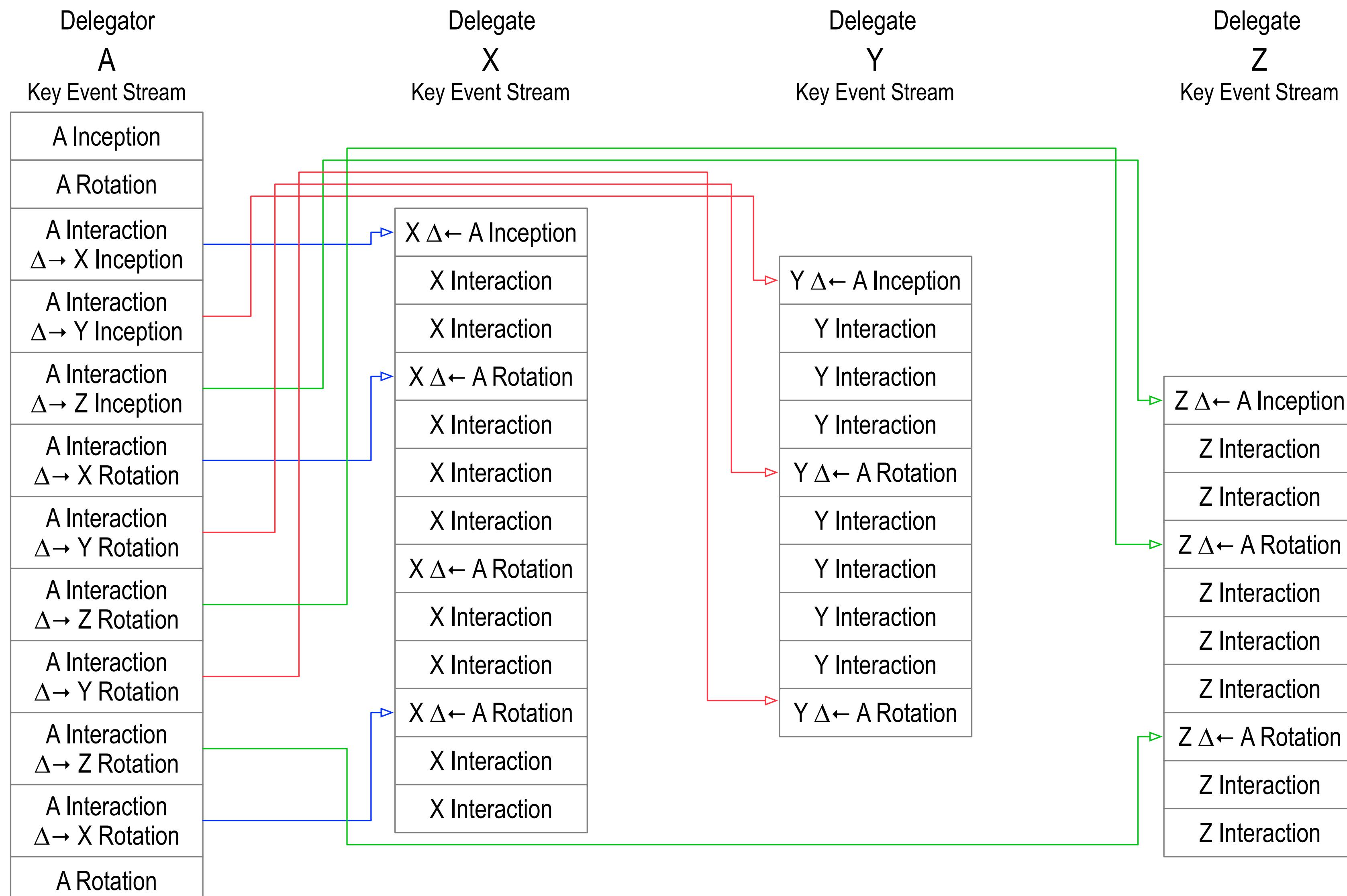
Interaction Delegation



Rotation Delegation



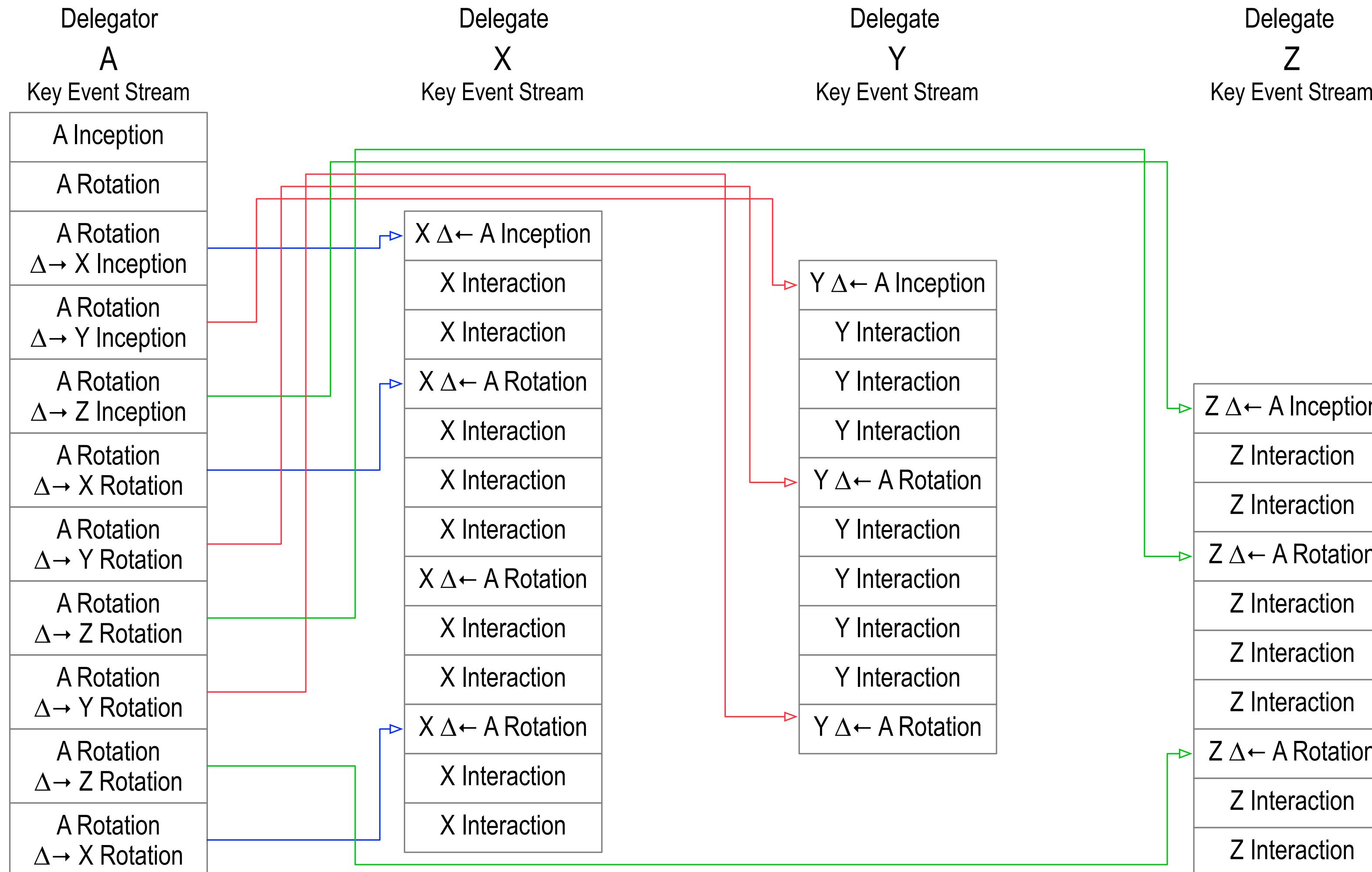
Scaling Delegation via Interaction



$\Delta \rightarrow X$: Delegation to X

$\Delta \leftarrow A$: Delegation from A

Scaling Delegation via Rotation



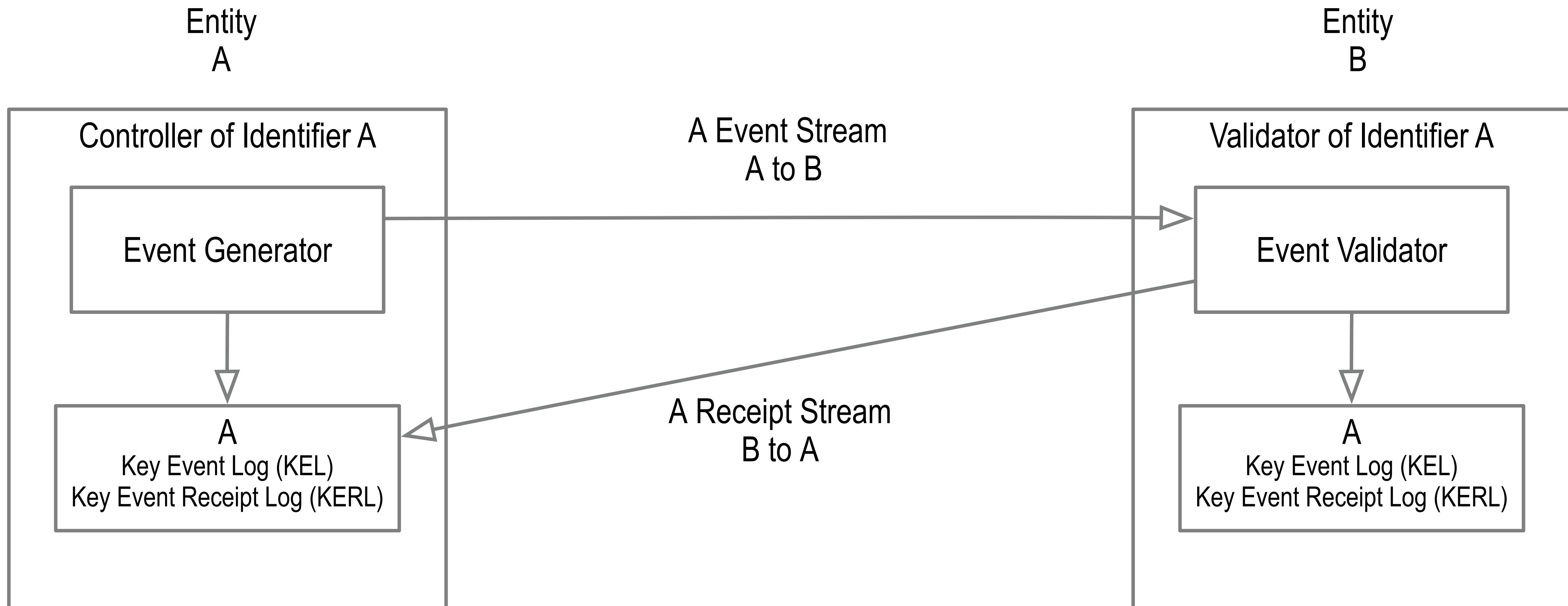
$\Delta \rightarrow X$: Delegation to X
 $\Delta \leftarrow A$: Delegation from A

Protocol Operational Modes

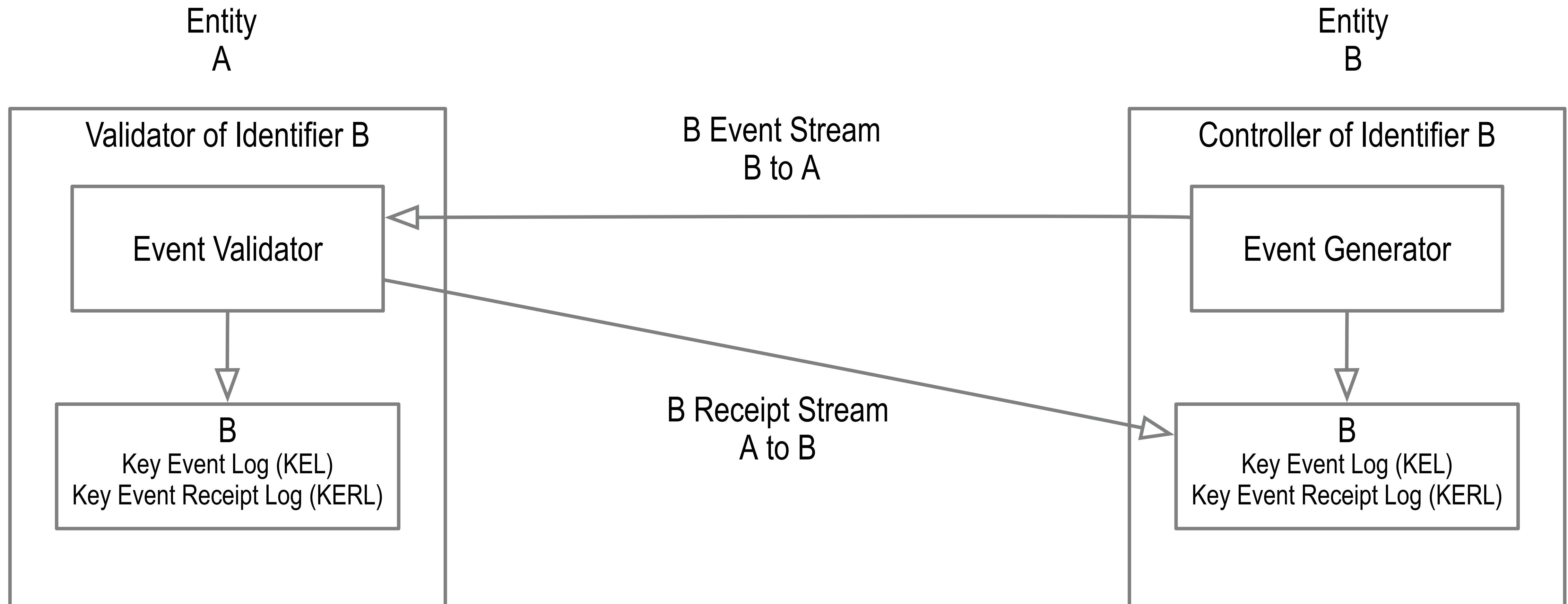
Direct Event Replay Mode (one-to-one)

Indirect Event Replay Mode (one-to-any)

