Verifiable Trust Bases

Renewing the Web of Trust

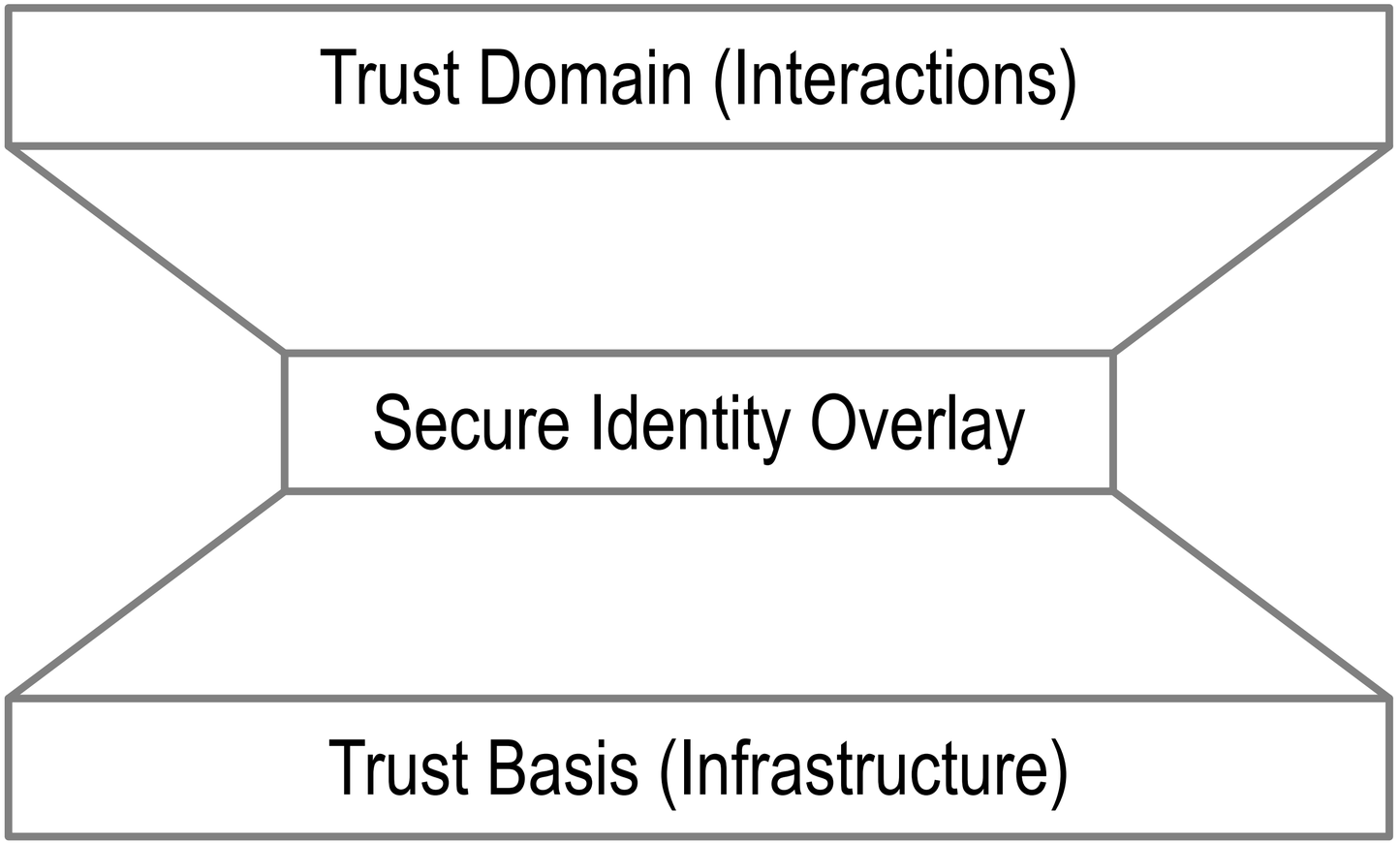
Samuel M. Smith Ph.D.
sam@keri.one
https://keri.one
version 2.53
2020/10/20