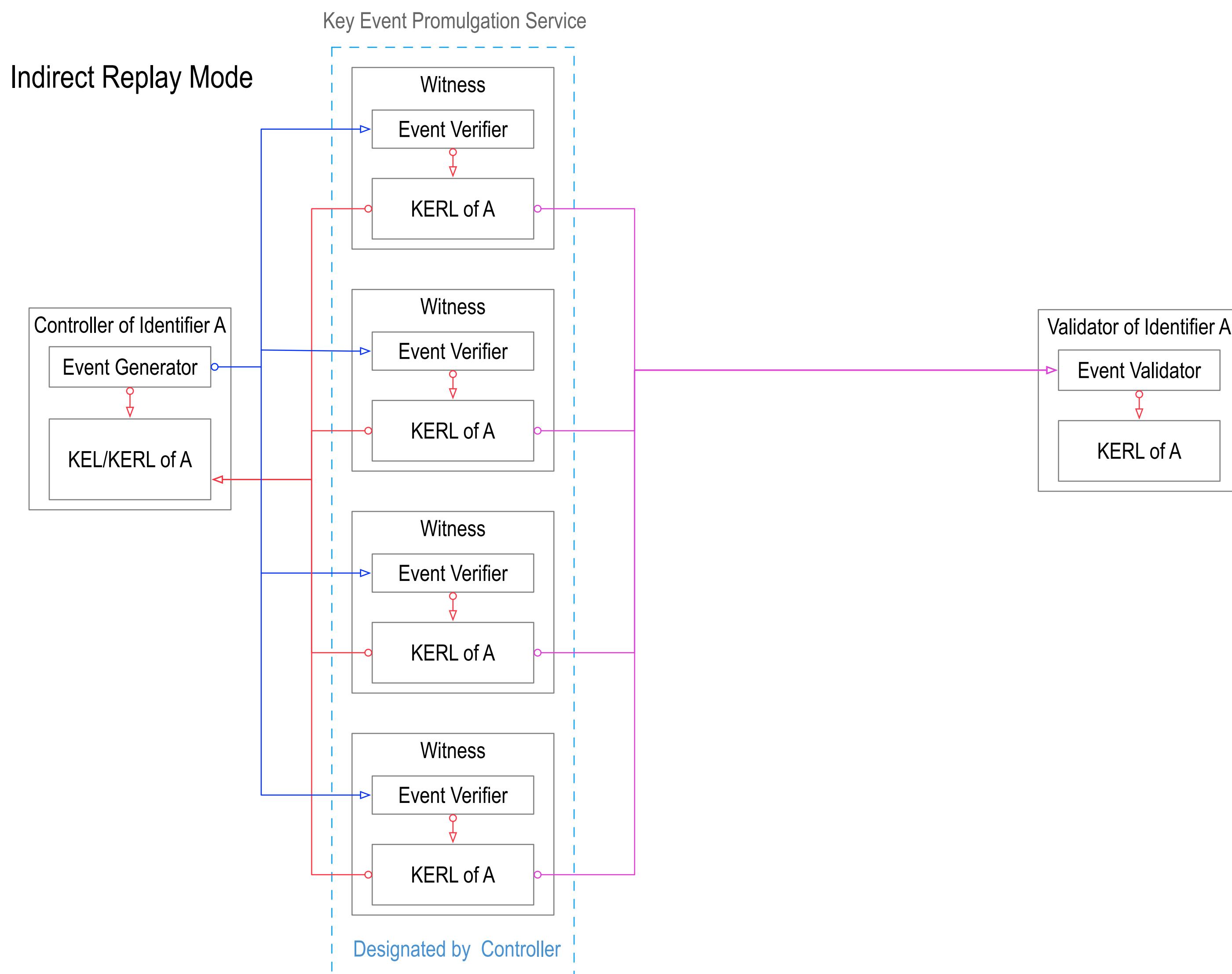
Direct Mode: A to B



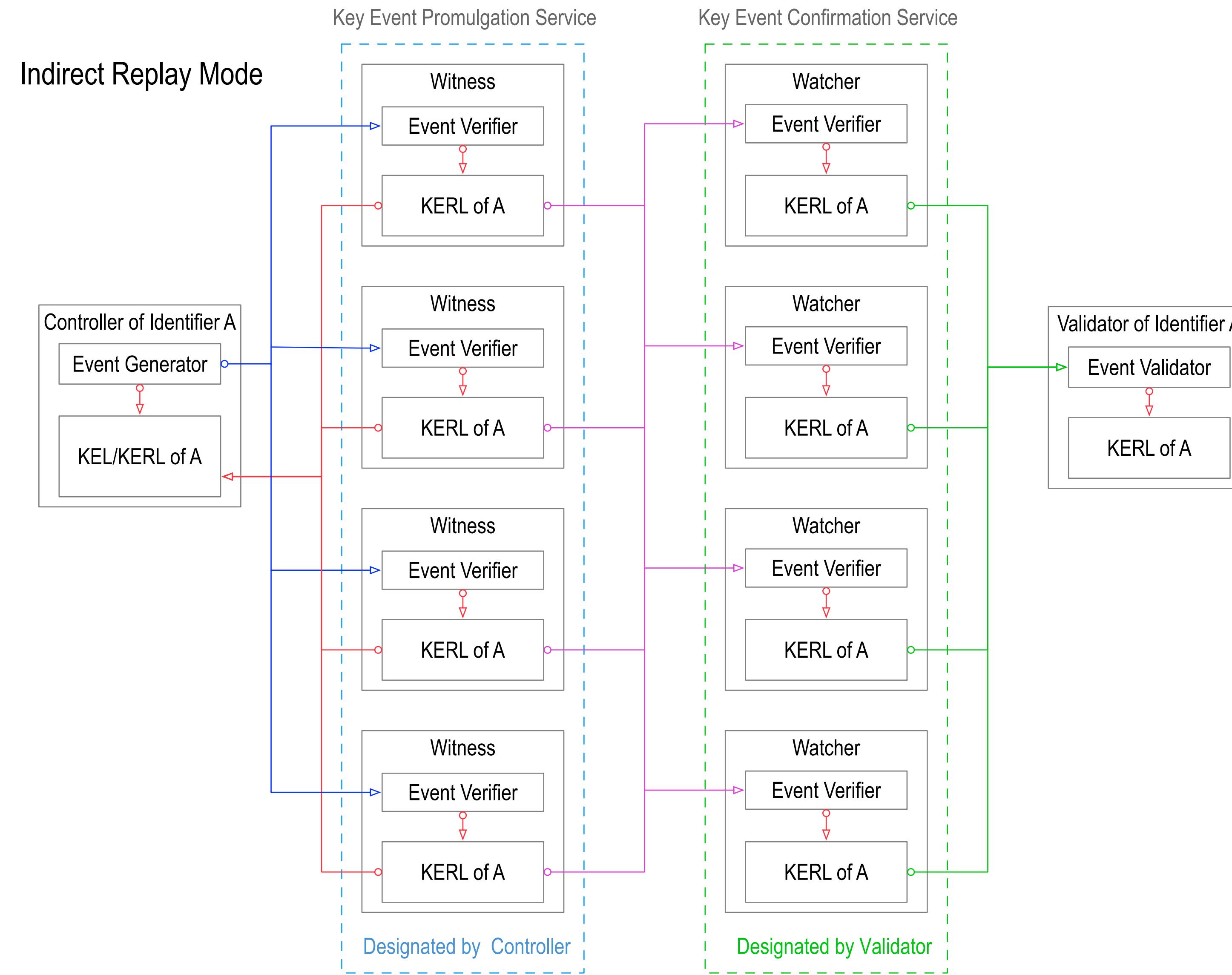
Direct Mode: B to A



Indirect Mode Promulgation Service

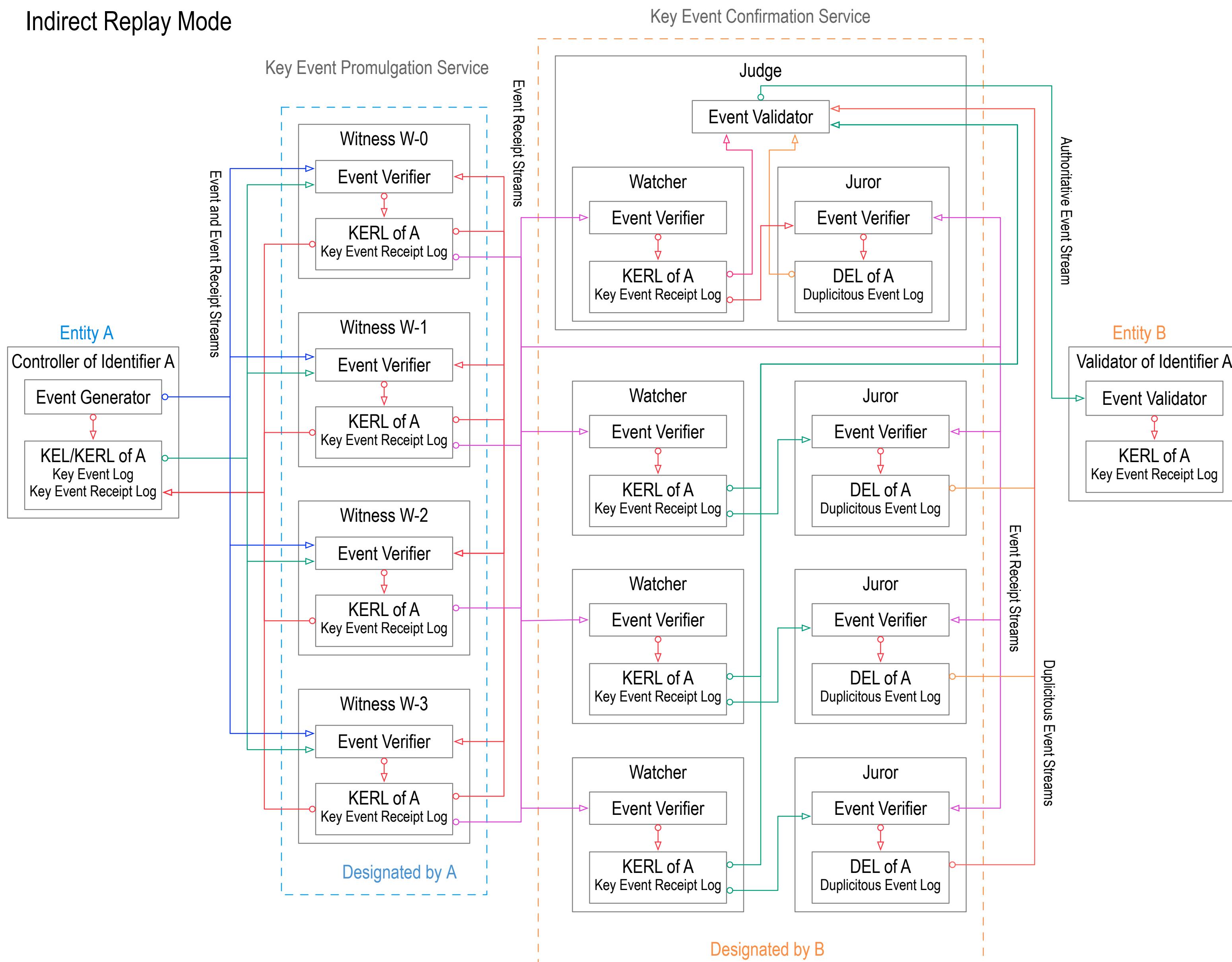


Indirect Mode Promulgation and Confirmation Services



Indirect Mode Full

Indirect Replay Mode



Separation of Control

Shared (permissioned) ledger = *shared control* over *shared data*.

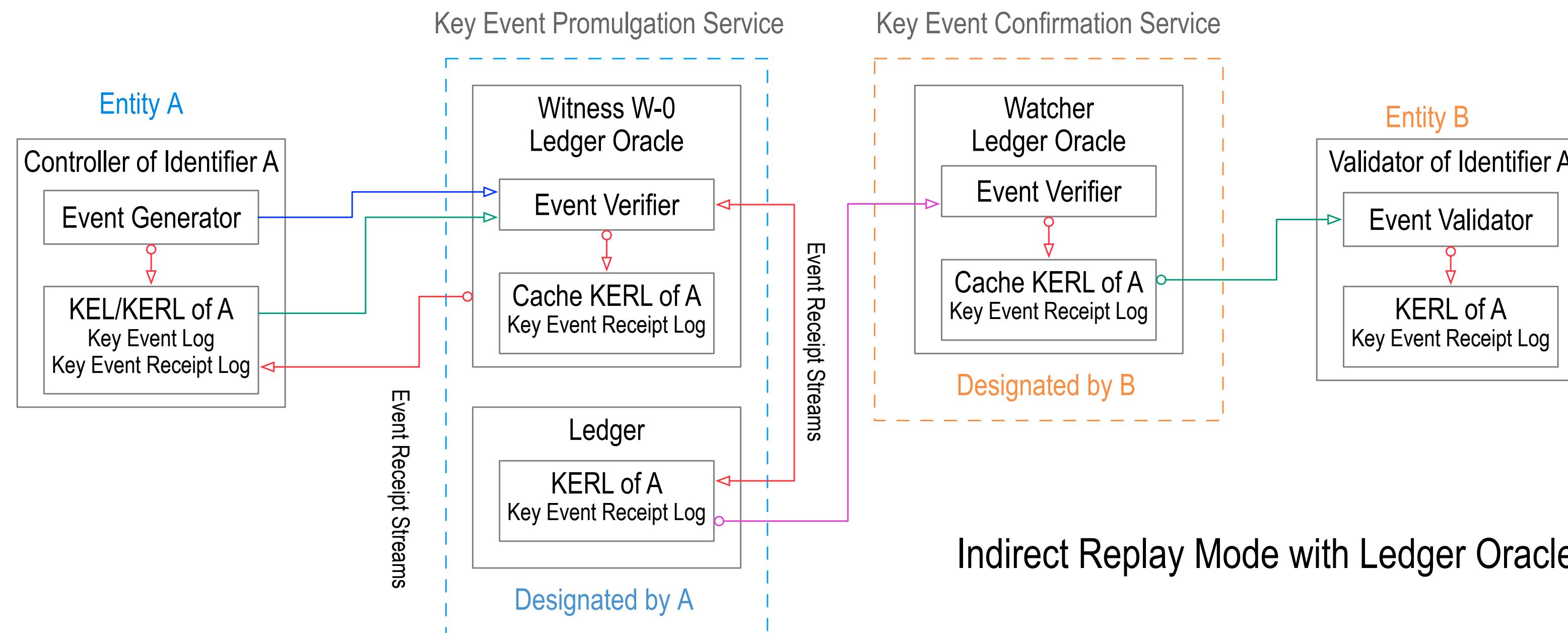
Shared *data* = good, shared *control* = bad.

Shared control between controller and validator may be problematic for governance, scalability, and performance.

KERI = *separated control* over *shared data*.

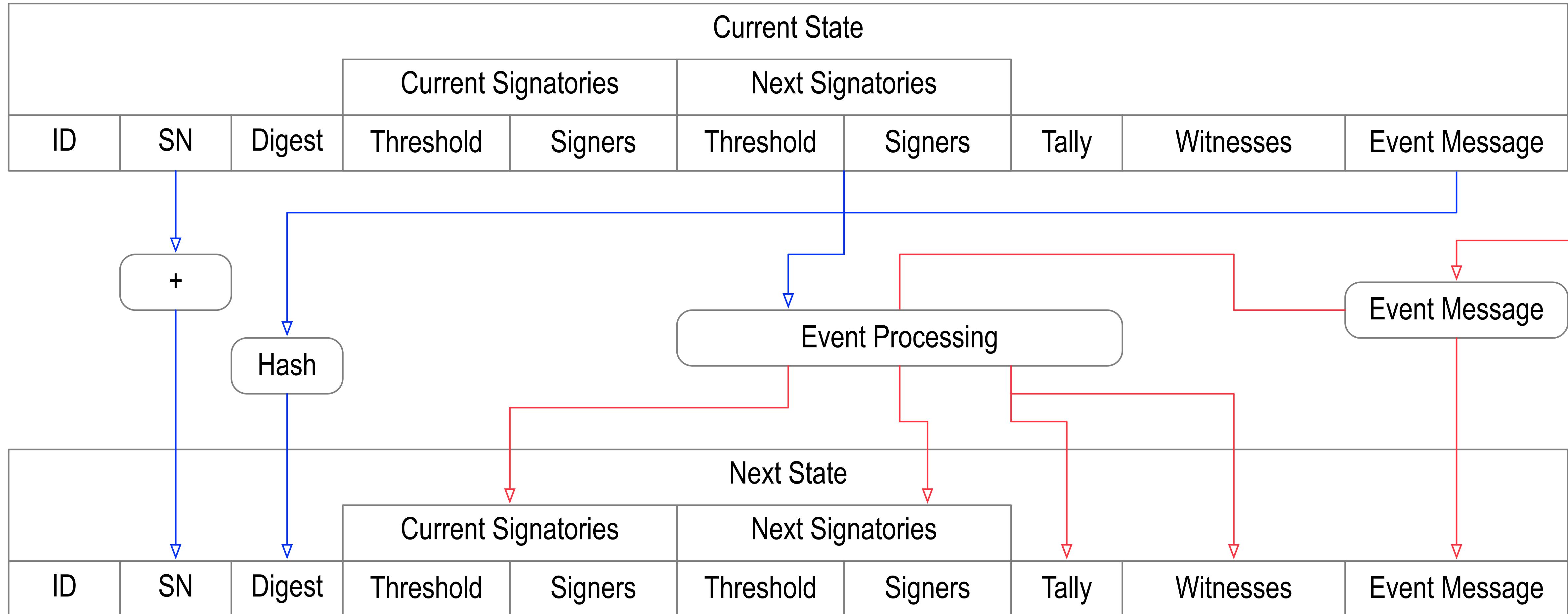
Separated control between controller and validator may provide better decentralization, more flexibility, better scalability, lower cost, higher performance, and more privacy at comparable security.

Indirect Mode with Ledger Oracles



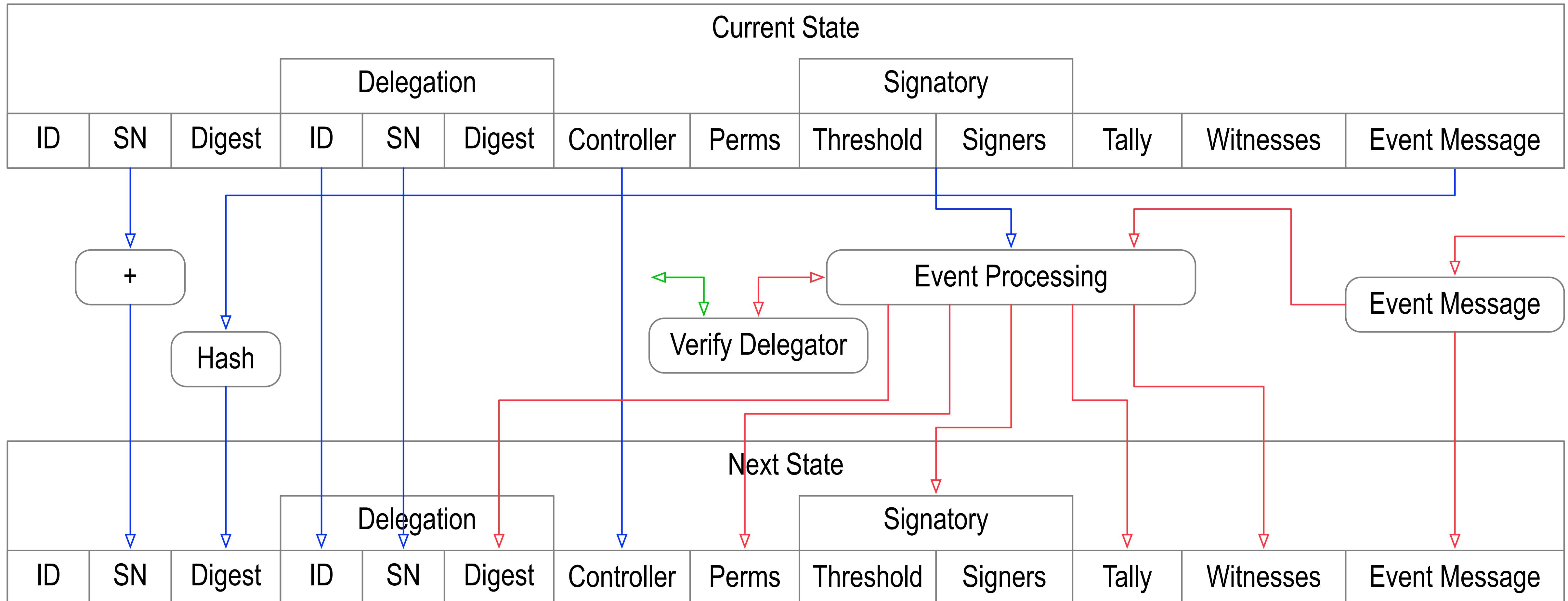
State Verifier Engine

KERI Core — State Verifier Engine

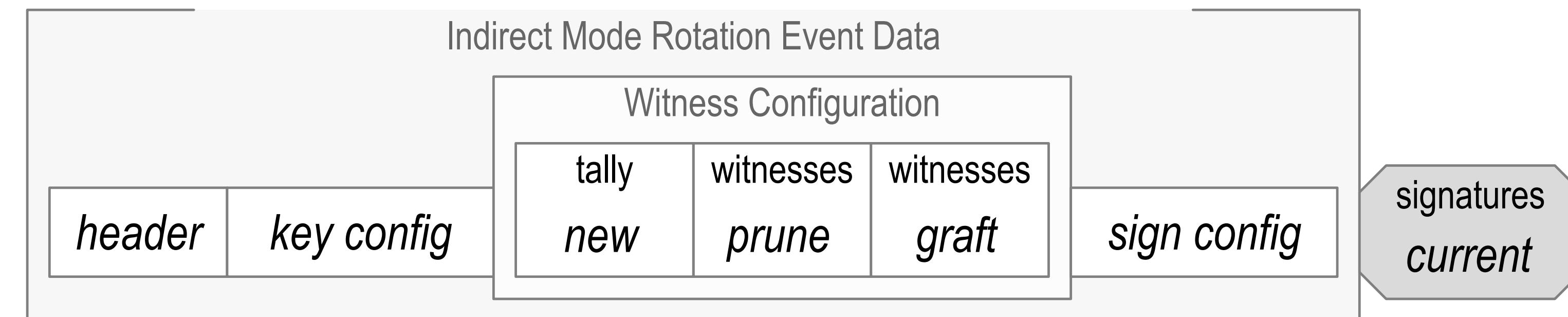
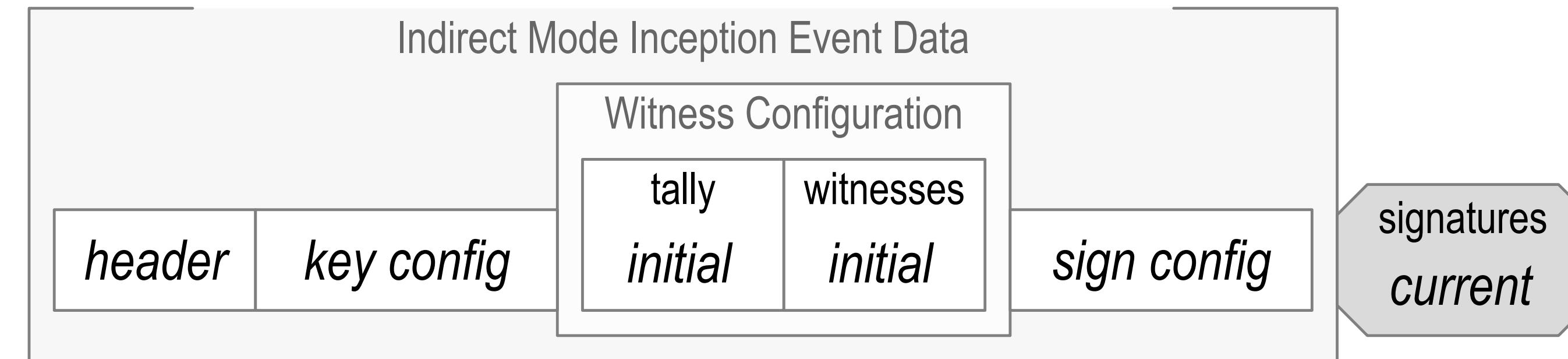


Delegated State Verifier Engine

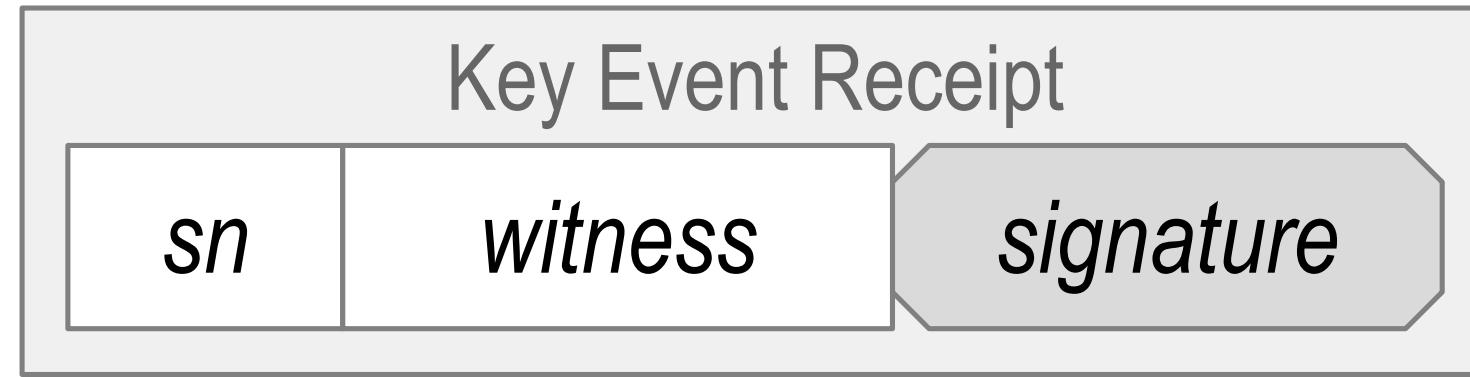
KERI Delegated Core — State Verifier Engine



Witness Designation



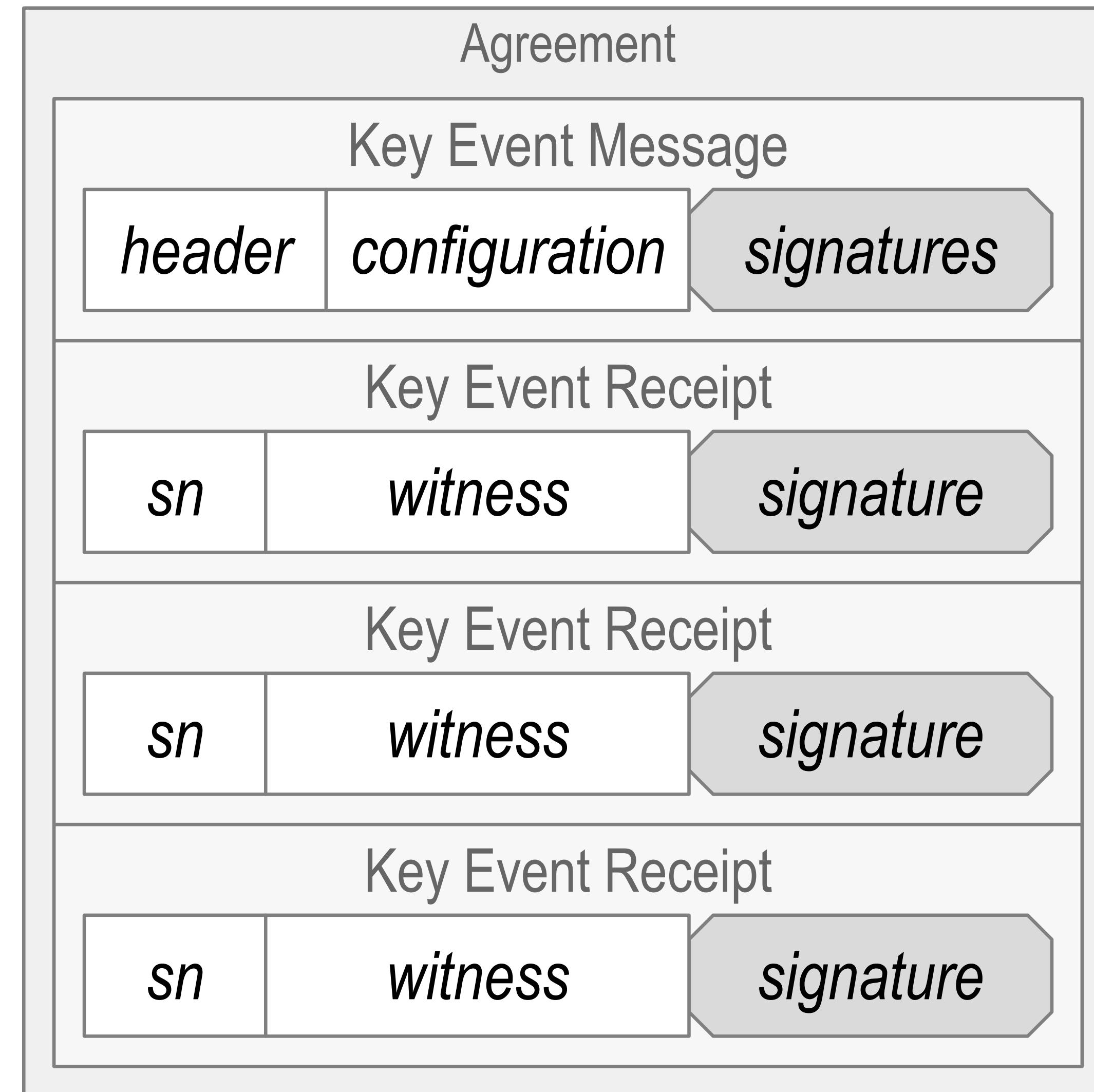
Witnessed Key Event Receipt



(KA²CE)