Resources

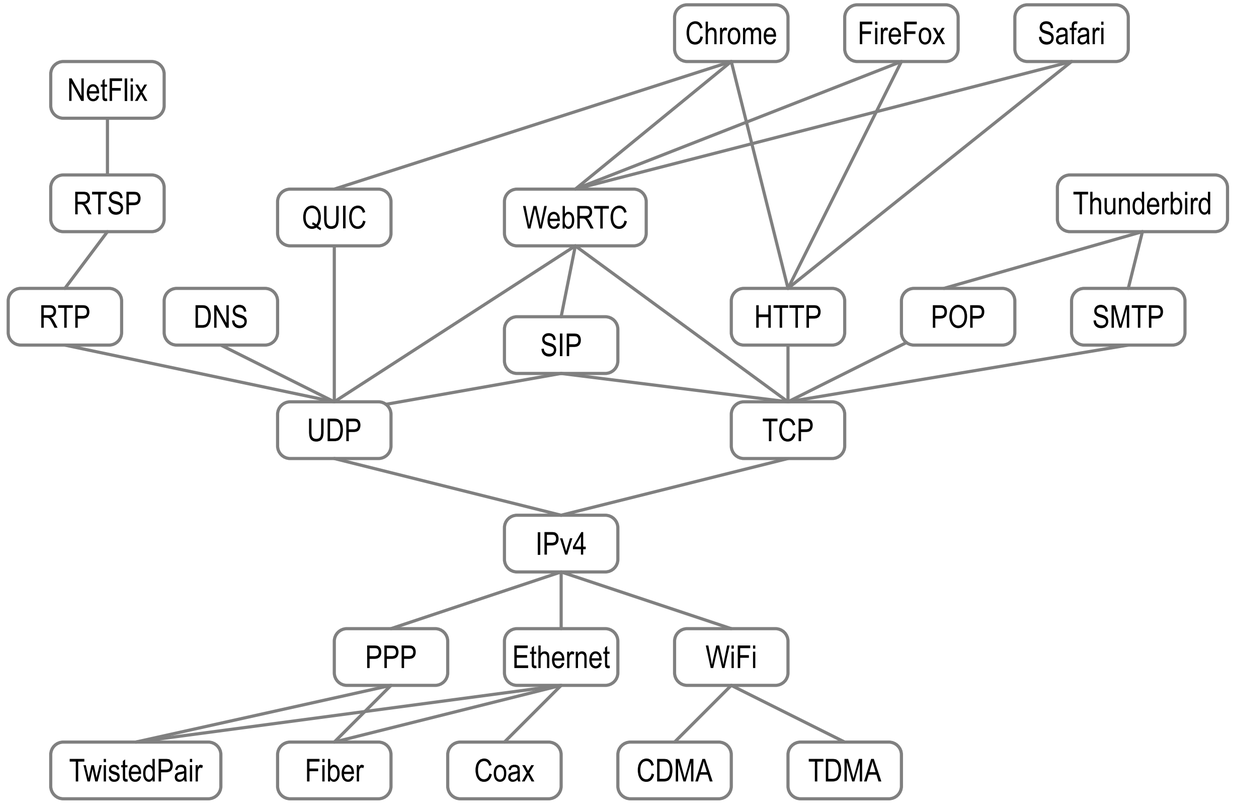
keri.one

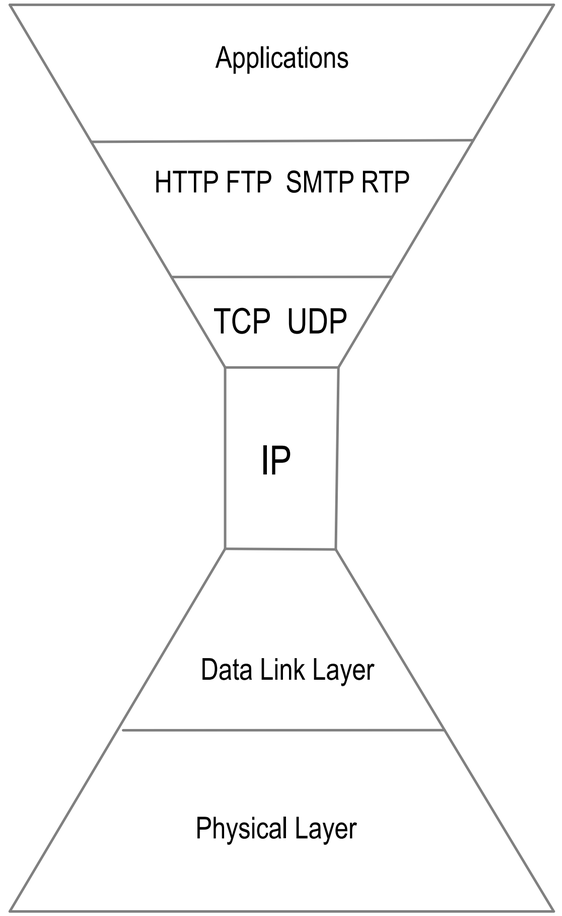
Universal Identifier Theory

Identity System Security Overlay

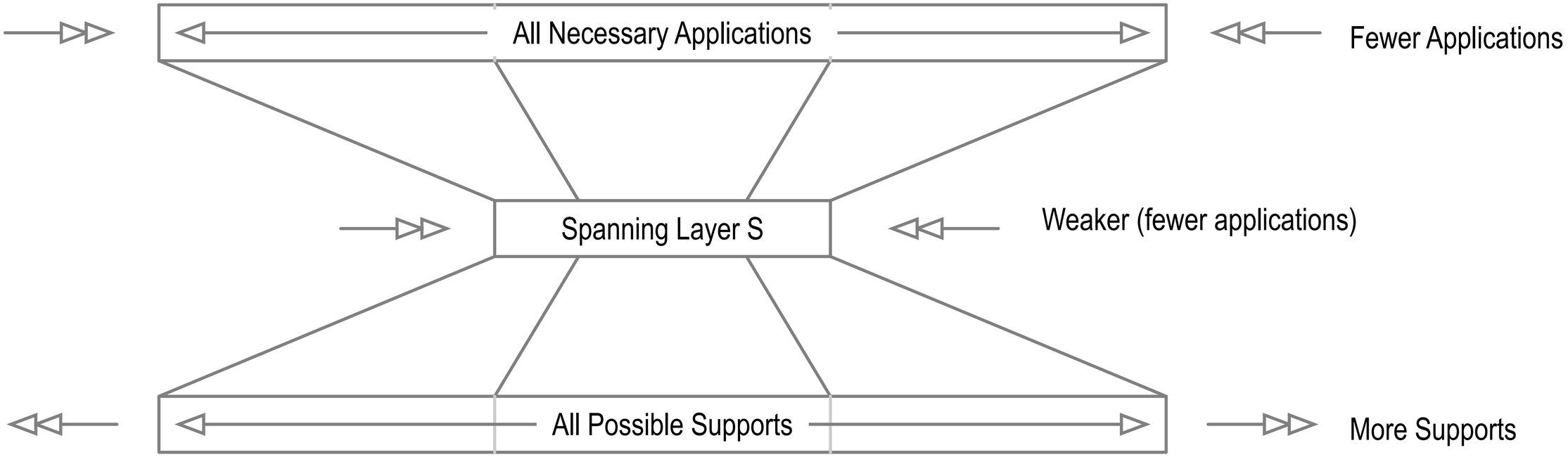


Spanning Layer

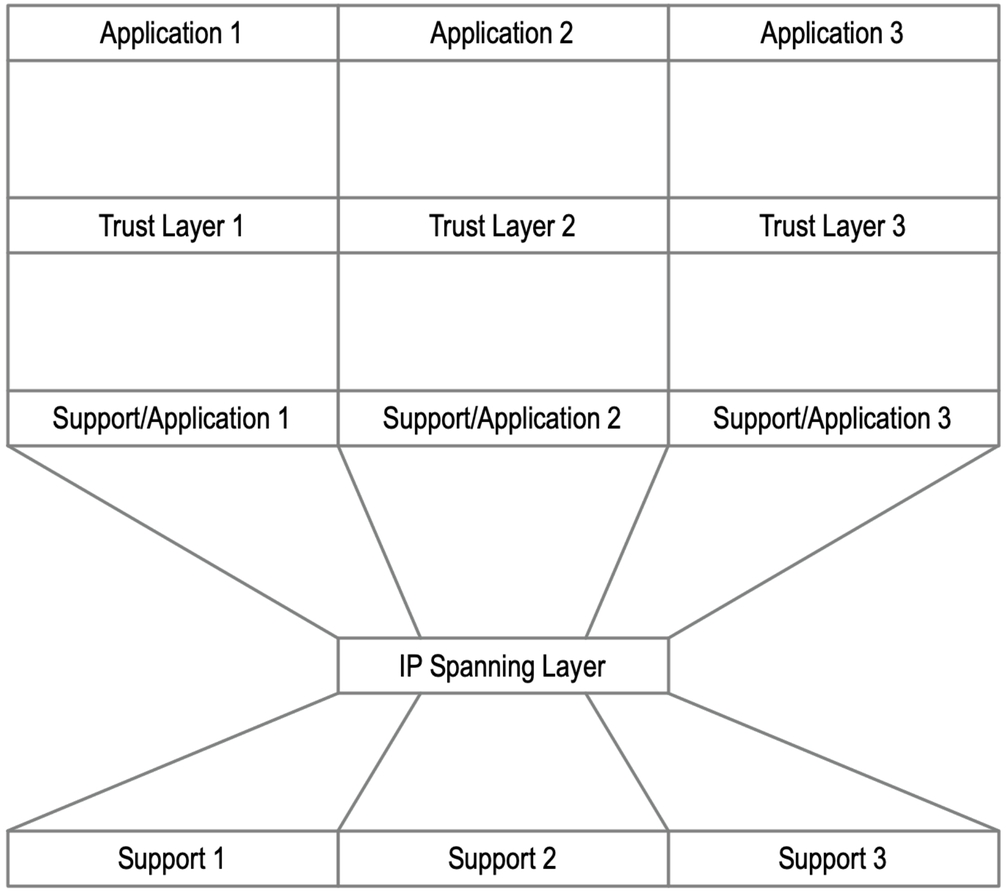




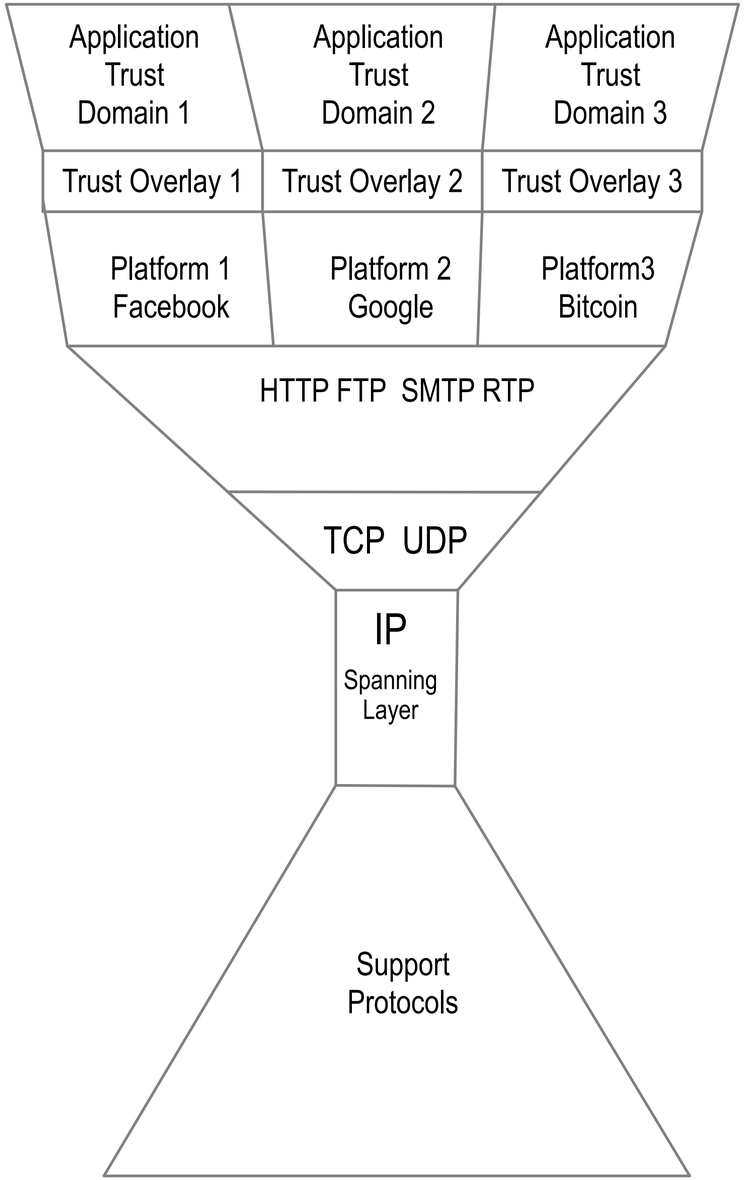
Hourglass


All Necessary Applications
Fewer Applications
Spanning Layer S
Weaker (fewer applications)
All Possible Supports
More Supports

Platform Locked Trust



Trust Domain Based Segmentation

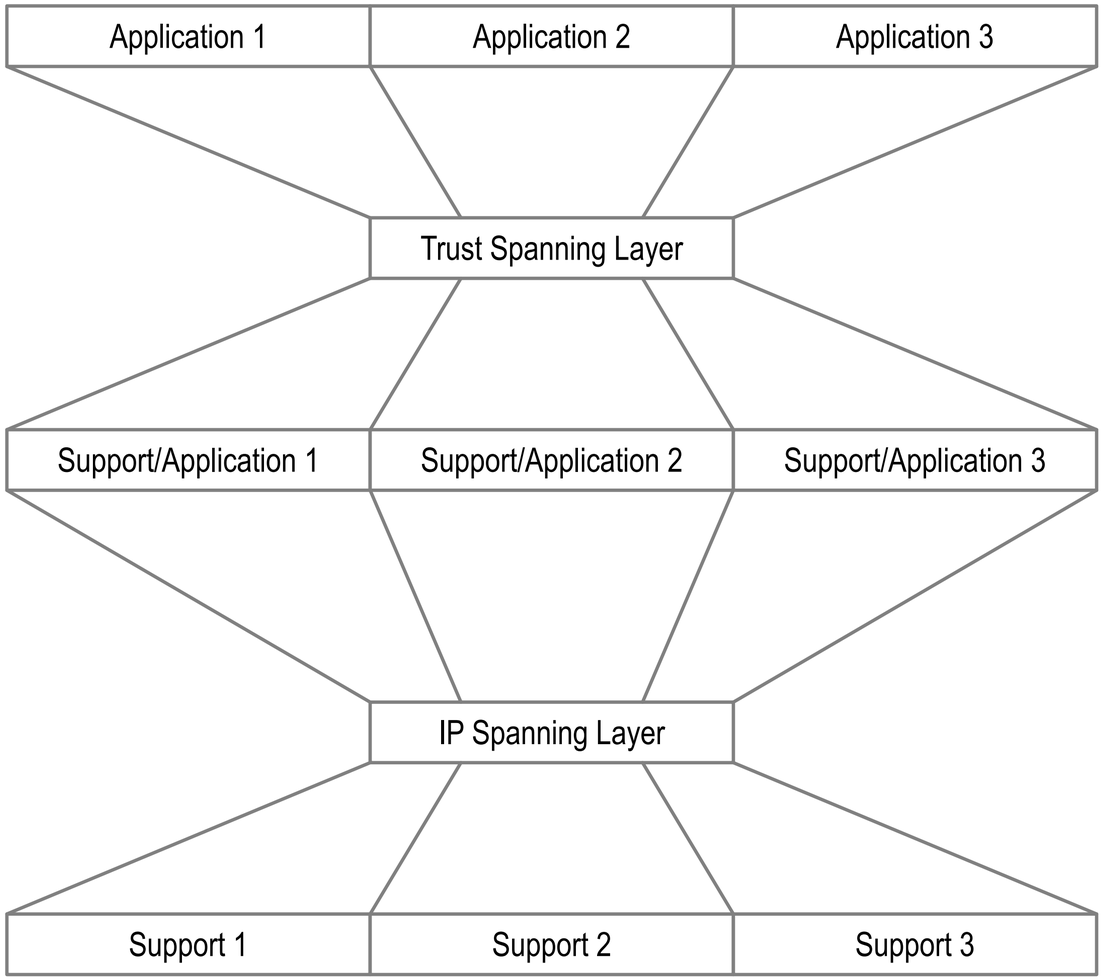


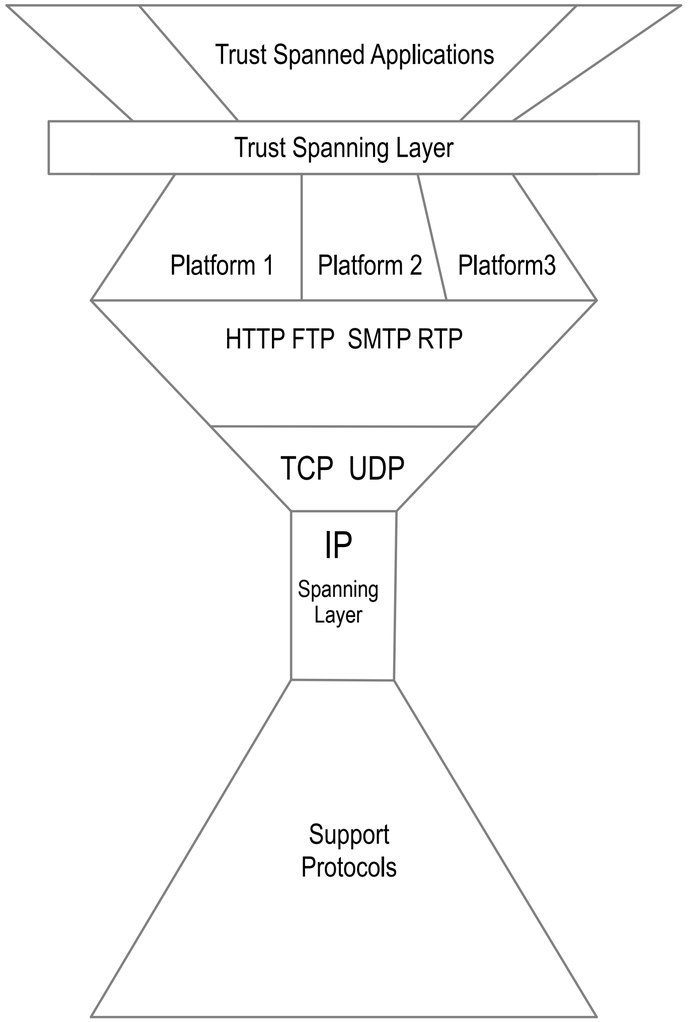
Each trust layer only spans platform specific applications

Bifurcates the internet trust map

No spanning trust layer

Waist and Neck



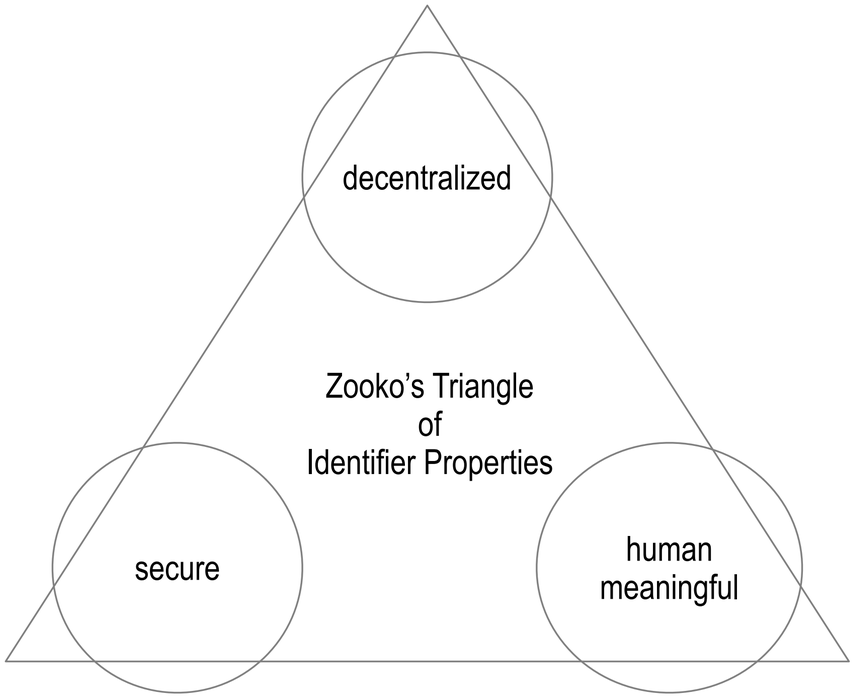


Zooko's Trilemma

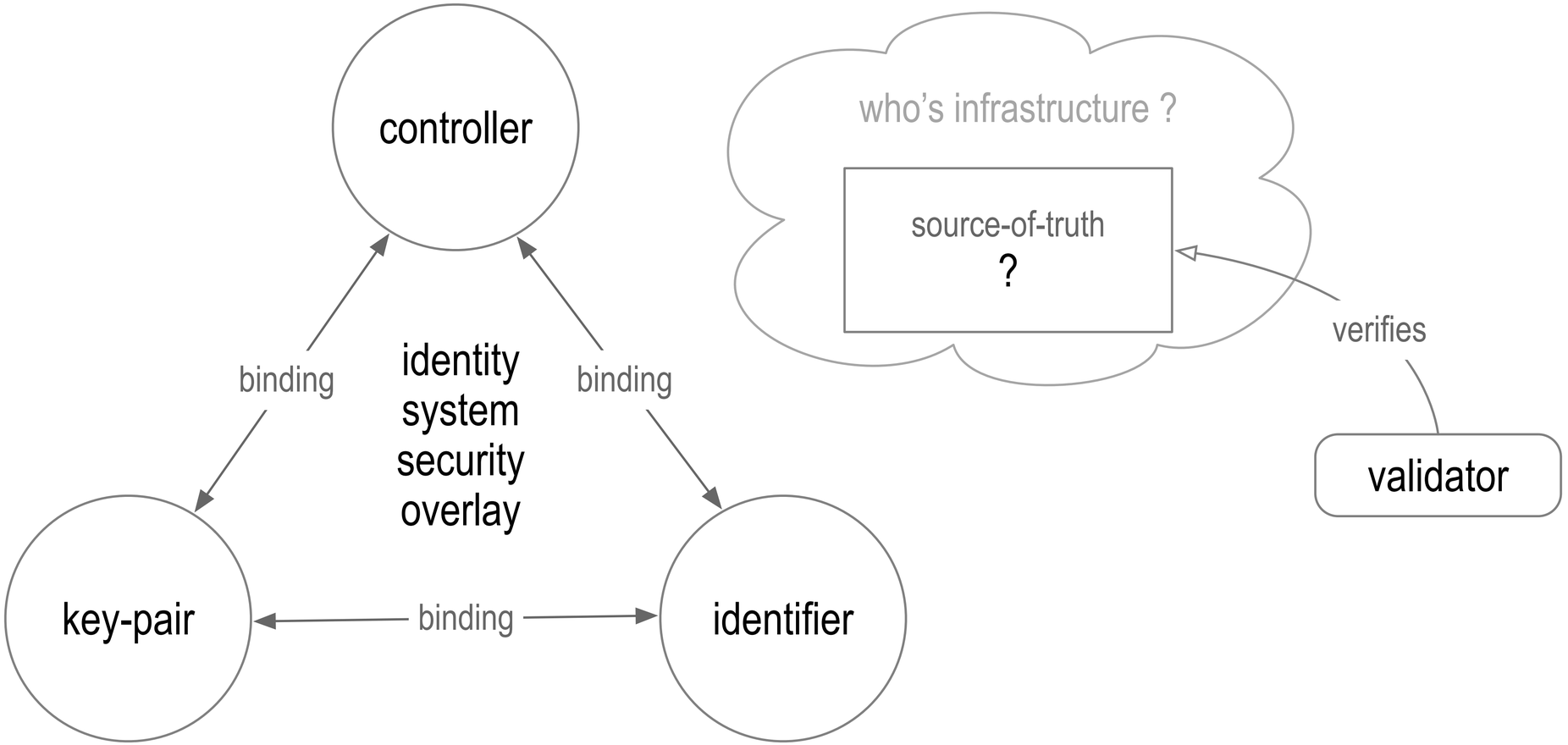
Desirable identifier properties: secure, decentralized, human meaningful

Trilemma: May have any two of the three properties but not all three.

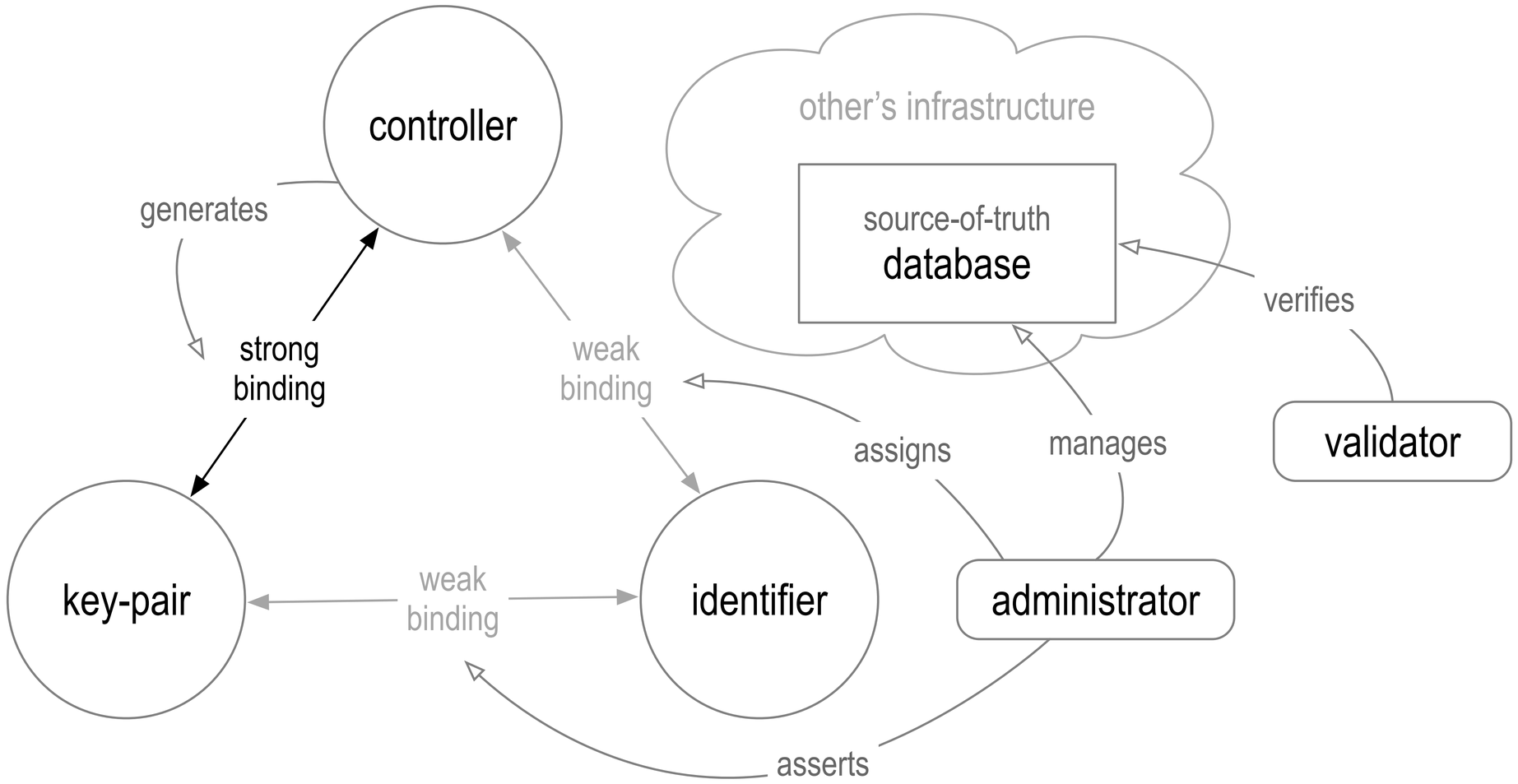
One way to sort of solve the trilemma is to uniquely register a human meaningful identifier on a ledger controlled by a different identifier that is secure and decentralized but not human meaningful.



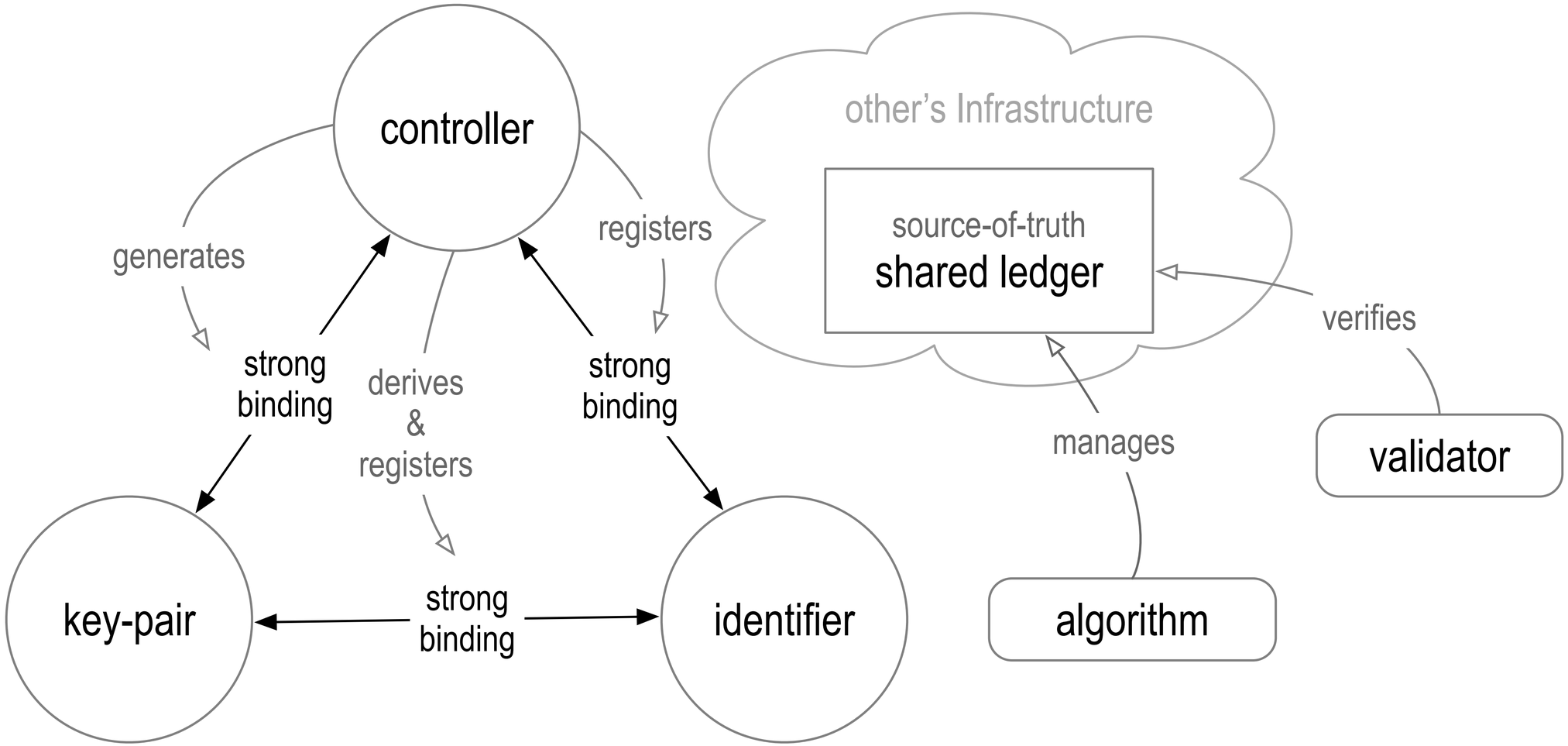
Trust Basis


controller
who's infrastructure ?
source-of-truth
?
verifies
validator
binding
identity
system
security
overlay
binding
key-pair
binding
identifier

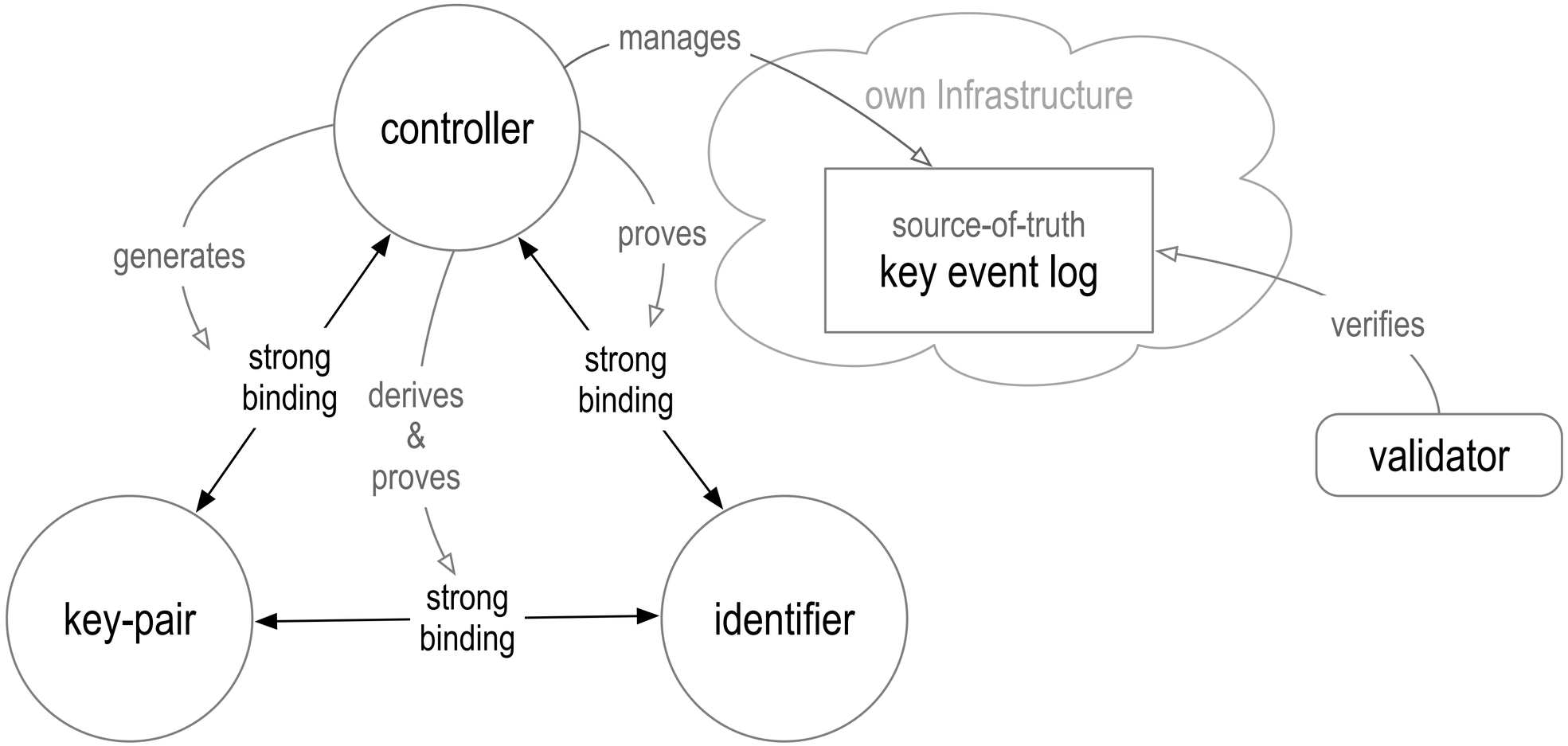
Administrative Trust Basis



Algorithmic Trust Basis



Autonomic Trust Basis



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 + KA^{2}CE + recoverable + post-quantum.