Keri's Agreement Algorithm for Control Establishment

Produce Agreements
with Guarantees



Agreement Constraints

Proper Agreement

$$F + 1$$

Sufficient Agreement

$$M > F$$

$$M \leq N - F$$

$$F < M \leq N - F$$

Intact Agreement

$$N \geq 2F + 1$$

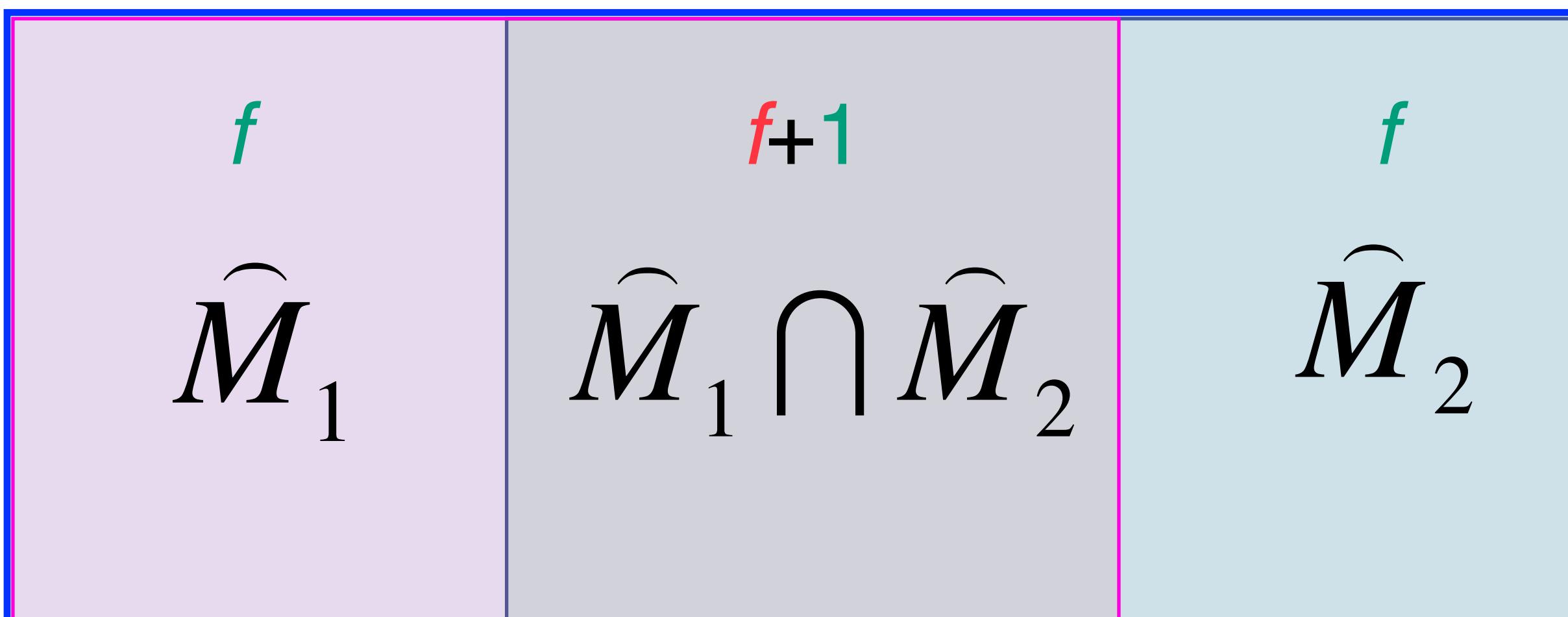
One Agreement or None at All

$$|\hat{N}| = N \quad |\hat{M}_1| = |\hat{M}_2| = M$$

$$|\hat{M}_1 \cup \hat{M}_2| = |\hat{N}| = N$$

Overlapping Sets

$$\hat{M}_1 \cup \hat{M}_2 = \hat{N}$$



One honest witness if:

$$|\hat{M}_1 \cap \hat{M}_2| \geq F + 1$$

$$|\hat{M}_1| + |\hat{M}_2| = |\hat{M}_1 \cup \hat{M}_2| + |\hat{M}_1 \cap \hat{M}_2|$$

$$2M = N + F + 1$$

$$M \geq \left\lceil \frac{N + F + 1}{2} \right\rceil$$

$$M \leq N - F$$

Immune Agreement

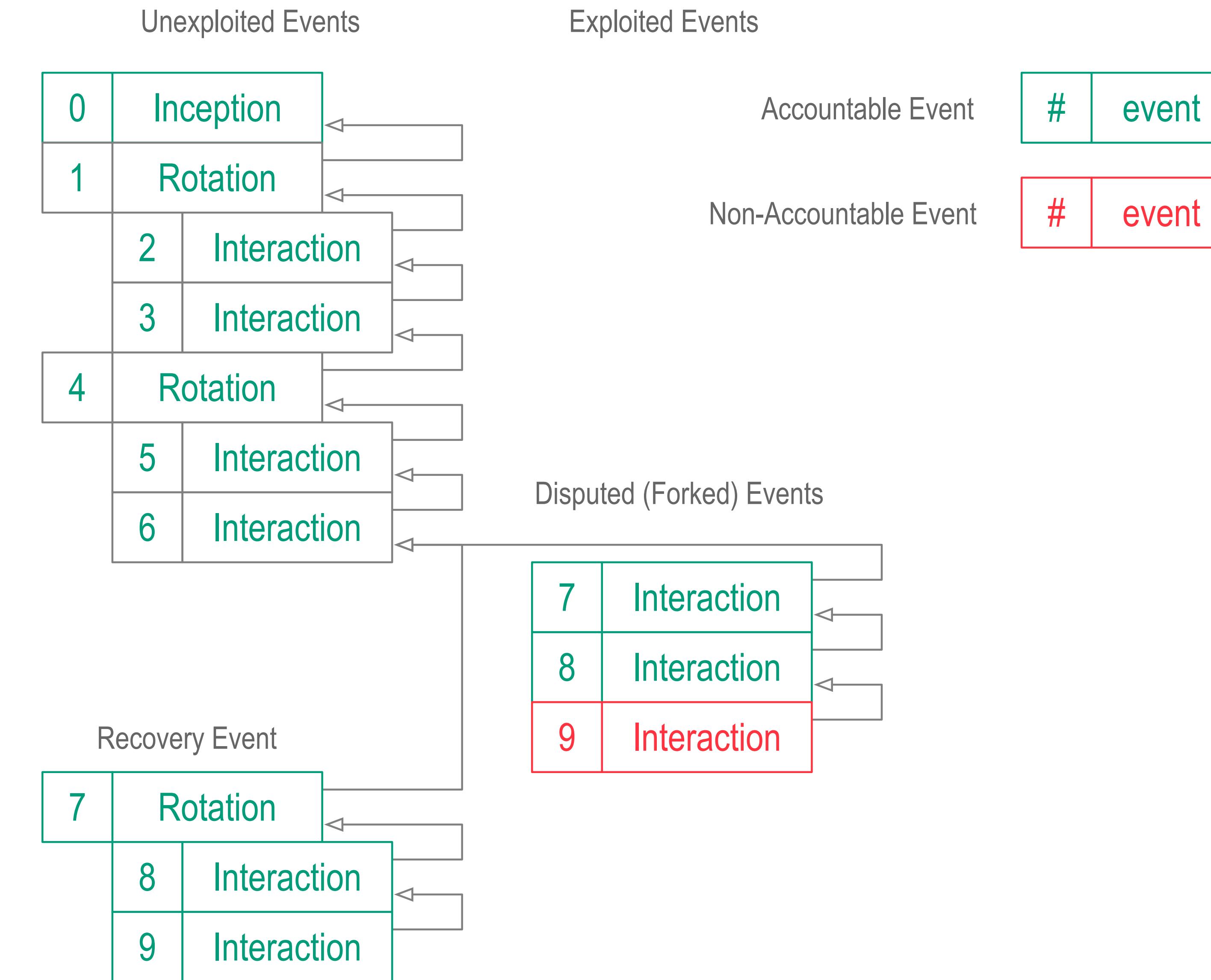
$$\frac{N + F + 1}{2} \leq M \leq N - F$$

Example Values

		Immunity			
F	N	3F+1	$\left\lceil \frac{N+F+1}{2} \right\rceil$	N-F	M
1	4	4	3	3	3
1	5	4	4	4	4
1	6	4	4	5	4, 5
1	7	4	5	6	5, 6
1	8	4	5	7	5, 6, 7
1	9	4	6	8	6, 7, 8
2	7	7	5	5	5
2	8	7	6	6	6
2	9	7	6	7	6, 7
2	10	7	7	8	7, 8
2	11	7	7	9	7, 8, 9
2	12	7	8	10	8, 9, 10
3	10	10	7	7	7
3	11	10	8	8	8
3	12	10	8	9	8, 9
3	13	10	9	10	9, 10
3	14	10	9	11	9, 10, 11
3	15	10	10	12	10, 11, 12

Recovery from Live Exploit

Recovery from Live Exploit



Generic Event Formats

Inception Event Data												
Header				Key Config			Witness Config		Other Config	Sign Config		
version <i>v1</i>	prefix <i>C</i>	sn <i>0</i>	ilk <i>icp</i>	threshold <i>initial</i>	public keys <i>initial</i>	threshold key digest <i>next</i>	tally <i>initial</i>	witnesses <i>initial</i>	data <i>initial</i>	sign indices <i>initial</i>		
Rotation Event Data												
Header					Key Config			Witness Config		Payload Config	Sign Config	
version <i>v1</i>	prefix <i>C</i>	sn <i>1</i>	ilk <i>rot</i>	digest <i>prev</i>	threshold <i>current</i>	public keys <i>current</i>	threshold key digest <i>next</i>	tally <i>new</i>	witnesses <i>prune</i>	witnesses <i>graft</i>	payload <i>details</i>	sign indices <i>current</i>
Interaction Event Data												
Header					Payload Config		Sign Config		signatures <i>current</i>			
version <i>v1</i>	prefix <i>C</i>	sn <i>2</i>	ilk <i>icx</i>	digest <i>prev</i>	payload <i>details</i>		sign indices <i>current</i>		signatures <i>current</i>			

Generic Inception

$$\varepsilon_0^C = \left\langle v_0^C, C, t_0^C, \text{icp}, K_0^C, \hat{C}_0^C, \eta_0^C \left(\langle K_1^C, \hat{C}_1^C \rangle \right), M_0^C, \hat{W}_0^C, \{data\}, \hat{s}_0^C \right\rangle \hat{\sigma}_0^C$$

$$\hat{C}_0^C = \left[C^0, \dots, C^{L_0^C - 1} \right]_0^C$$

$$\hat{C}_1^C = \left[C^{r_1}, \dots, C^{r_1 + L_1^C - 1} \right]_1^C$$

$$\hat{W}_0^C = \left[W_0^C, \dots, W_{N_0^C - 1}^C \right]_0^C$$

$$\hat{s}_0^C = \left[s_0, \dots, s_{S_0^C - 1} \right]_0^C$$

$$\hat{\sigma}_0^C = \sigma_{C^{s_0}} \dots \sigma_{C^{s_{S_0^C - 1}}}$$

Generic Rotation

$$\varepsilon_k^C = \left\langle v_k^C, C, t_k^C, \eta_k^C(\varepsilon_{k-1}^C), \text{rot}, K_l^C, \hat{C}_l^C, \eta_l^C(\langle K_{l+1}^C, \hat{C}_{l+1}^C \rangle), M_l^C, \hat{X}_l^C, \hat{Y}_l^C, \{data\}, \hat{s}_{kl}^C \right\rangle \hat{\sigma}_{kl}^C$$

$$\hat{C}_l^C = \left[C^{r_l^C}, \dots, C^{r_l^C + L_l^C - 1} \right]_l^C$$

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

$$\hat{X}_l^C = \left[X_0^C, \dots, X_{O_l^C - 1}^C \right]_l^C$$

$$\hat{Y}_l^C = \left[Y_0^C, \dots, Y_{P_l^C - 1}^C \right]_l^C$$

$$\hat{s}_{kl}^C = \left[s_0, \dots, s_{S_{kl}^C - 1} \right]_{kl}^C$$

$$\hat{\sigma}_{kl}^C = \sigma_{C^{r_l^C + s_0}} \dots \sigma_{C^{r_l^C + s_{S_{kl}^C - 1}}}$$