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.*

AIDs and ANs are *portable* = truly self-sovereign.

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

Unified Identifier Model

AID: Autonomic Identifier (primary)

self-managing self-certifying identifier with cryptographic root of trust

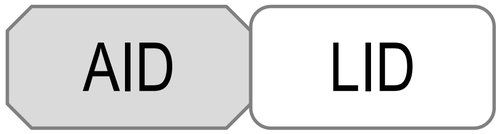
secure, decentralized, portable, universally unique

L*ID*: Legitimized Human Meaningful Identifier (secondary)

legitimized within trust domain of given AID by a verifiable authorization from AID controller

authorization is verifiable to the root-of-trust of AID

Forms *AID*|*LID* couplet within trust domain of AID



AID|LID Couplet

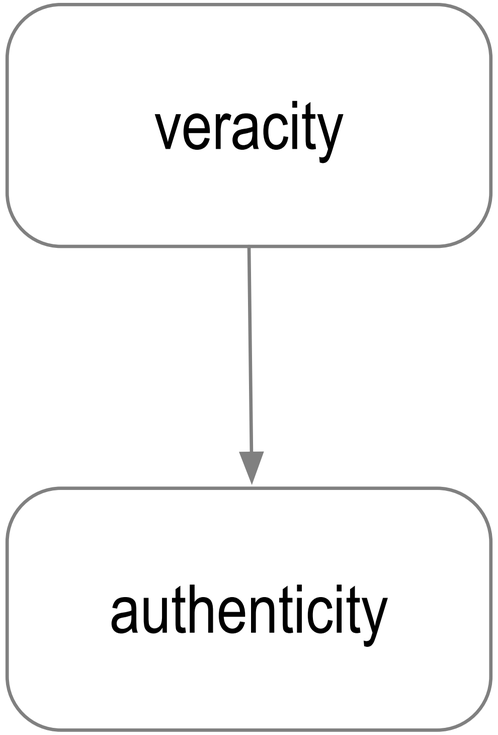
625.127C125r

EXq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148|625.127C125r

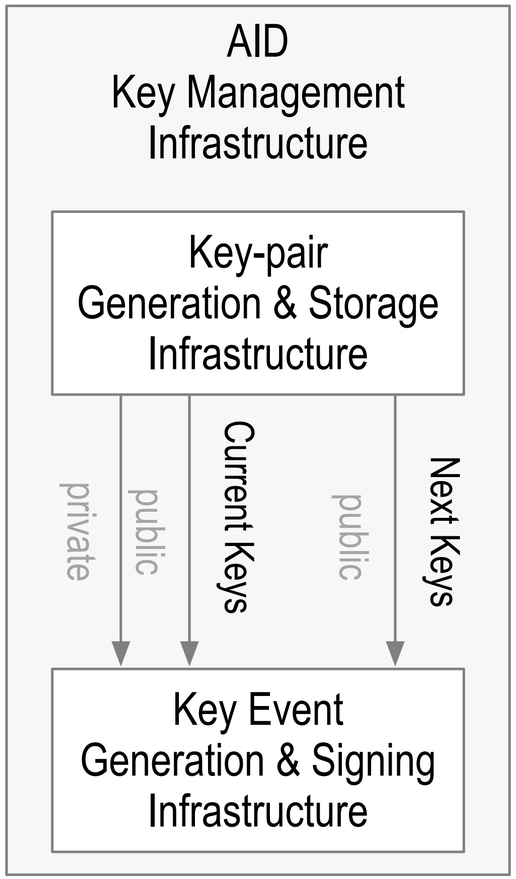
Trust Balance

Reputational Trust

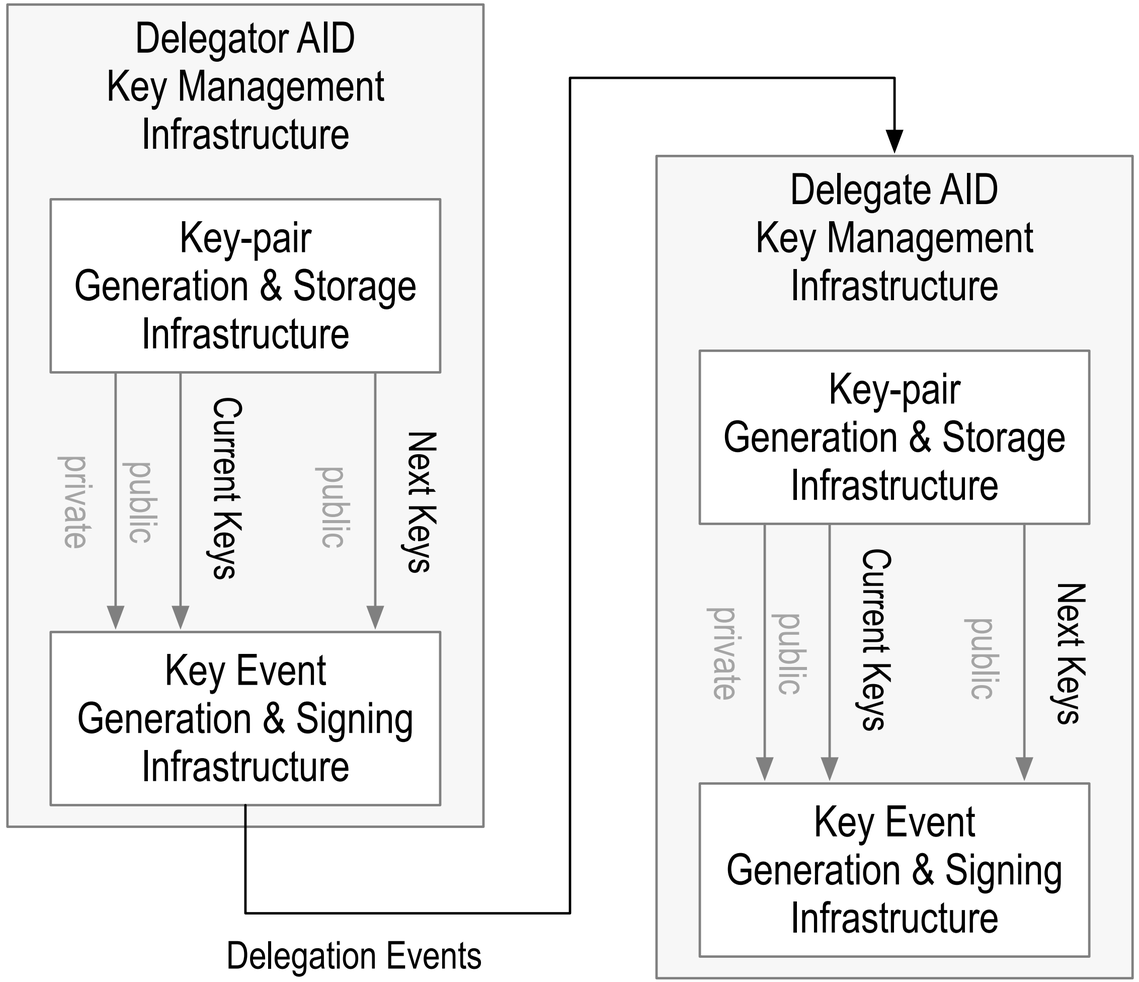
Cryptographic Trust



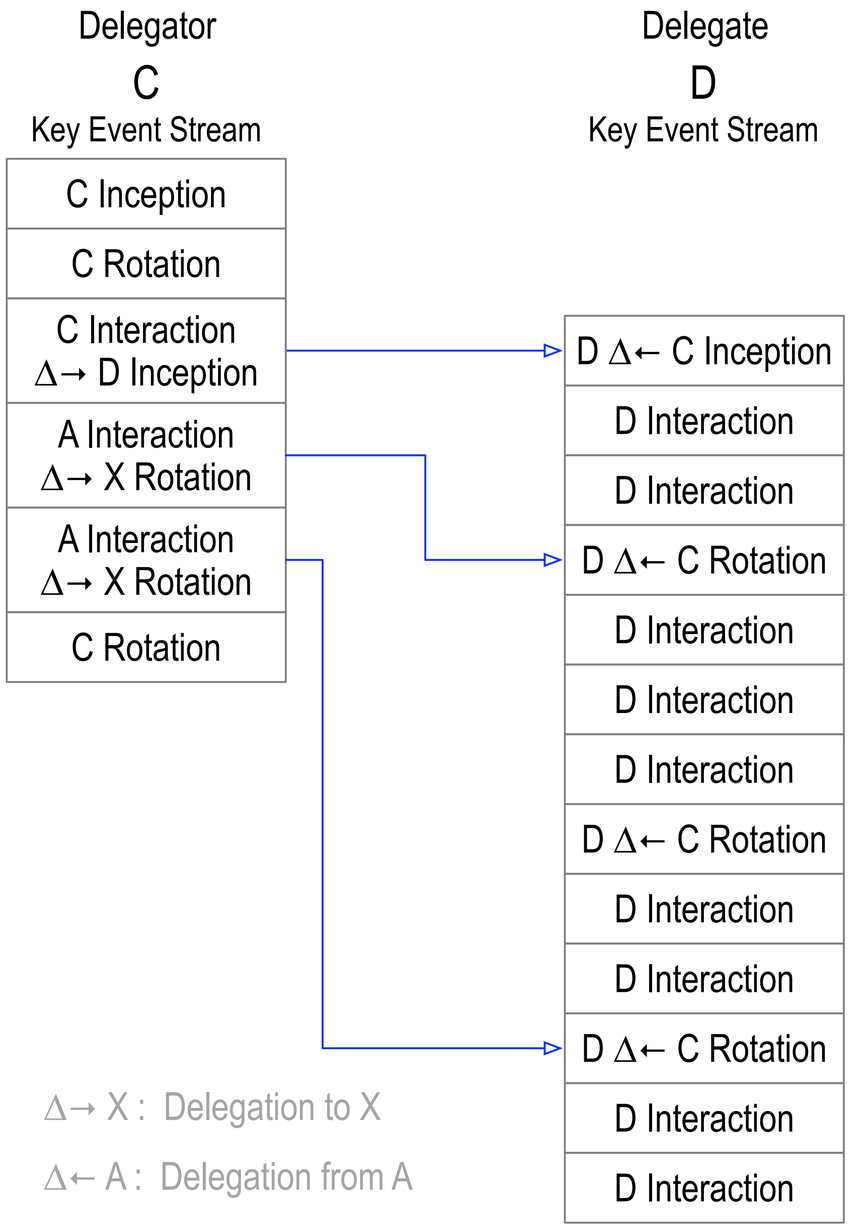
Decentralized Key Management Infrastructure (Univalent DKMI)