Inception Delegation

$$\widehat{\Delta}_0^D = \left\{ D, t_0^D, \eta_k^C \left(\widehat{\delta}_0^D \right) \right\}$$

$$\widehat{\delta}_0^D = \left\langle v_0^D, D, t_0^D, \mathbf{dip}, perms, K_0^D, \widehat{D}_0^D, M_0^D, \widehat{W}_0^D \right\rangle$$

$$\widehat{D}_0^D = \left[D^0, \dots, D^{L_0^D - 1} \right]_0^D$$

$$\widehat{W}_0^C = \left[W_0^C, \dots, W_{N_0^C - 1}^C \right]_0^C$$

$$\mathcal{E}_0^D = \left\langle v_0^D, D, t_0^D, \mathbf{dip}, perms, K_0^D, \widehat{D}_0^D, M_0^D, \widehat{W}_0^D, \widehat{\Delta}_k^C, \widehat{s}_0^D \right\rangle \widehat{\sigma}_0^D$$

$$\widehat{\Delta}_k^C = \left\{ C, t_k^C, \eta_0^D \left(\mathcal{E}_k^C \right) \right\}$$

$$\widehat{s}_0^D = \left[s_0, \dots, s_{S_0^D - 1} \right]_0^D$$

$$\widehat{\sigma}_0^D = \sigma_{D^{s_0}} \dots \sigma_{D^{s_{S_0^D - 1}}}$$

Rotation Delegation

$$\widehat{\Delta}_k^D = \left\{ D, t_k^D, \eta_k^C \left(\widehat{\delta}_k^D \right) \right\}$$

$$\widehat{\delta}_k^D = \left\langle v_k^D, D, t_k^D, \eta_k^D \left(\varepsilon_{k-1}^D \right), \text{drt}, \text{perms}, K_l^D, \widehat{D}_l^D, M_l^D, \widehat{X}_l^D, \widehat{Y}_l^D \right\rangle$$

$$\widehat{D}_l^D = \left[D^{r_l^D}, \dots, D^{r_l^D + L_l^D - 1} \right]_l^D$$

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

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

$$\varepsilon_k^D = \left\langle v_k^D, D, t_k^D, \eta_k^D \left(\varepsilon_{k-1}^D \right), \text{drt}, \text{perms}, K_l^D, \widehat{D}_l^D, M_l^D, \widehat{X}_l^D, \widehat{Y}_l^D, \widehat{\Delta}_k^C, \widehat{s}_{kl}^D \right\rangle \widehat{\sigma}_{kl}^D$$

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

$$\widehat{s}_{kl}^D = \left[s_0, \dots, s_{S_{kl}^D - 1} \right]_{kl}^D$$

$$\widehat{\sigma}_{kl} = \sigma_{C + r_l^D + s_0} \dots \sigma_{C + r_l^D + s_{kl}^D - 1}$$

Generic Interaction

$$\varepsilon_k^C = \left\langle v_k^C, C, t_k^C, \eta_k^C(\varepsilon_{k-1}^C), \text{ixn}, \{data\}, \hat{s}_{kl}^C \right\rangle \hat{\sigma}_{kl}^C$$

$$K_l^C$$

$$\hat{C}_l^C = \left[C^{r_l^C}, \dots, C^{r_l^C + L_l^C - 1} \right]_l^C$$

$$\hat{s}_{kl}^C = \left[s_0, \dots, s_{S_{kl}^C - 1} \right]_{kl}^C$$

$$\hat{\sigma}_{kl}^C = \sigma_{C^{r_l^C + s_0}} \dots \sigma_{C^{r_l^C + s_{S_{kl}^C - 1}}}$$

$$\varepsilon_k^D = \left\langle v_k^D, D, t_k^D, \eta_k^D(\varepsilon_{k-1}^D), \text{ixn}, \{data\}, \hat{s}_{kl}^D \right\rangle \hat{\sigma}_{kl}^D$$

Witness Rotations

$$\hat{W}_0 = \left[W_0 \ , W_1 \ , \cdots , W_{N-1} \right]$$

$$\hat{W}_l = \left(\hat{W}_{l-1} - \hat{X}_l \right) \cap \hat{Y}_l$$

$$\hat{X}_l \subseteq \hat{W}_{l-1} \quad \hat{Y}_l \not\subset \hat{W}_{l-1} \quad \hat{X}_l \not\subset \hat{W}_l$$

$$N_l = N_{l-1} - O_l + P_l$$

$$M_l \leq N_l$$

$$\left| \hat{X}_l \right| = O_l \quad \left| \hat{Y}_l \right| = P_l \quad \left| \hat{W}_l \right| = N_l$$

$$\hat{U}_{l-1} \subseteq \hat{W}_{l-1} \quad \left| \hat{U}_{l-1} \right| \geq M_{l-1}$$

$$\hat{U}_l \subseteq \hat{W}_l \quad \left| \hat{U}_l \right| \geq M_l$$

$$\left| \hat{U}_{l-1} \cup \hat{U}_l \right| \leq M_{l-1} + M_l$$

Complex Weighted Signing Thresholds

$$\hat{C}_l = [C_l^1, \dots, C_l^{L_l}]_l$$

$$\hat{K}_l = [U_l^1, \dots, U_l^{L_1}]_l \quad \hat{C} = [C^1, C^2, C^3]$$

$$0 < U_l^j \leq 1$$

$$U_l^j = \frac{1}{K_l}$$

$$\hat{s}_k^l = [s_0, \dots, s_{S_k^l - 1}]_k^l$$

$$\hat{K} = [\frac{1}{2}, \frac{1}{2}, \frac{1}{2}]$$

$$\bar{U}_l = \sum\nolimits_{i=s_0}^{s_{S_k-1}} U_l^i \geq 1$$

$$\hat{K}_l = [\frac{1}{2}, \frac{1}{2}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}]_l$$

$$\hat{K}_l = [[\frac{1}{2}, \frac{1}{2}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}], [\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}], [1, 1, 1, 1]]$$

BACKGROUND

Derivation Code Tables

Length of crypt material determines number of pad characters. One character table for one pad char. Two character table for two pad char.

One Character KERI Base64 Prefix Derivation Code

Derivation Code	Prefix Description
-	Skip. Start code with next character.
-	
0	Two character derivation code. Use next character from two character table.
1	Four character derivation code. Use next three characters from four character table.
2	Five character derivation code. Use next four characters from five character table.
3	Six character derivation code. Use next five characters from six character table.
4	Eight character derivation code. Use next seven characters from eight character table.
5	Nine character derivation code. Use next eight characters from nine character table.
6	Ten character derivation code. Use next nine characters from ten character table.

One Character KERI Base64 Prefix Derivation Code-1

Derivation Code	Prefix Description	Data Length Bytes	Pad Length h	Derivation Code Length h	Prefix Length Base64	Prefix Length h Bytes
A	Ed25519 public key. Basic derivation.	32	1	1	44	33
B	Non-transferable prefix using Ed25519 public key. Basic derivation.	32	1	1	44	33
C	ECDSA secp256k1 public key. Basic derivation.	32	1	1	44	33
D	Non-transferable prefix using ECDSA secp256k1 public key. Basic derivation.	32	1	1	44	33
E	Blake3-256 Digest. Self-addressing derivation.	32	1	1	44	33
F	SHA3-256 Digest. Self-addressing derivation.	32	1	1	44	33
G	Blake2b-256 Digest. Self-addressing derivation.	32	1	1	44	33
H	Blake2s-256 Digest. Self-addressing derivation.	32	1	1	44	33
I	SHA2-256 Digest. Self-addressing derivation.	32	1	1	44	33
J	Blake3-256 Digest. Self-addressing delegated derivation.	32	1	1	44	33
K	SHA3-256 Digest. Self-addressing delegated derivation.	32	1	1	44	33
L	Blake2b-256 Digest. Self-addressing delegated derivation.	32	1	1	44	33
M	Blake2s-256 Digest. Self-addressing delegated derivation.	32	1	1	44	33
N	SHA2-256 Digest. Self-addressing derivation.	32	1	1	44	33

Derivation Code Tables

Length of crypt material determines number of pad characters. One character table for one pad char. Two character table for two pad char.

Two Character KERI Base64 Prefix Derivation Code

Derivation Code	Prefix Description	Data Length Bytes	Pad Length h	Derivation Code Length h	Prefix Length Base64	Prefix Length Bytes
0A	Ed25519 signature. Self-signing derivation.	64	2	2	88	66
0B	ECDSA secp256k1 signature. Self-signing derivation.	64	2	2	88	66
0C	Blake3-512 Digest. Self-addressing derivation.	64	2	2	88	66
0D	SHA3-512 Digest. Self-addressing derivation.	64	2	2	88	66
0E	Blake2b-512 Digest. Self-addressing derivation.	64	2	2	88	66
0F	Blake2s-512 Digest. Self-addressing derivation.	64	2	2	88	66
0G	SHA2-512 Digest. Self-addressing derivation.	64	2	2	88	66
0H	Blake3-512 Digest. Self-addressing delegated derivation.	64	2	2	88	66
0I	SHA3-512 Digest. Self-addressing delegated derivation.	64	2	2	88	66
0J	Blake2b-512 Digest. Self-addressing delegated derivation.	64	2	2	88	66
0K	Blake2s-512 Digest. Self-addressing delegated derivation.	64	2	2	88	66
0L	SHA2-512 Digest. Self-addressing derivation.	64	2	2	88	66

Derivation Code	Prefix Description	Data Length Bytes	Pad Length	Derivation Code Length	Prefix Length Base64	Prefix Length Bytes
-	Skip. Start code with next character.					
-						
0	Two character derivation code. Use next character from two character table.					
1	Four character derivation code. Use next three characters from four character table.					
2	Five character derivation code. Use next four characters from five character table.					
3	Six character derivation code. Use next five characters from six character table.					
4	Eight character derivation code. Use next seven characters from eight character table.					
5	Nine character derivation code. Use next eight characters from nine character table.					
6	Ten character derivation code. Use next nine characters from ten character table.					
7						
8						
9						
A	Ed25519 public key. Basic derivation.	32	1	1	44	33
B	Non-transferable prefix using Ed25519 public key. Basic derivation.	32	1	1	44	33
C	ECDSA secp256k1 public key. Basic derivation.	32	1	1	44	33
D	Non-transferable prefix using ECDSA secp256k1 public key. Basic derivation.	32	1	1	44	33
E	Blake3-256 Digest. Self-addressing derivation.	32	1	1	44	33
F	SHA3-256 Digest. Self-addressing derivation.	32	1	1	44	33
G	Blake2b-256 Digest. Self-addressing derivation.	32	1	1	44	33
H	Blake2s-256 Digest. Self-addressing derivation.	32	1	1	44	33
I	SHA2-256 Digest. Self-addressing derivation.	32	1	1	44	33
J	Blake3-256 Digest. Self-addressing delegated derivation.	32	1	1	44	33
K	SHA3-256 Digest. Self-addressing delegated derivation.	32	1	1	44	33
L	Blake2b-256 Digest. Self-addressing delegated derivation.	32	1	1	44	33
M	Blake2s-256 Digest. Self-addressing delegated derivation.	32	1	1	44	33
N	SHA2-256 Digest. Self-addressing derivation.	32	1	1	44	33
O						
P						
Q						
R						
S						
T						
U						
V						
W						
X						
Y						
Z						
a						
b						
c						
d						
e						
f						
g						
h						
i						
j						
k						
l						
m						
n						
o						
p						
q						
r						
s						
t						
u						
v						
w						
x						
y						
z						

Derivation Code Tables

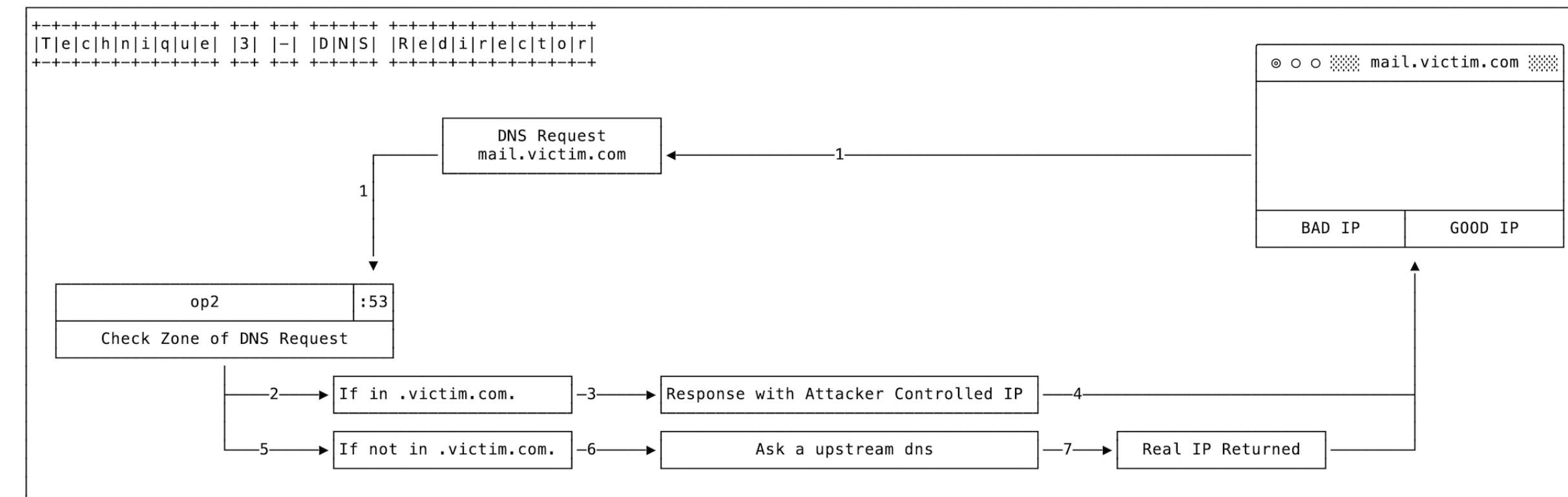
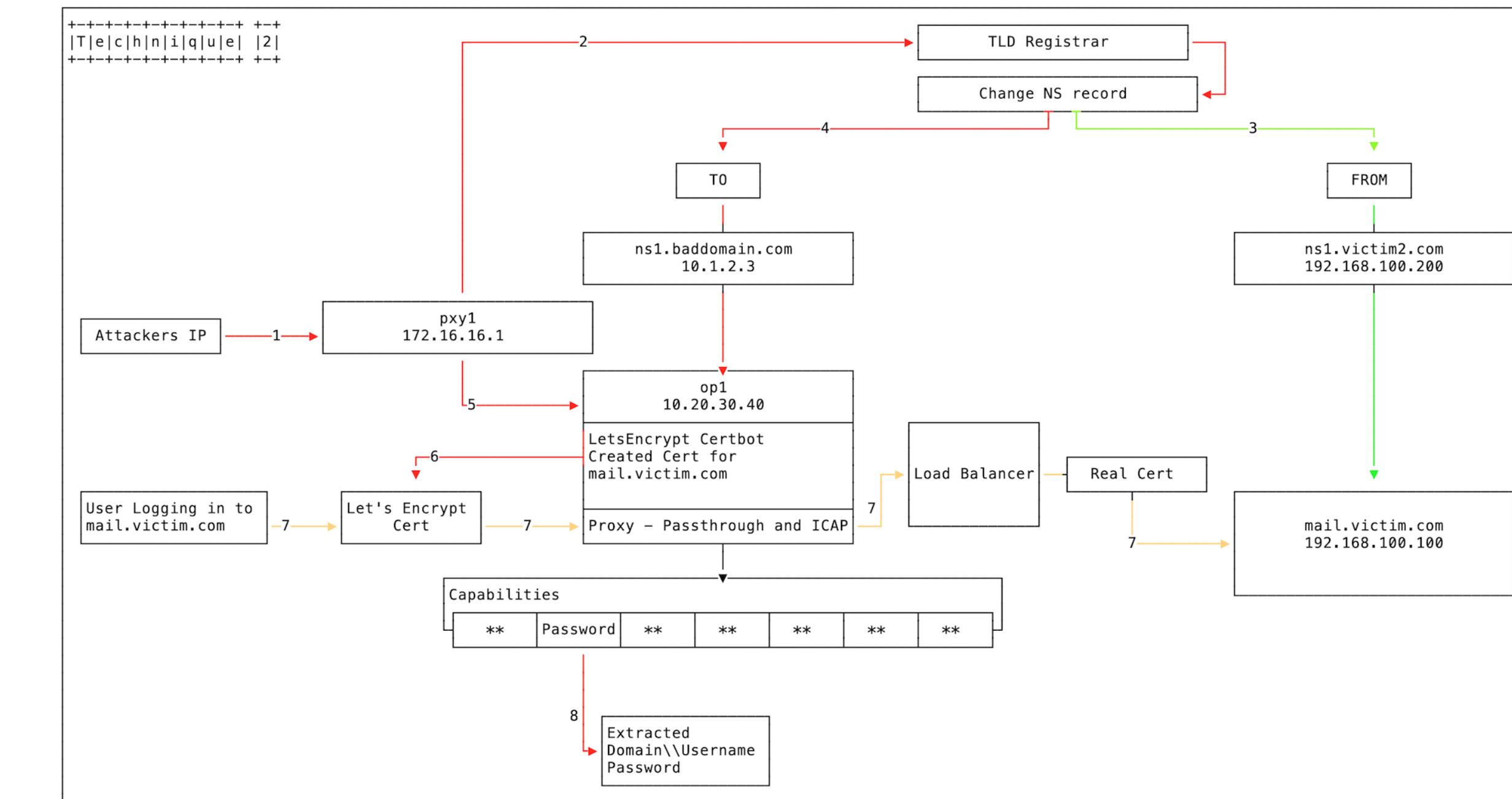
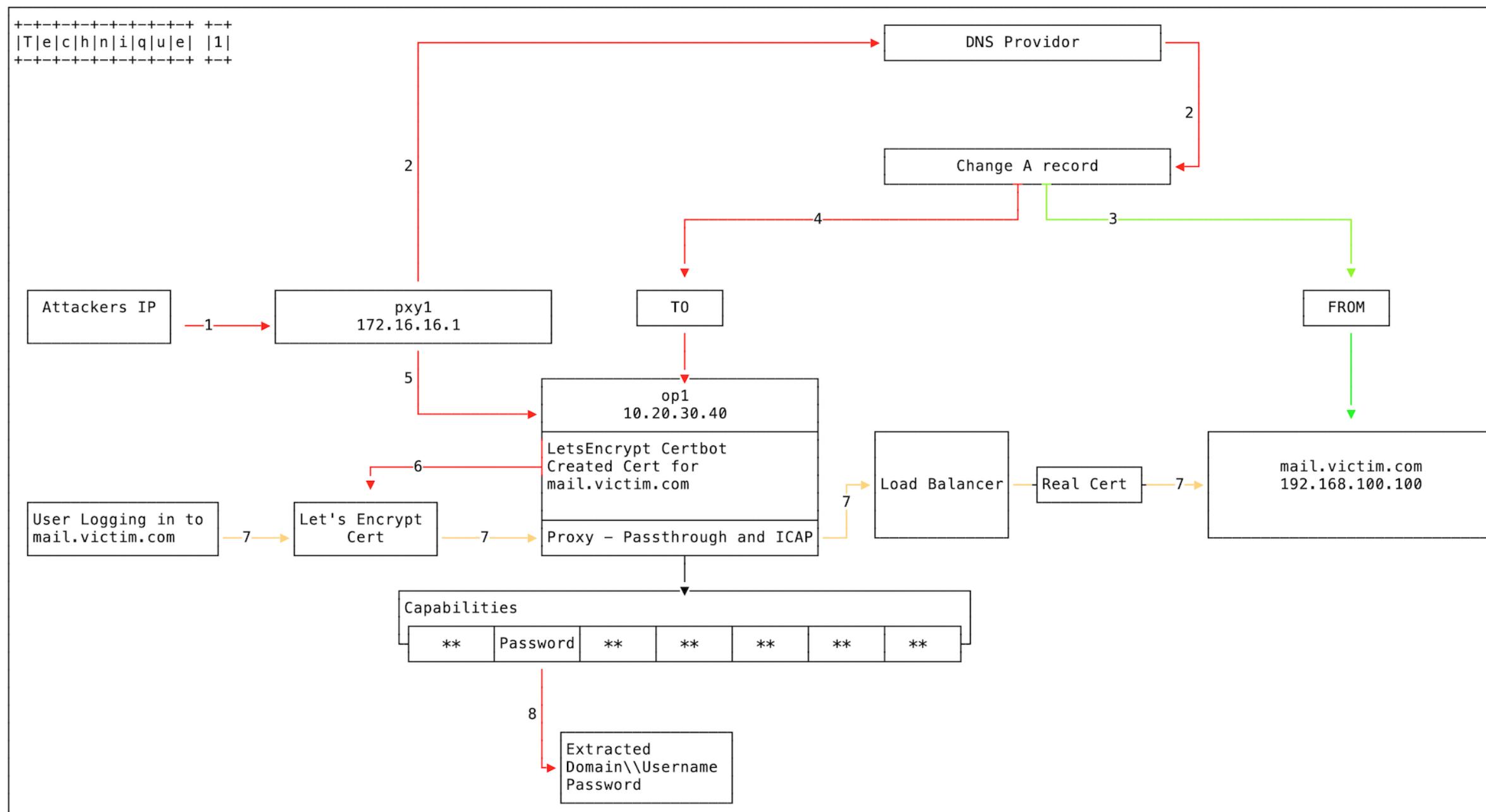
Length of crypt material determines number of pad characters. One character table for one pad char. Two character table for two pad char.

Derivation Code	Prefix Description	Data Length Bytes	Pad Length	Derivation Code Length	Prefix Length Base64	Prefix Length Bytes
0_						
0-						
00						
01						
02						
03						
04						
05						
06						
07						
08						
09						
0A	Ed25519 signature. Self-signing derivation.	64	2	2	88	66
0B	ECDSA secp256k1 signature. Self-signing derivation.	64	2	2	88	66
0C	Blake3-512 Digest. Self-addressing derivation.	64	2	2	88	66
0D	SHA3-512 Digest. Self-addressing derivation.	64	2	2	88	66
0E	Blake2b-512 Digest. Self-addressing derivation.	64	2	2	88	66
0F	Blake2s-512 Digest. Self-addressing derivation.	64	2	2	88	66
0G	SHA2-512 Digest. Self-addressing derivation.	64	2	2	88	66
0H	Blake3-512 Digest. Self-addressing delegated derivation.	64	2	2	88	66
0I	SHA3-512 Digest. Self-addressing delegated derivation.	64	2	2	88	66
0J	Blake2b-512 Digest. Self-addressing delegated derivation.	64	2	2	88	66
0K	Blake2s-512 Digest. Self-addressing delegated derivation.	64	2	2	88	66
0L	SHA2-512 Digest. Self-addressing derivation.	64	2	2	88	66
0M						
0N						
0O						
0P						
0Q						
0R						
0S						
0T						
0U						
0V						
0W						
0X						
0Y						
0Z						
0a						
0b						
0c						
0d						
0e						
0f						
0g						
0h						
0i						
0j						
0k						
0l						
0m						
0n						
0o						
0p						
0q						
0r						
0s						
0t						
0u						
0v						
0w						
0x						
0y						
0z						

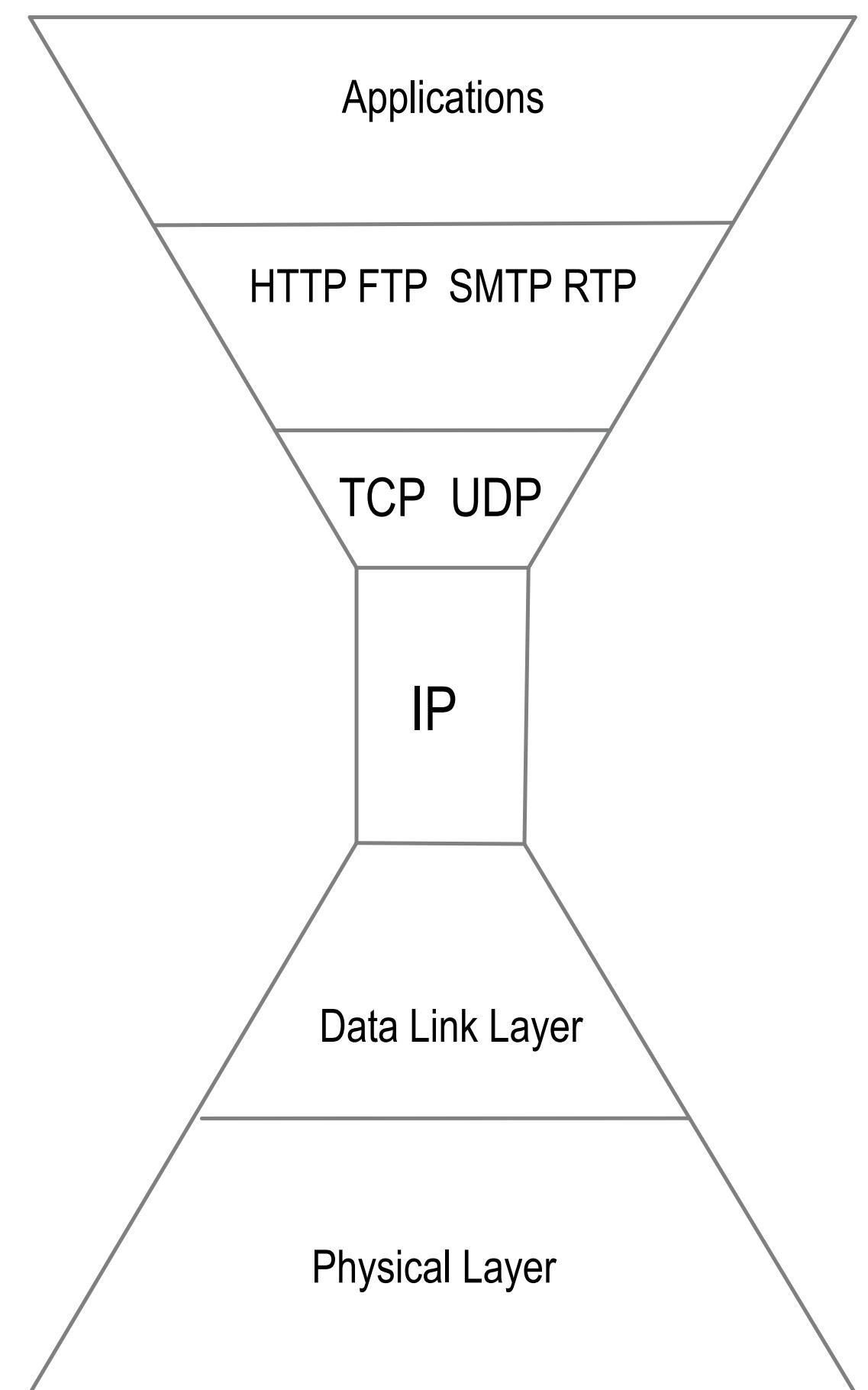
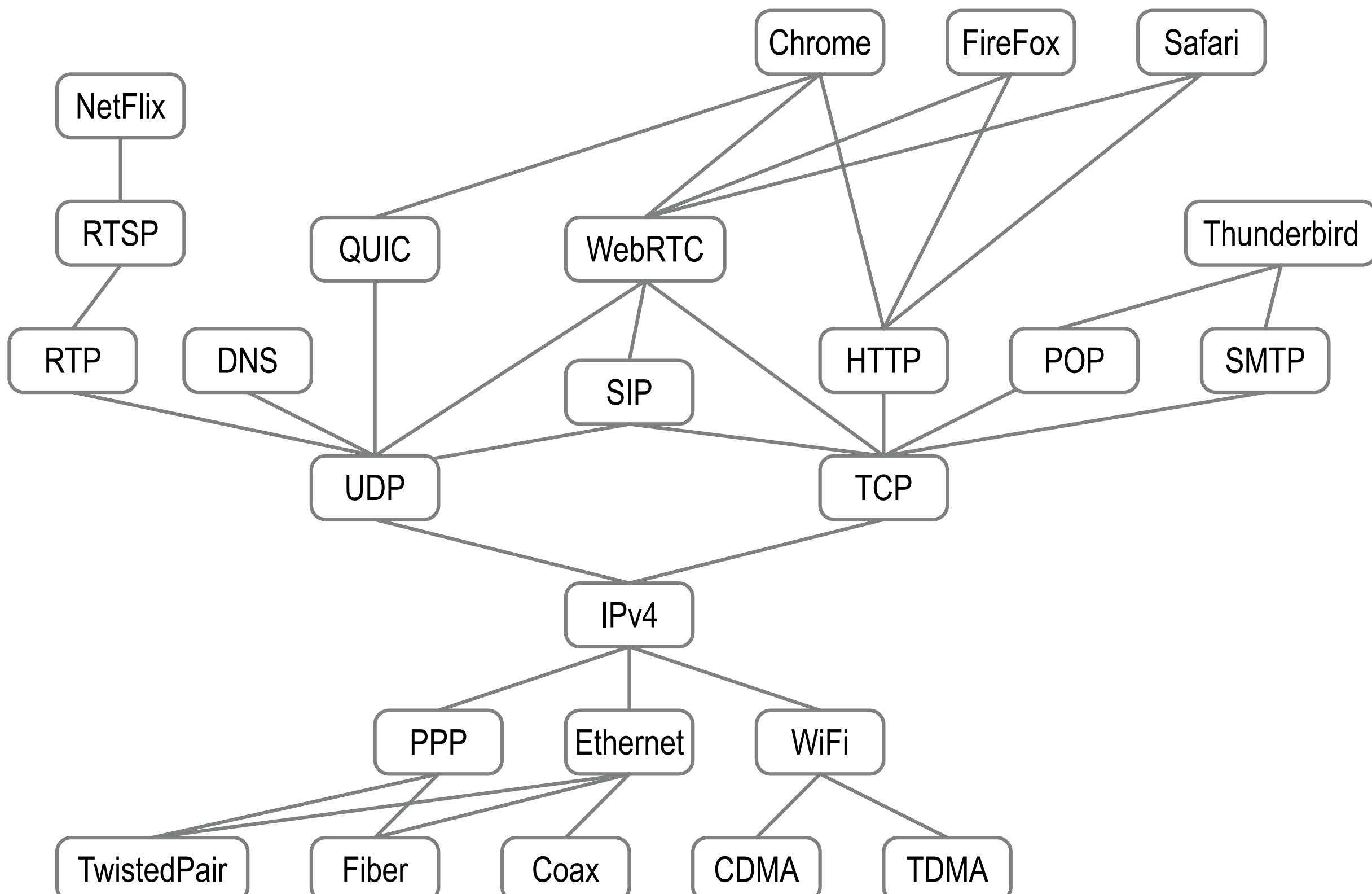
DNS Hijacking

A DNS hijacking wave is targeting companies at an almost unprecedented scale. Clever trick allows attackers to obtain valid TLS certificate for hijacked domains.