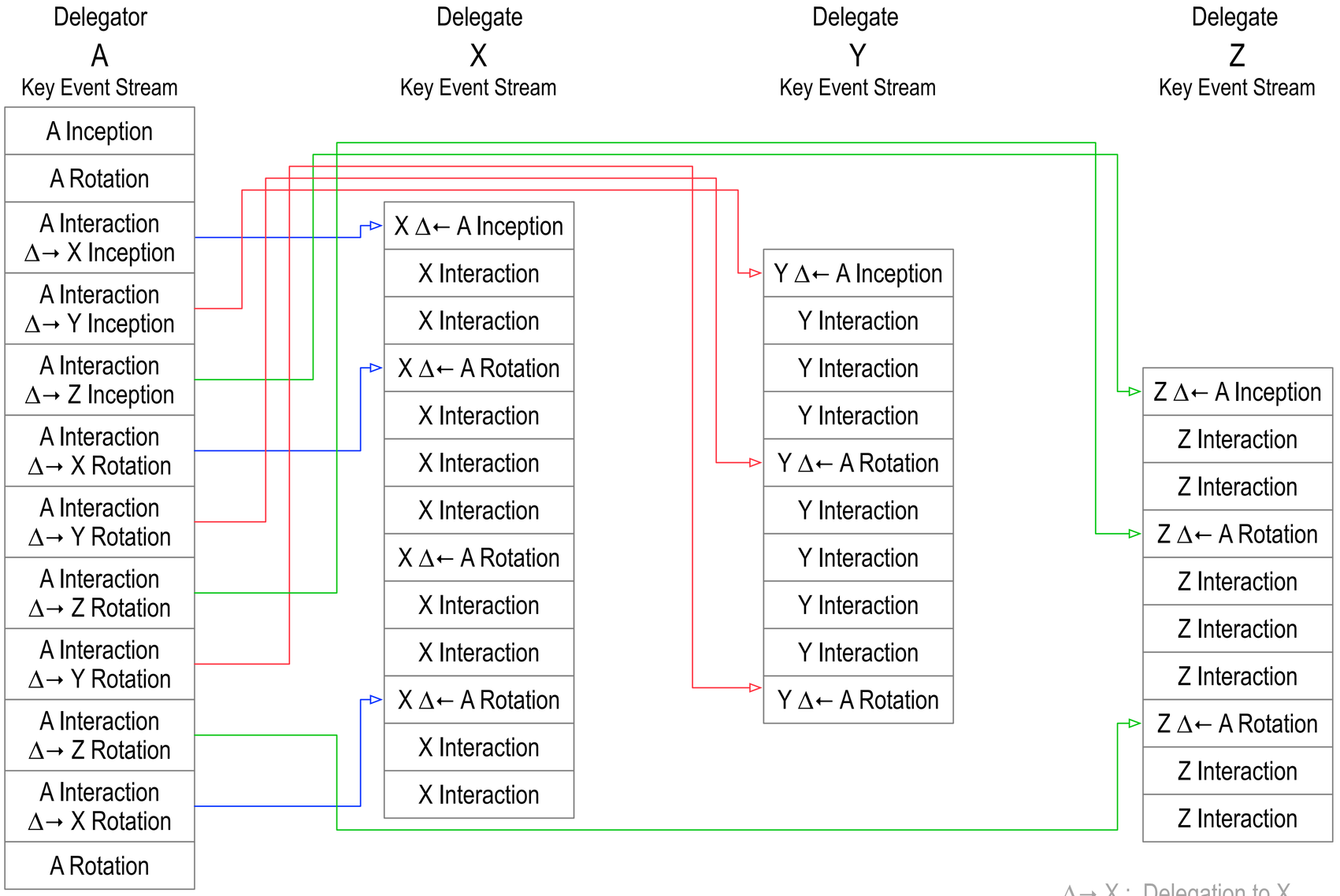
Hierarchical DKMI: Bivalent DKMI



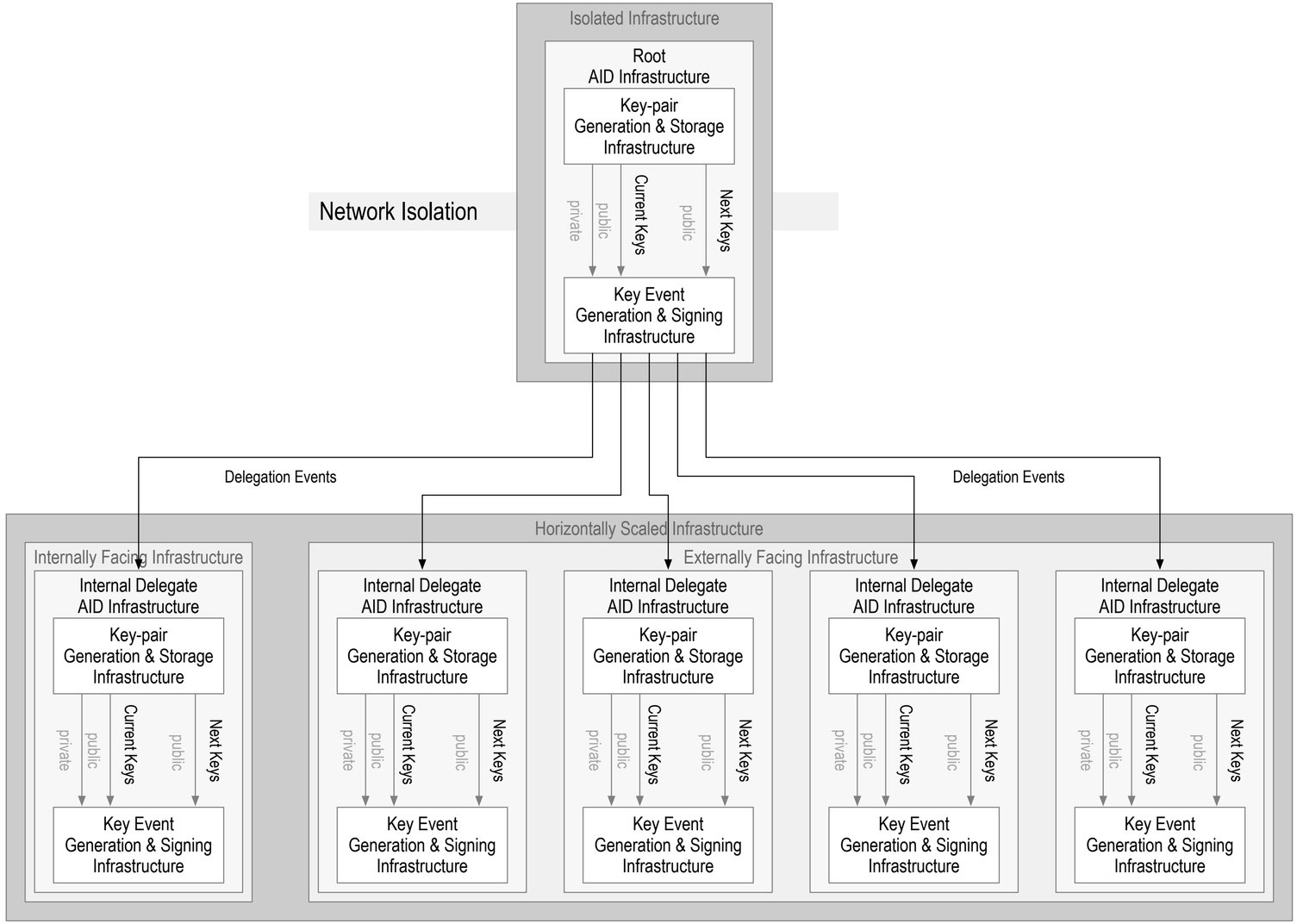
Interaction Delegation



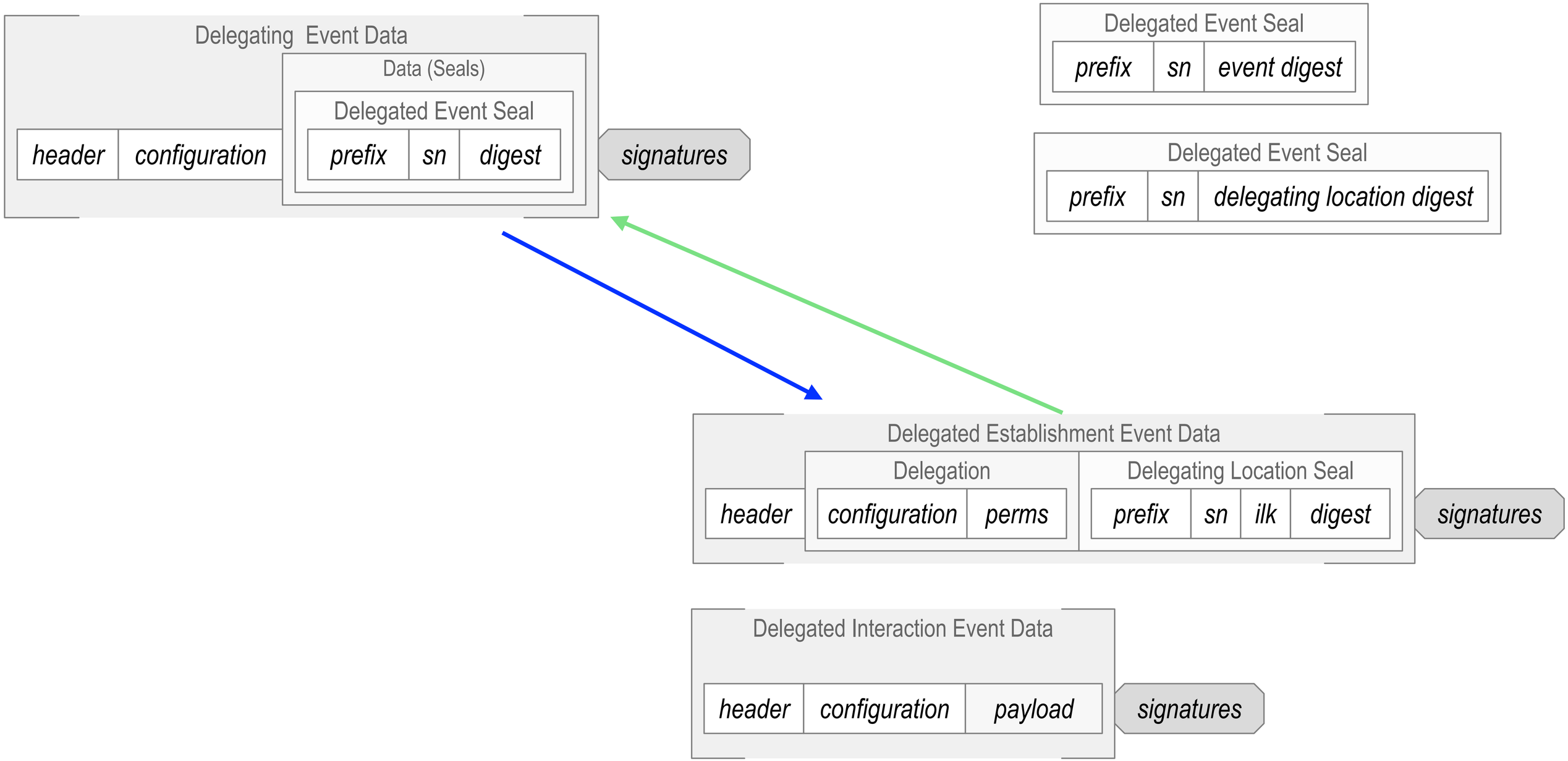
Scaling Delegation via Interaction



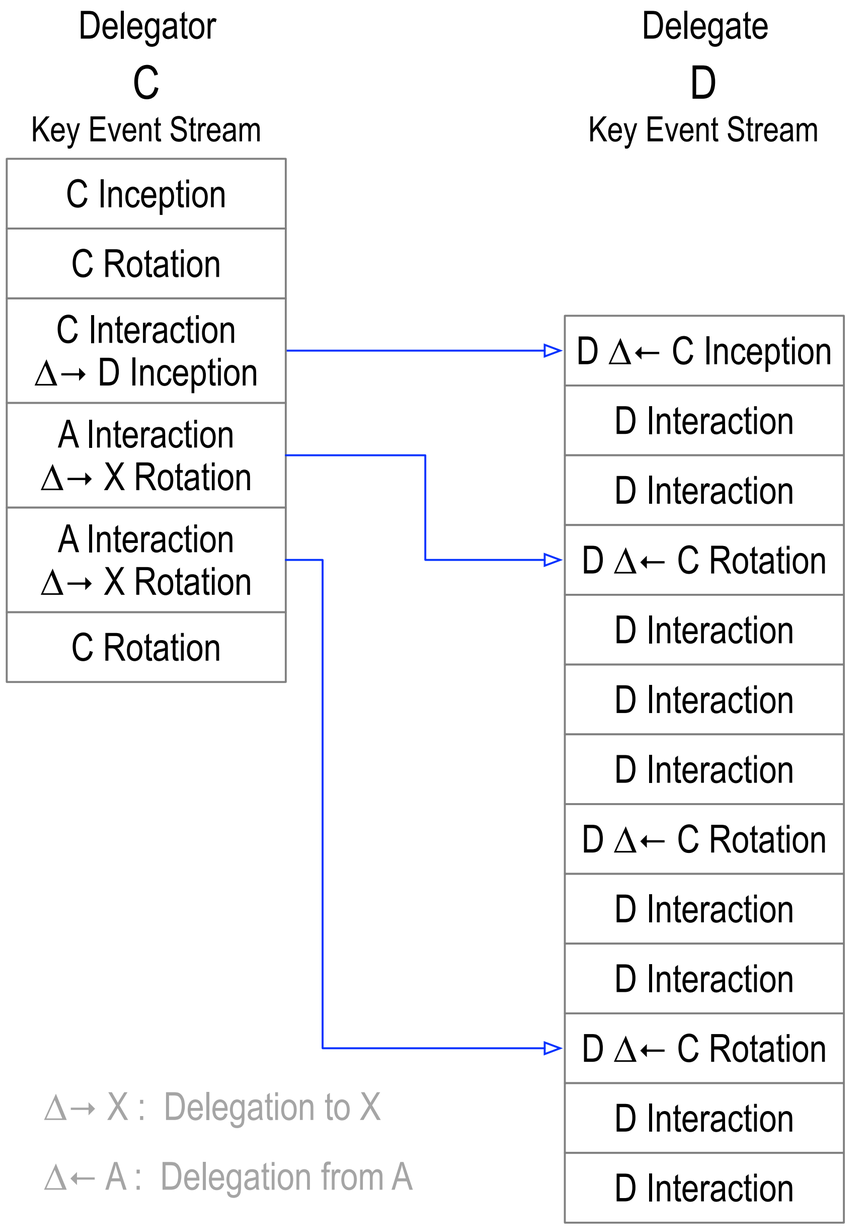
MultiValent Delegation



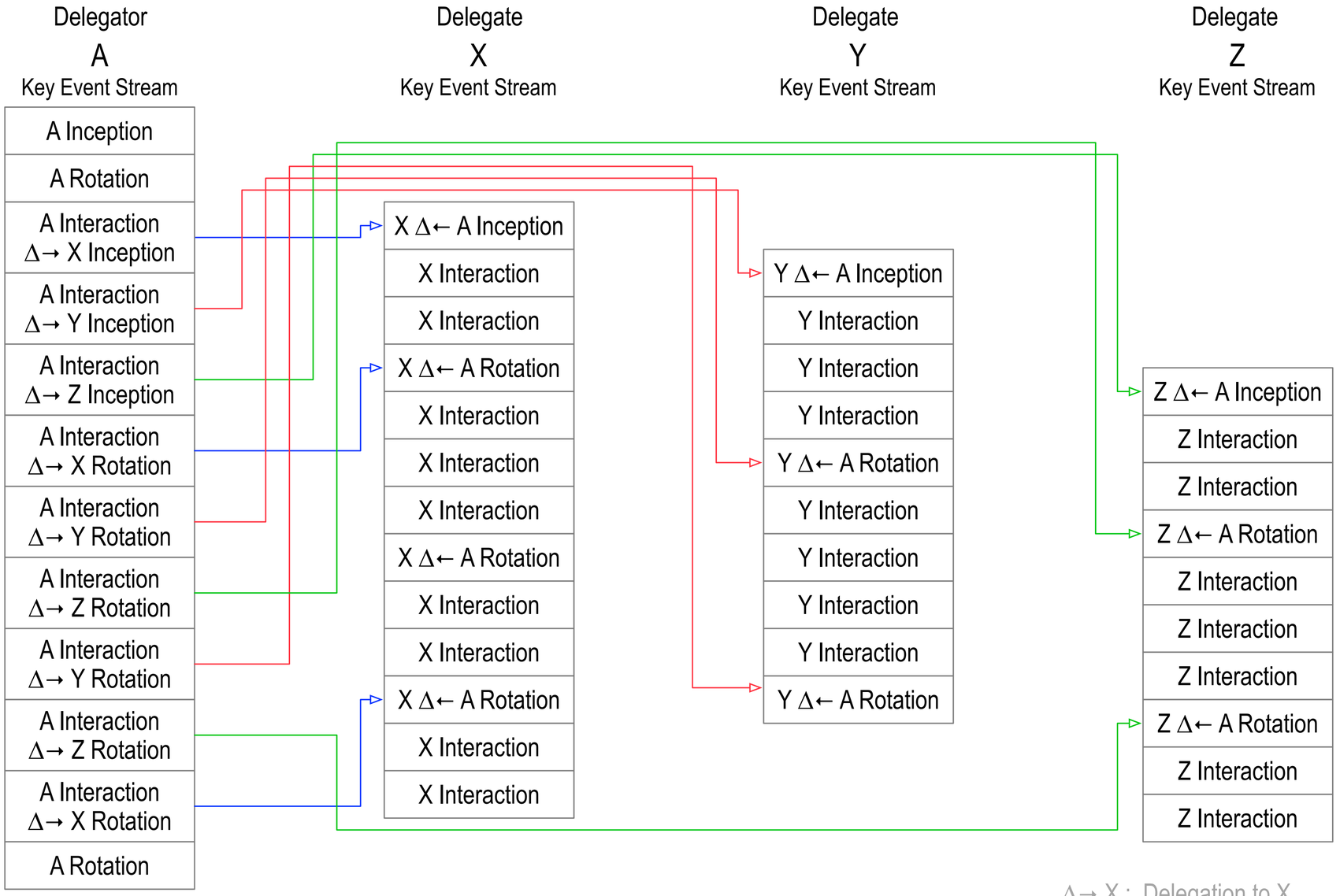
Delegation (Cross Anchor)



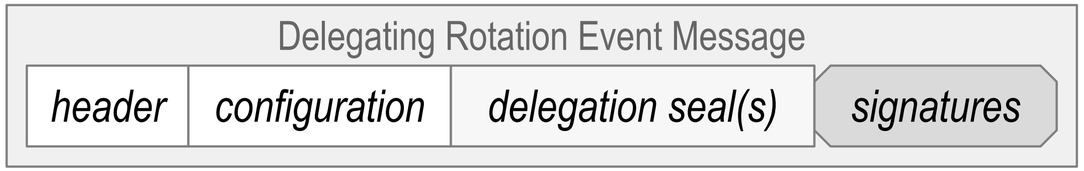
Interaction Delegation

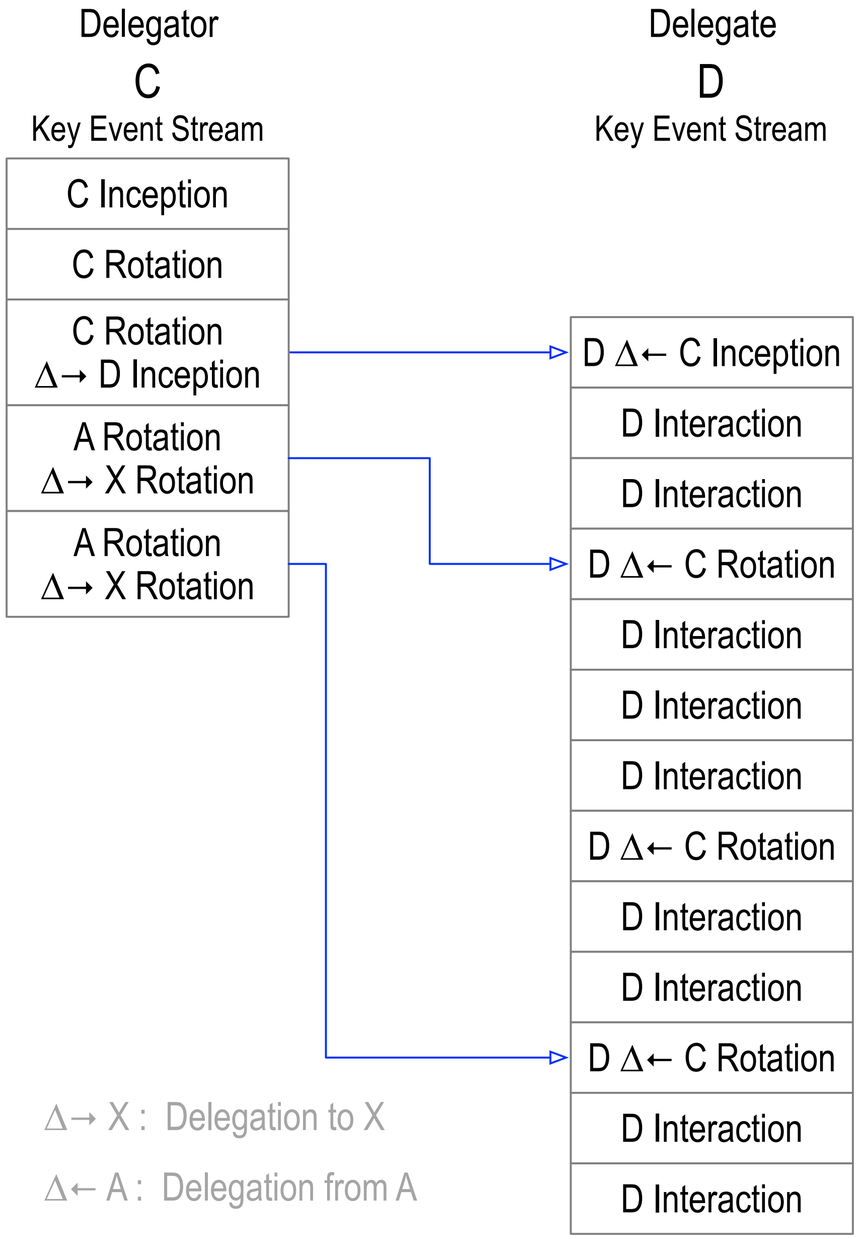


Scaling Delegation via Interaction

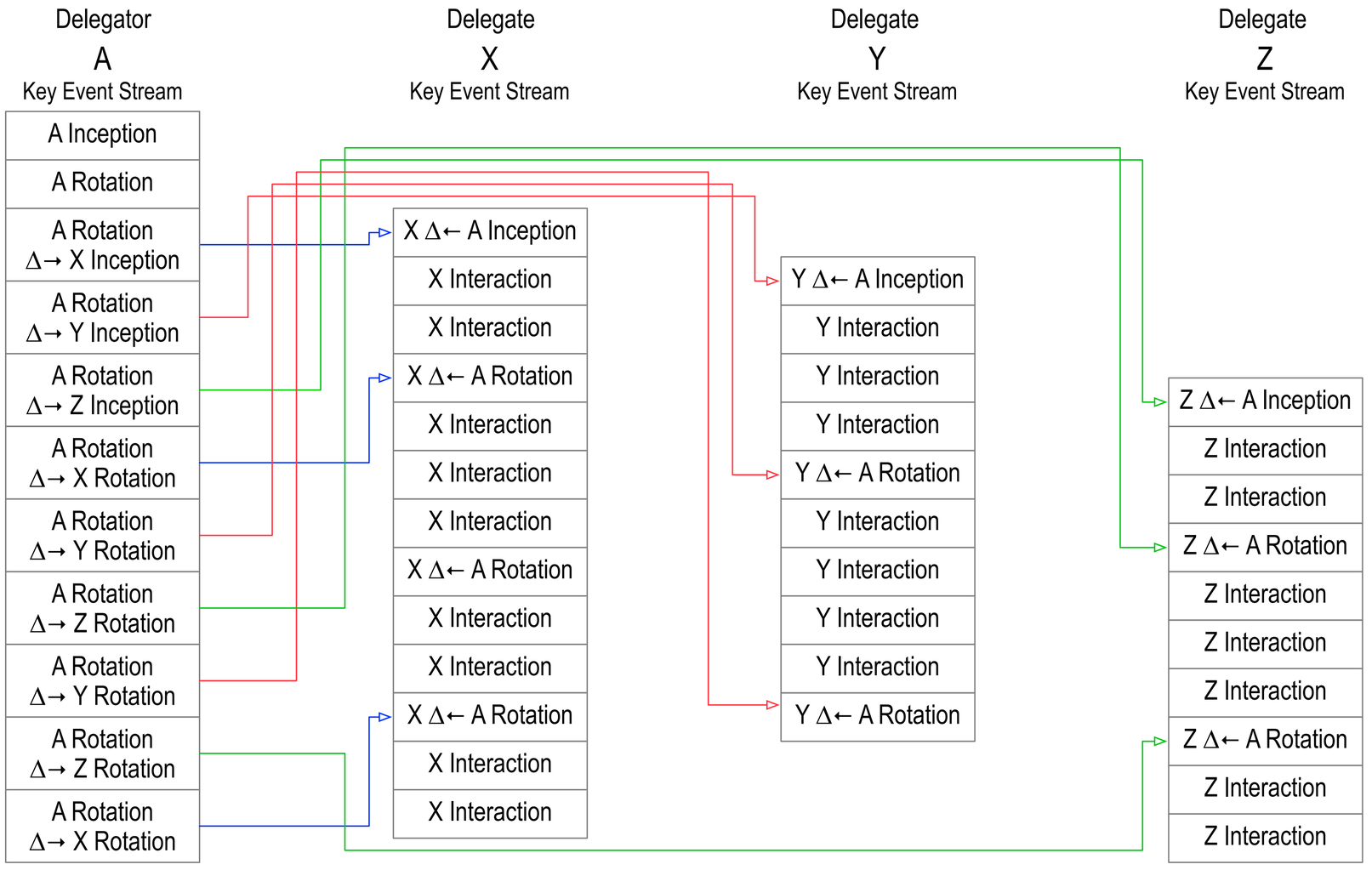


Rotation Delegation





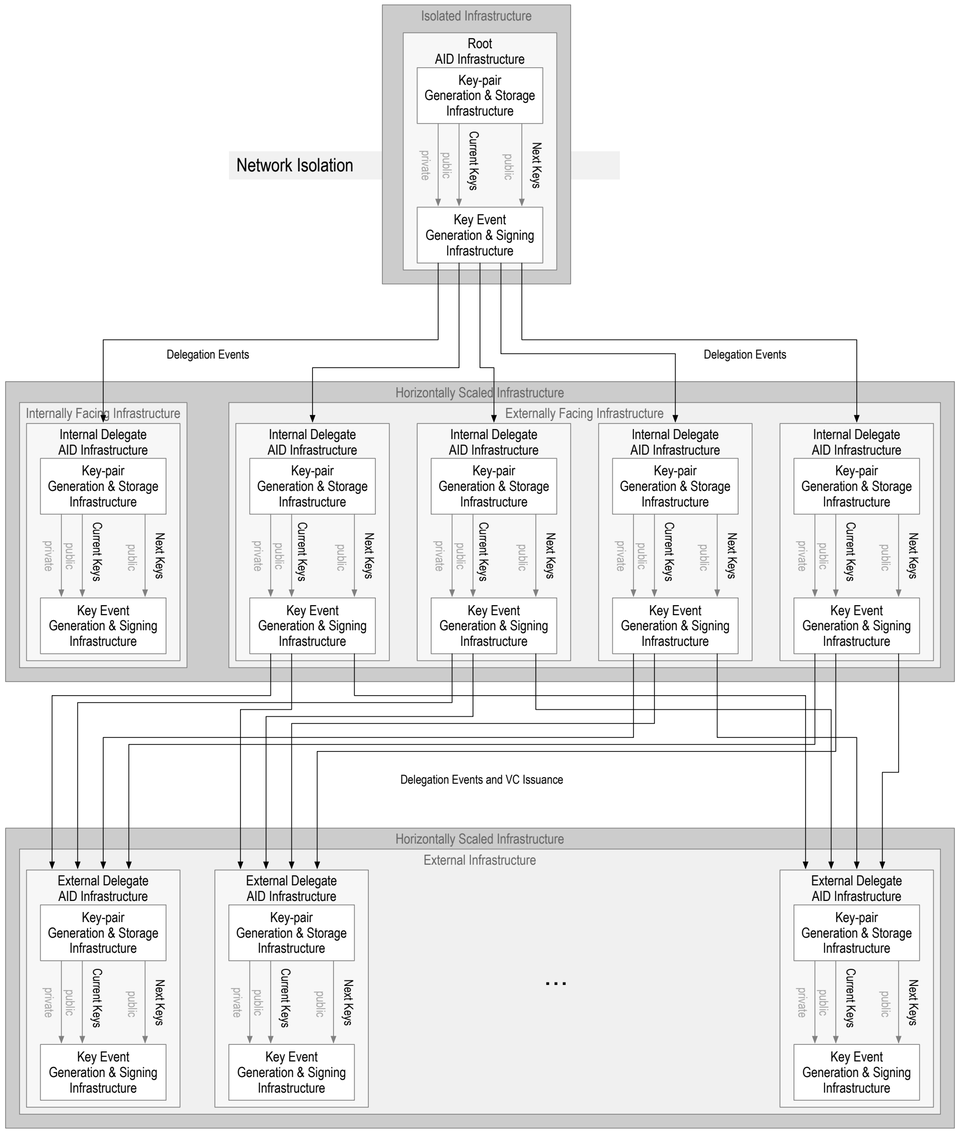
Scaling Delegation via Rotation



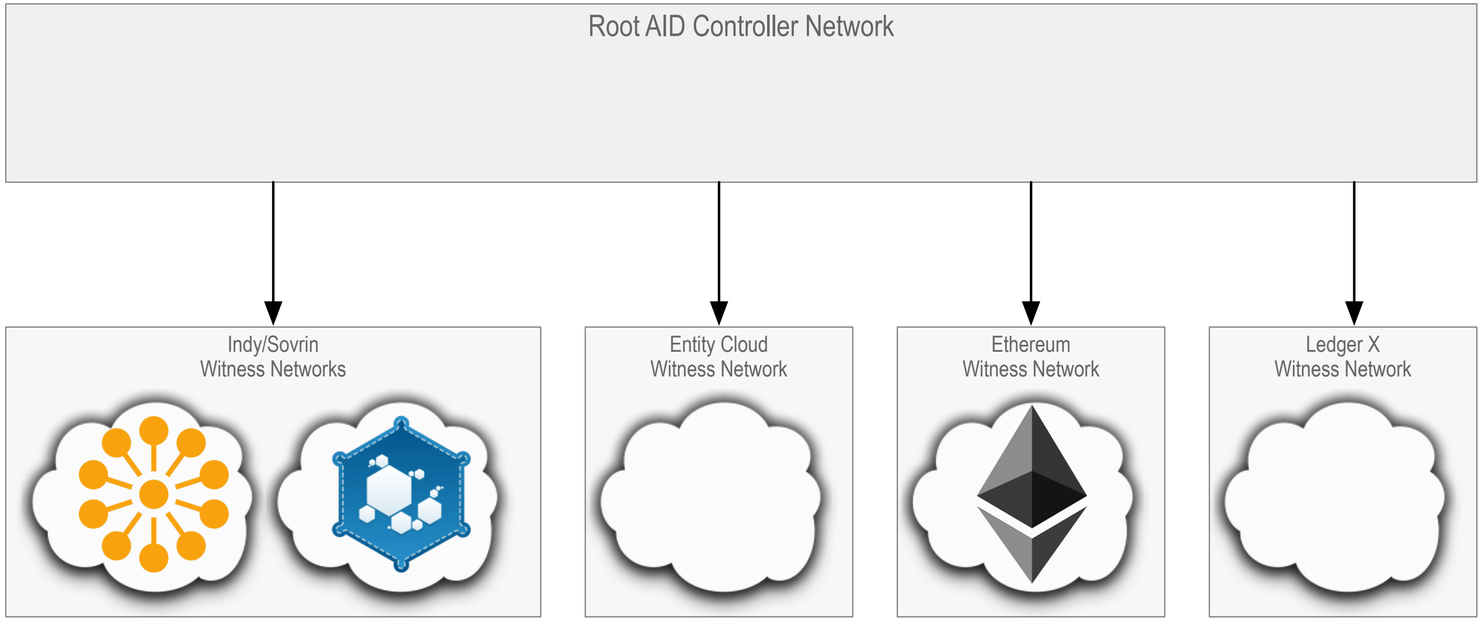
Δ→ X : Delegation to X

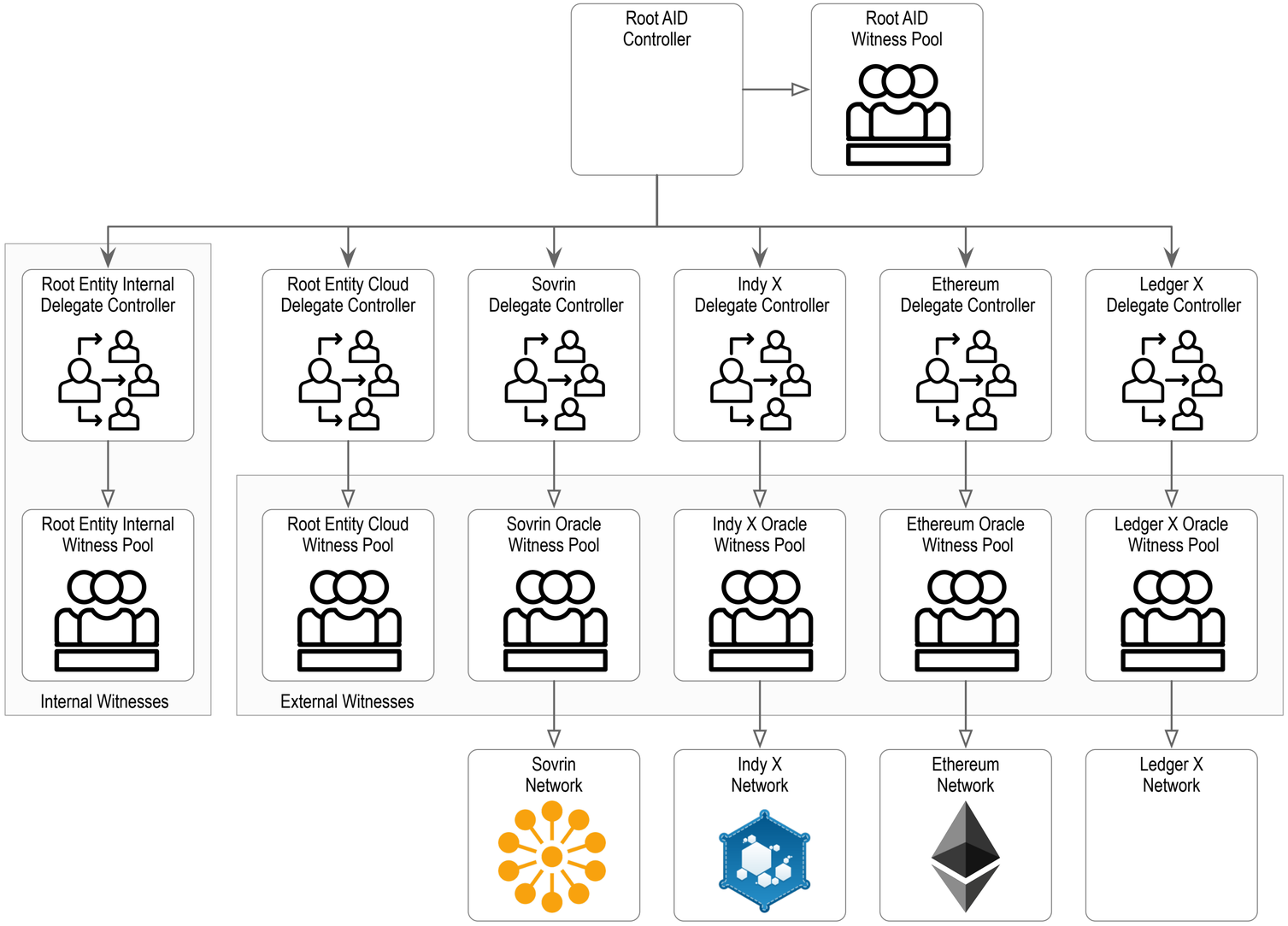
Δ← A : Delegation from A

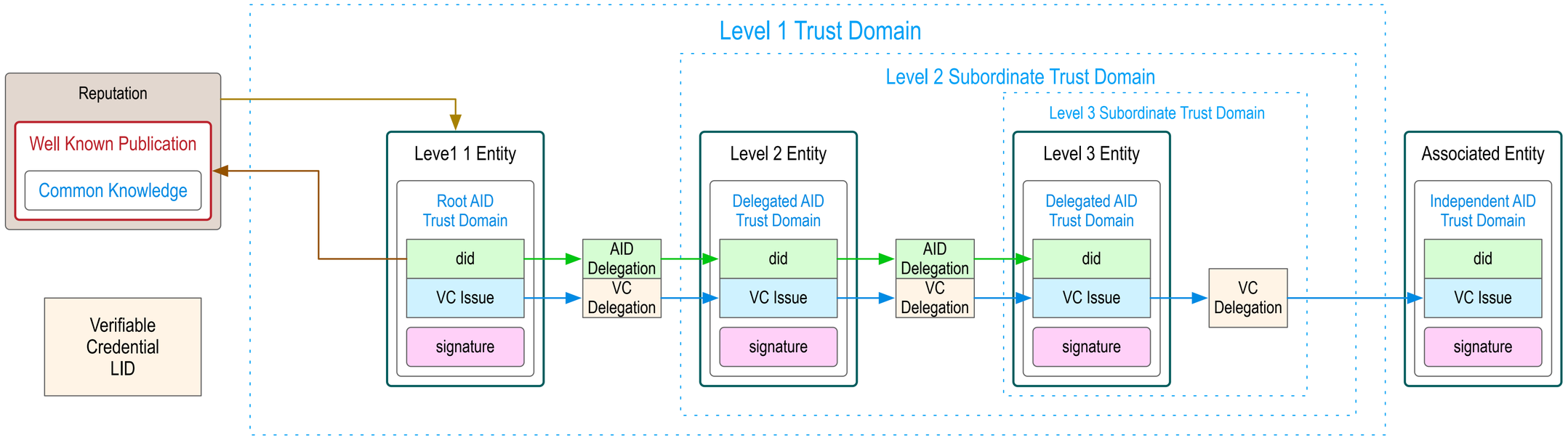
Multi-layer Multi-Valent Delegation

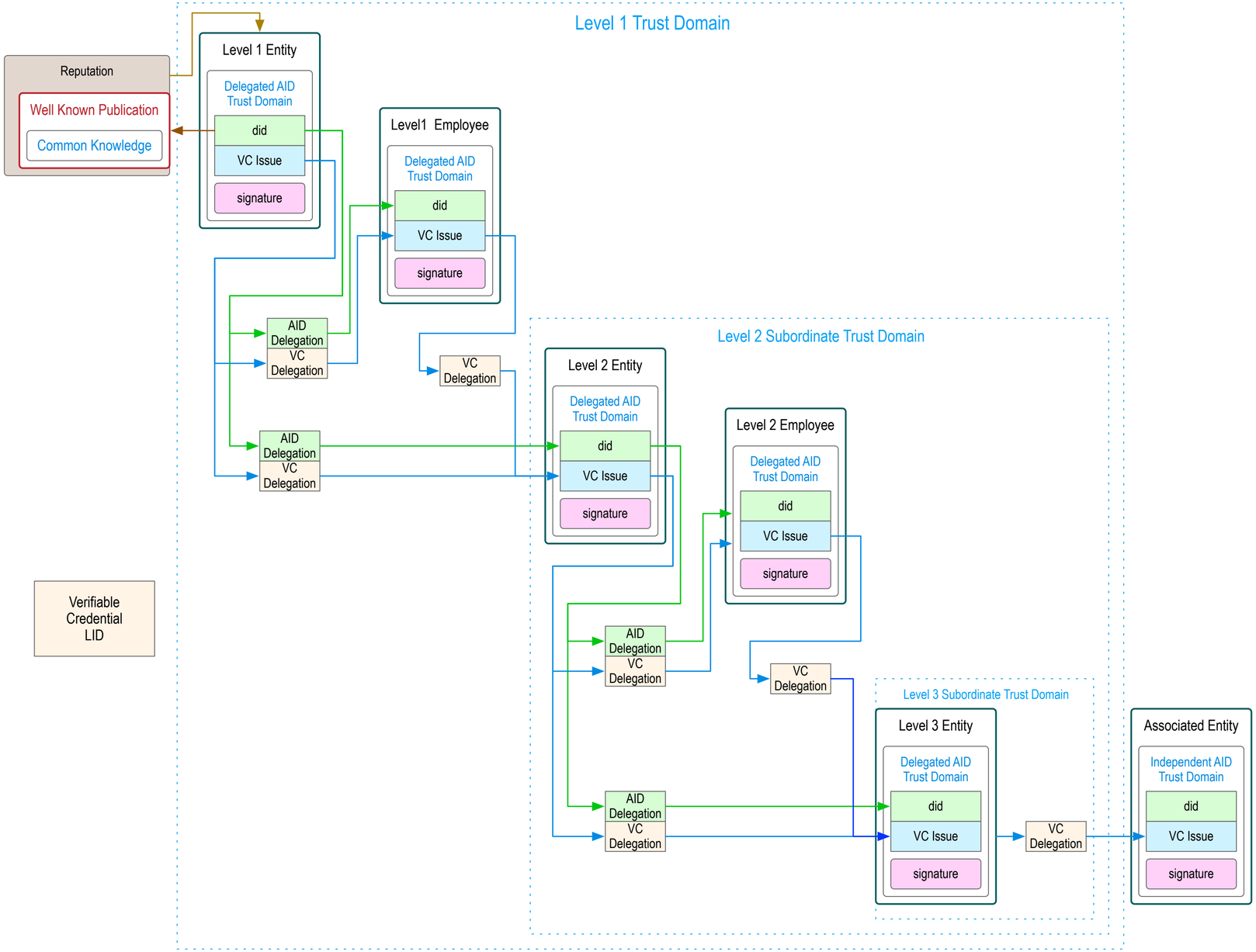


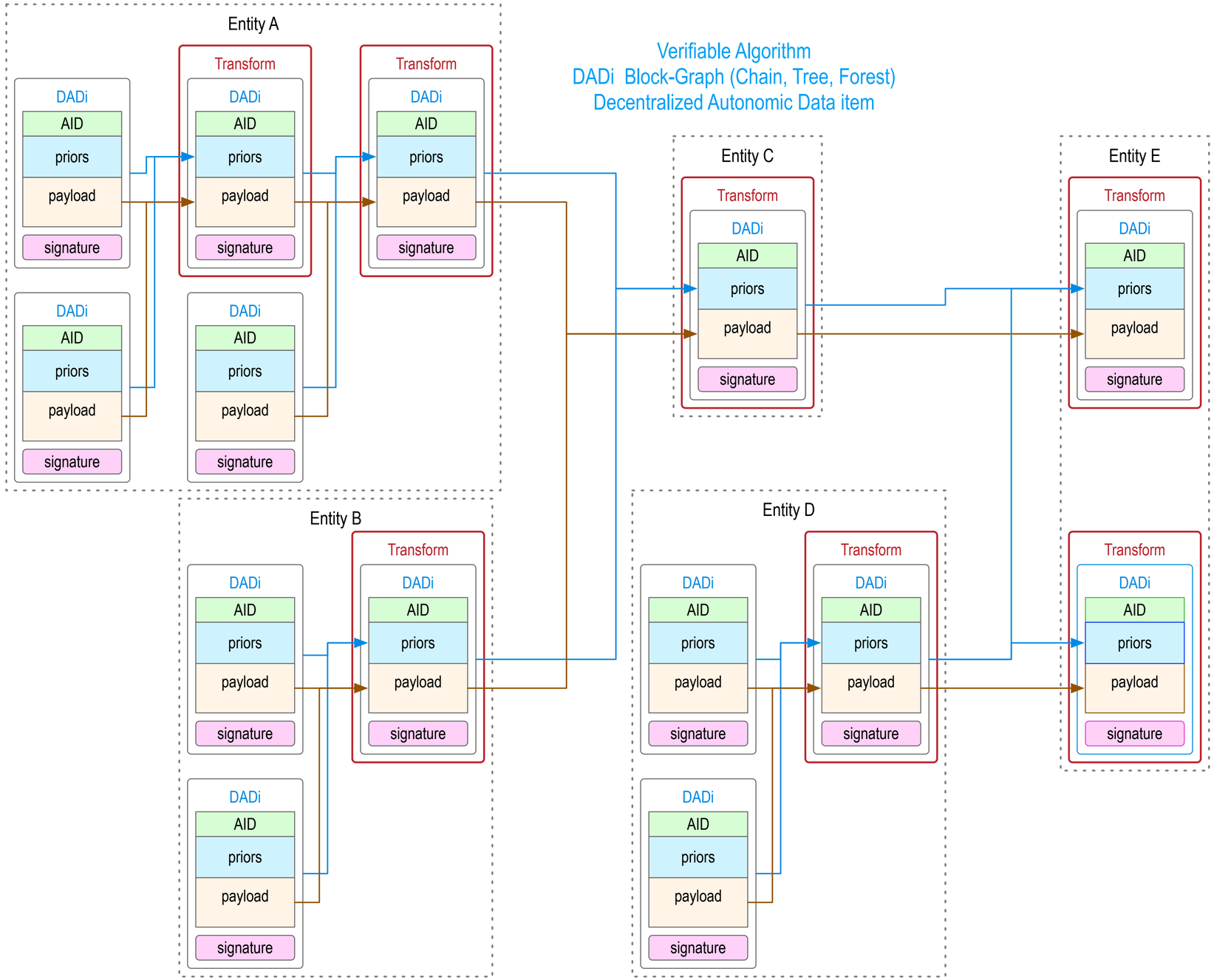
Witness
Network-of-Networks



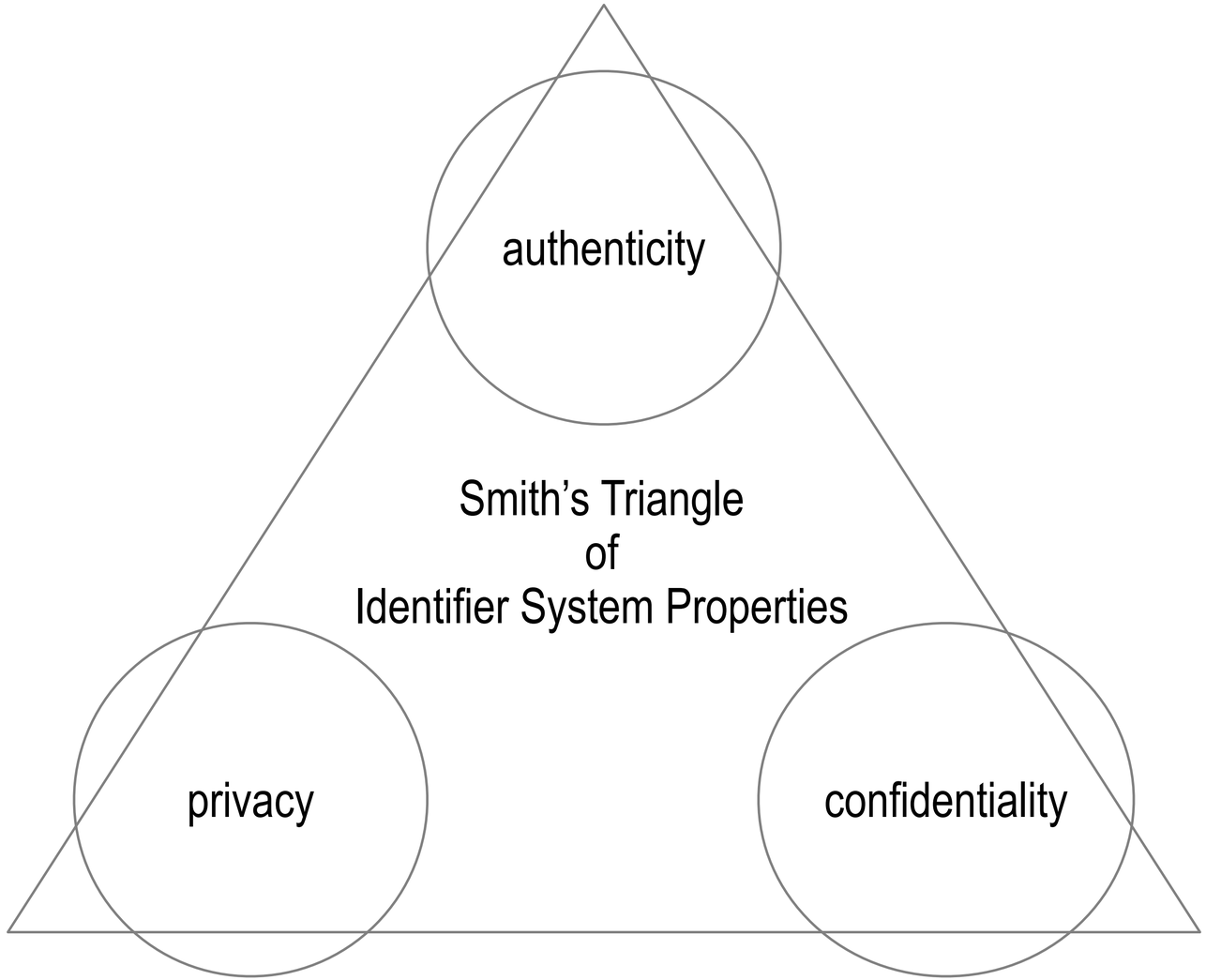

Root AID Controller
Root AID Witness Pool
Root Entity Internal Delegate Controller
Root Entity Cloud Delegate Controller
Sovrin Delegate Controller