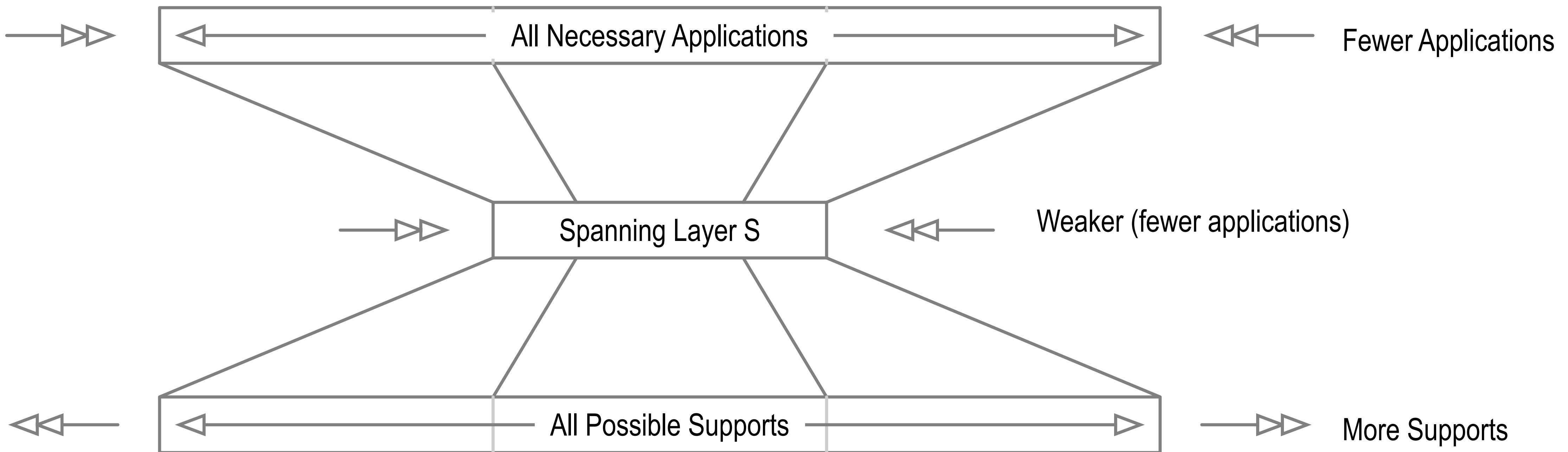
<https://arstechnica.com/information-technology/2019/01/a-dns-hijacking-wave-is-targeting-companies-at-an-almost-unprecedented-scale/>



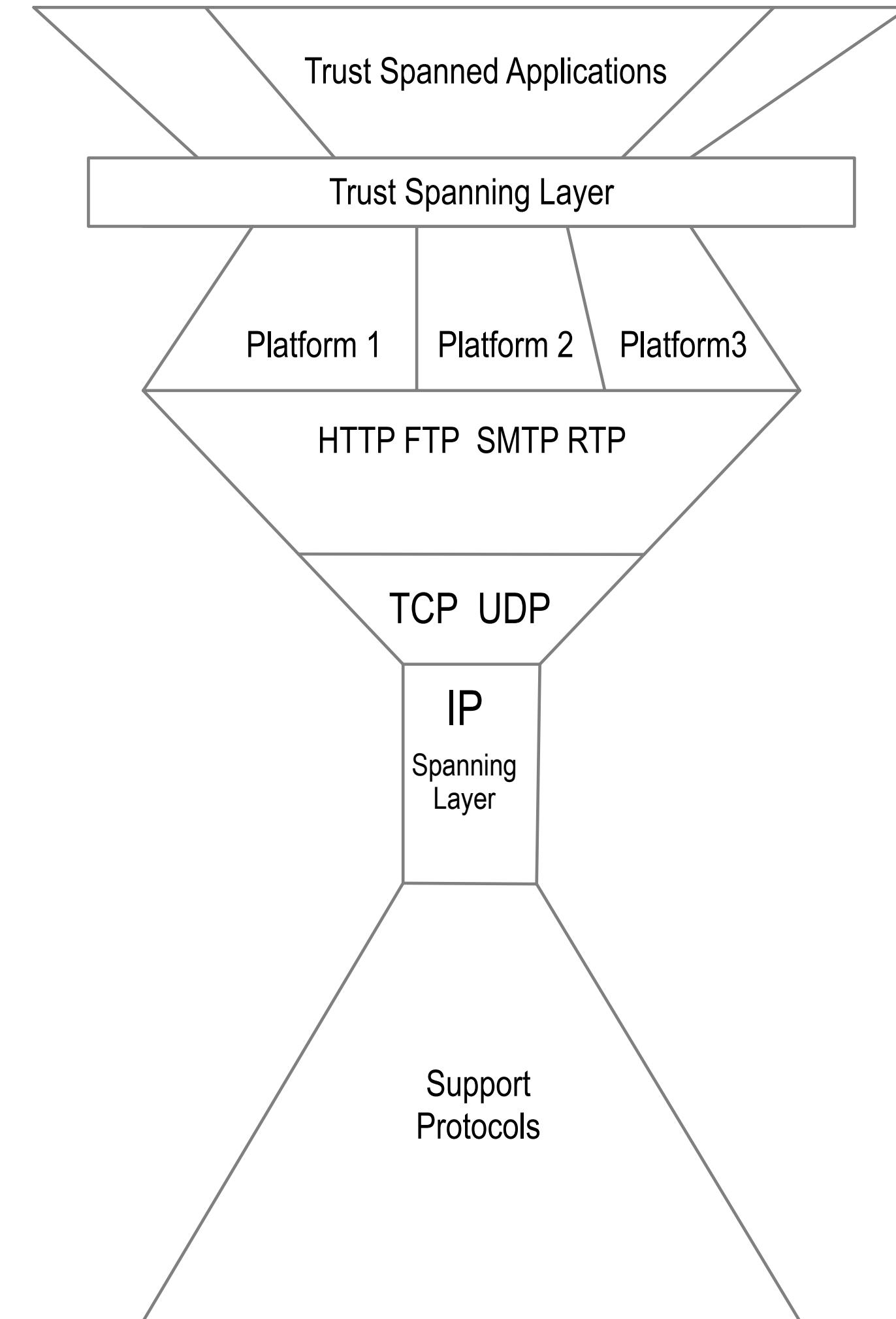
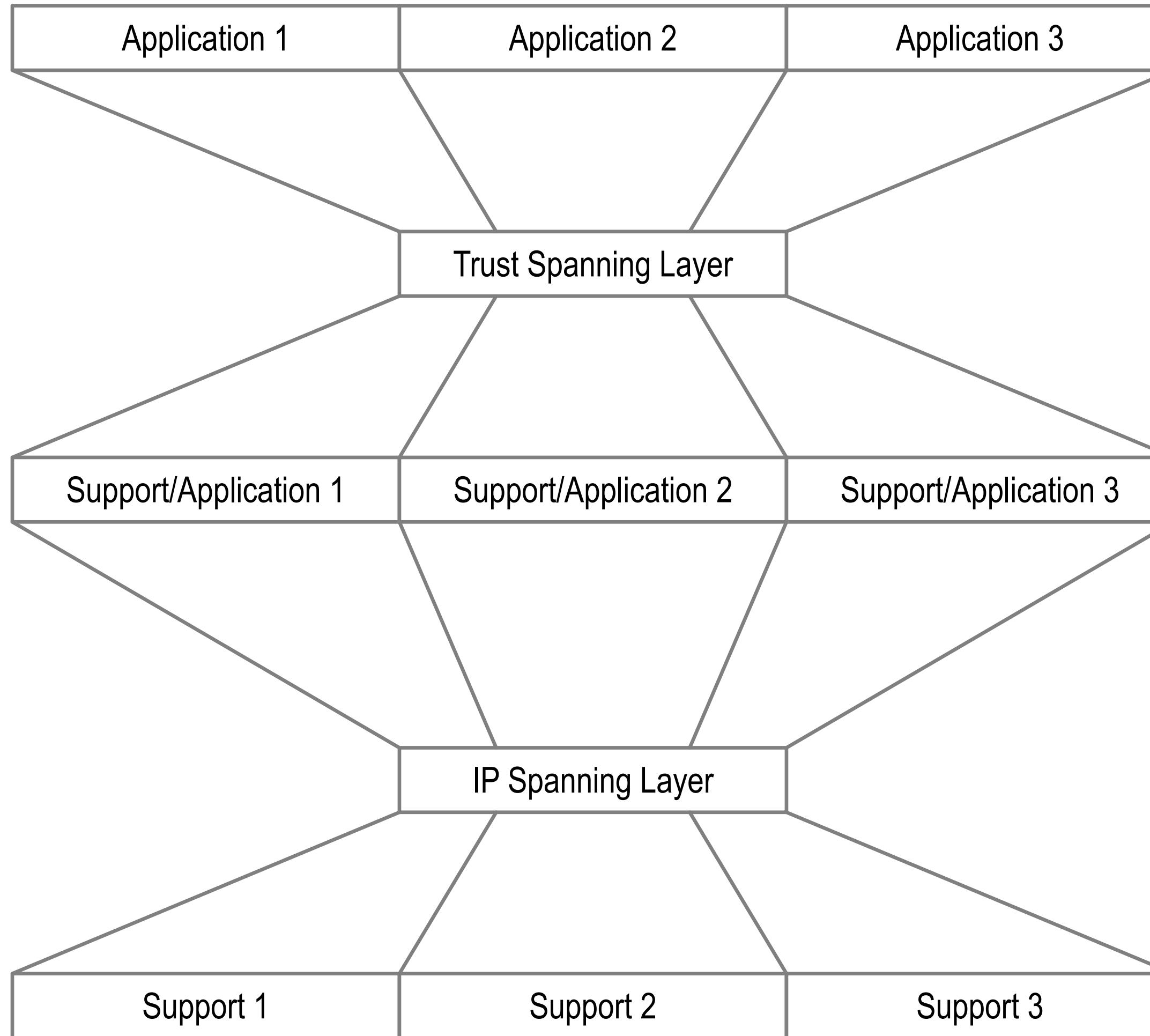
Spanning Layer



Hourglass



Waist and Neck



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 zero trust 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)

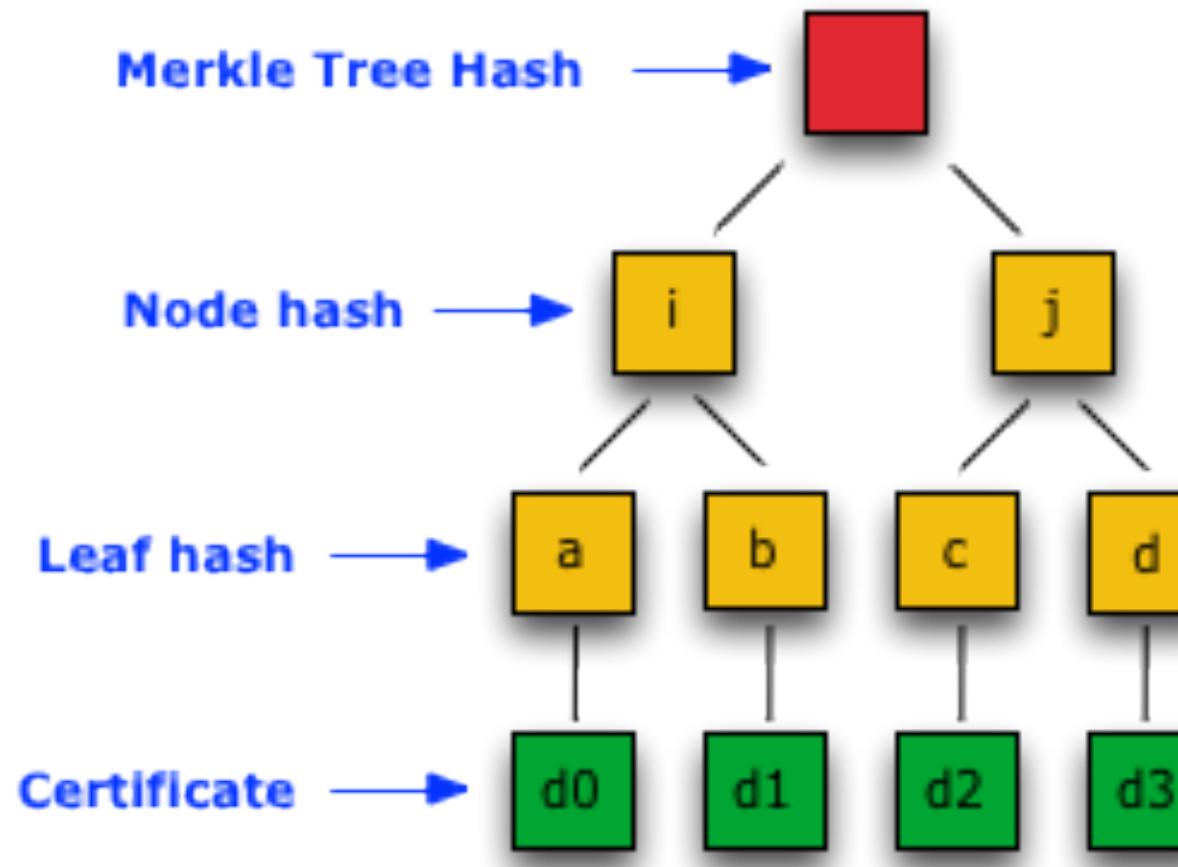


Figure 1

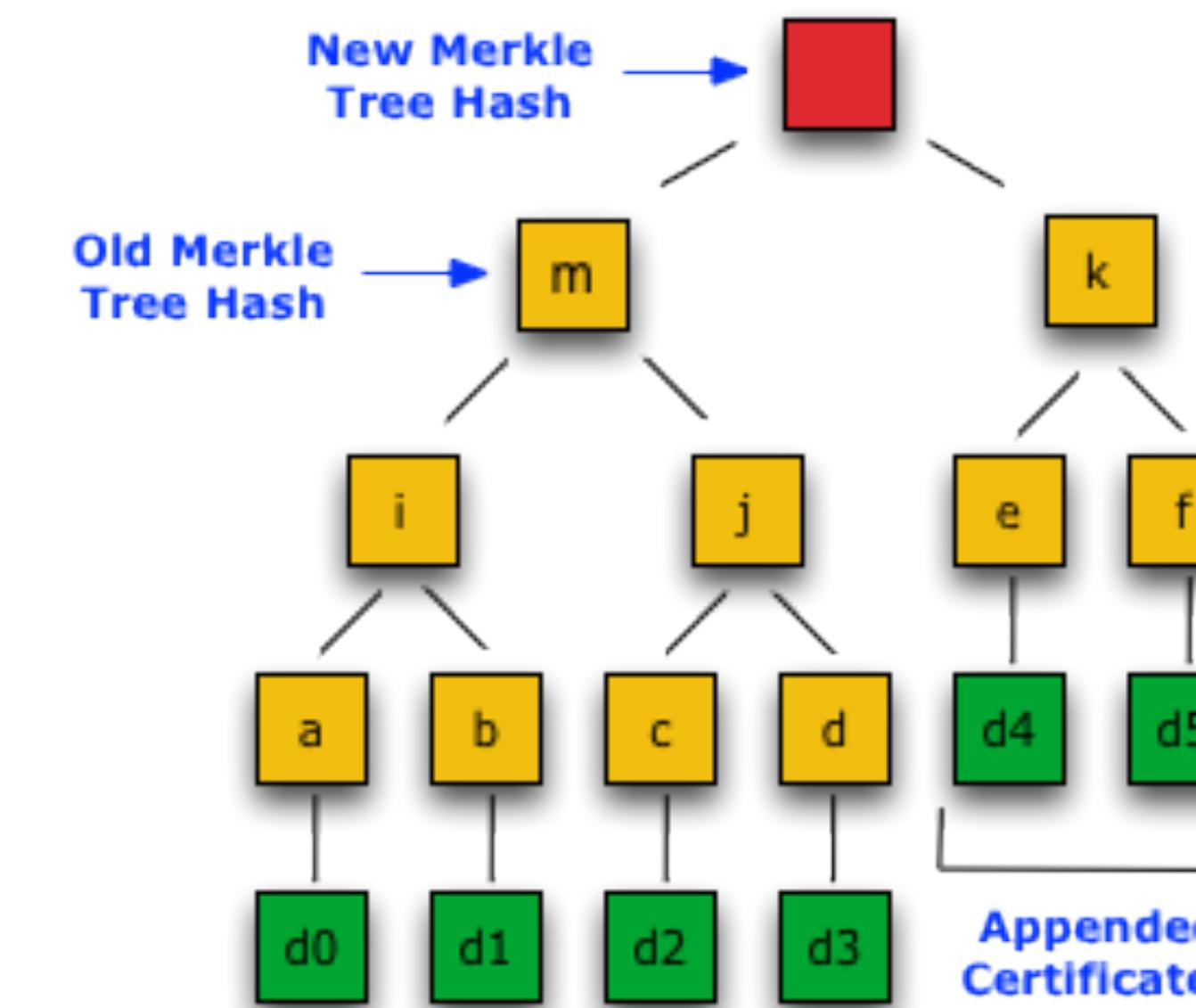


Figure 2

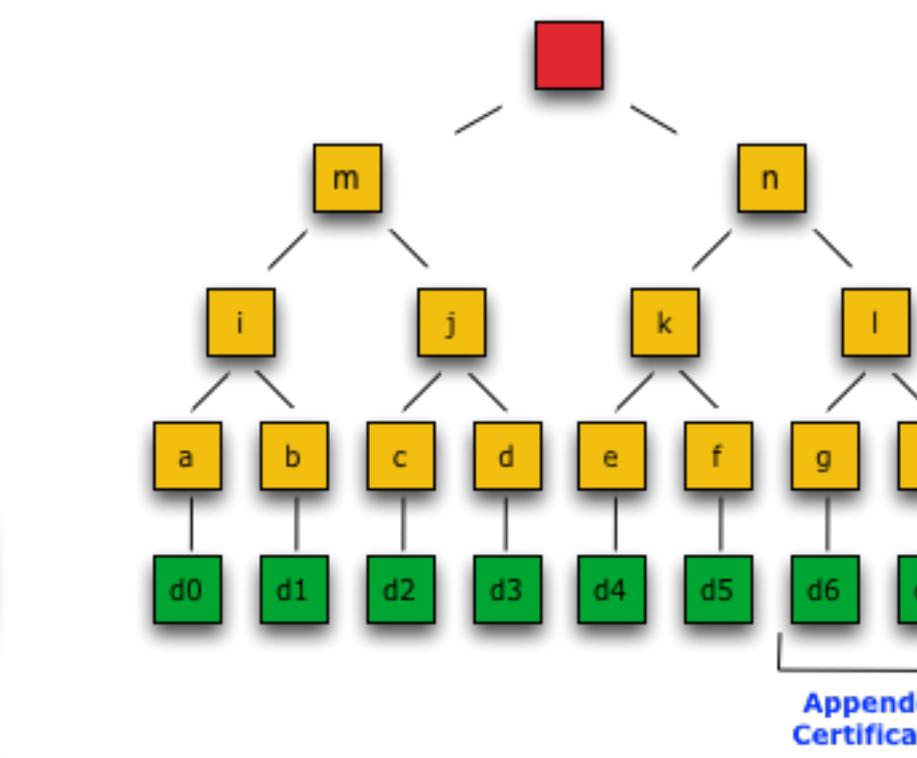


Figure 3

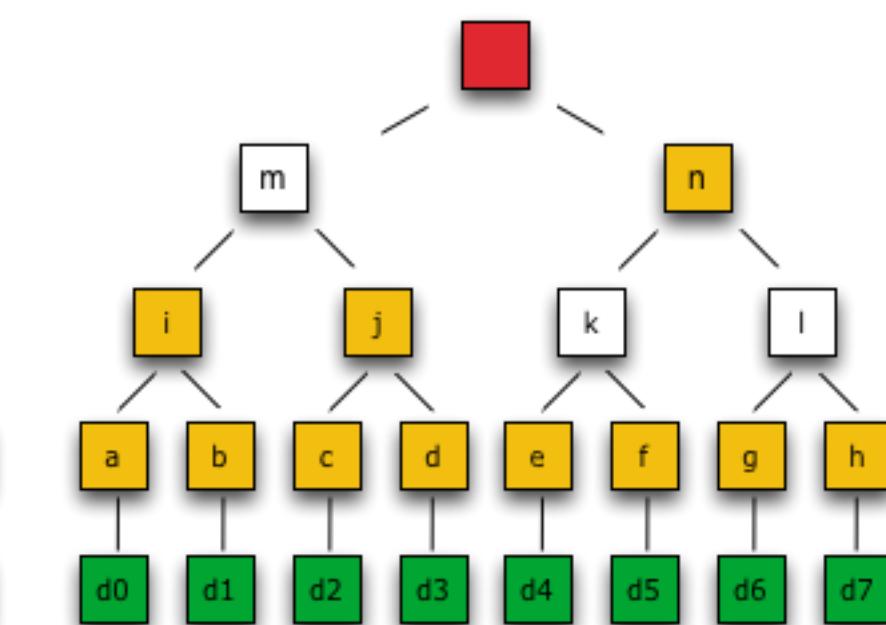


Figure 4

IETF/TCG Alternative Standards Stack

Interoperable Remote Device Configuration Attestation

“Remote ATtestation proceduresS (RATS) WG,” IETF,

<https://datatracker.ietf.org/wg/rats/about/>

Working Group, R. A. T. S., “Network Device Attestation Workflow,” IETF Internet-Draft, 2019/12/03

<https://tools.ietf.org/html/draft-fedorkow-rats-network-device-attestation-01>

Interoperable Authentic Device Configuration Attestations

CBOR/JSON CWT JWT FIDO

Source-of-Truth

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

Only the controller may 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

Loci-of-Control

Who controls **What**

Control over Identifier

Control over Support Infrastructure

Control over Operations on Identifier

AUTONOMIC IDENTIFIER (AID)

auto nomos self rule (autonomic, autonomous)

Self-Governing, Self-Sovereign, Self-Controlling

Self-Authorizing, Self-Certifying

Self-Managing, Self-Administering

AUTONOMIC NAMESPACE (AN)

Prefixed with AID

Extensible

Cryptographically Verifiable Identifier Derivation

“BEST” DECENTRALIZED SECURITY

End Verifiability

Ambient Verifiability = Anyone can verify

End Verifiability + Immutable Log = Duplicity Detection = Consistent Attribution

Ambient Verifiability + Ambient Immutable Logs = Ambient Duplicity Detection

EXPLOIT MANAGEMENT

Control exploit always eventually exhibits as duplicitous behavior:

i.e. inconsistent control statements

Exploit detection and containment via duplicity detection

Scalable performant duplicity detection via immutable logs of control statements

Control exploit potential mitigated via best practices for key management

KEY MANAGEMENT

Rotation = Revoke and Replace = Full Transfer of Control

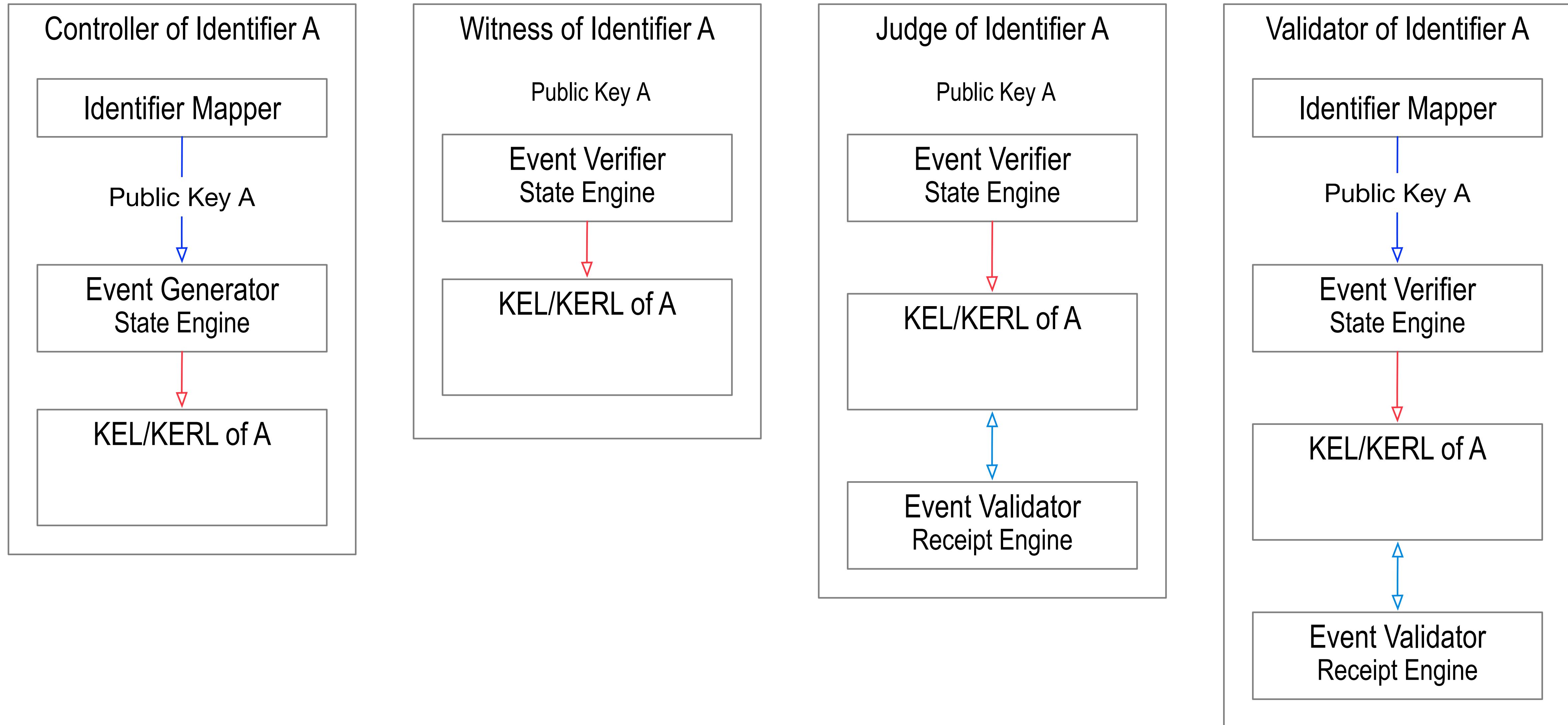
Reproduction = Creation, Derivation

Recovery

BEST PRACTICE

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

Offline Components



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