Indy X Delegate Controller
Ethereum Delegate Controller
Ledger X Delegate Controller
Root Entity Internal Witness Pool
Root Entity Cloud Witness Pool
Sovrin Oracle Witness Pool
Indy X Oracle Witness Pool
Ethereum Oracle Witness Pool
Ledger X Oracle Witness Pool
Internal Witnesses
External Witnesses
Sovrin Network
Indy X Network
Ethereum Network
Ledger X Network


Level 1 Trust Domain
Level 2 Subordinate Trust Domain
Level 3 Subordinate Trust Domain
Reputation
Well Known Publication
Common Knowledge
Verifiable Credential LID
Leve1 1 Entity
Root AID Trust Domain
did
VC Issue
signature
AID Delegation
VC Delegation
Level 2 Entity
Delegated AID Trust Domain
did
VC Issue
signature
AID Delegation
VC Delegation
Level 3 Entity
Delegated AID Trust Domain
did
VC Issue
signature
VC Delegation
Associated Entity
Independent AID Trust Domain
did
VC Issue
signature


Level 1 Trust Domain
Reputation
Well Known Publication
Common Knowledge
Level 1 Entity
Delegated AID Trust Domain
did
VC Issue
signature
Level1 Employee
Delegated AID Trust Domain
did
VC Issue
signature
AID Delegation
VC Delegation
VC Delegation
AID Delegation
VC Delegation
Level 2 Subordinate Trust Domain
Level 2 Entity
Delegated AID Trust Domain
did
VC Issue
signature
Level 2 Employee
Delegated AID Trust Domain
did
VC Issue
signature
AID Delegation
VC Delegation
VC Delegation
Level 3 Subordinate Trust Domain
AID Delegation
VC Delegation
Level 3 Entity
Delegated AID Trust Domain
did
VC Issue
signature
VC Delegation
Associated Entity
Independent AID Trust Domain
did
VC Issue
signature
Verifiable Credential LID

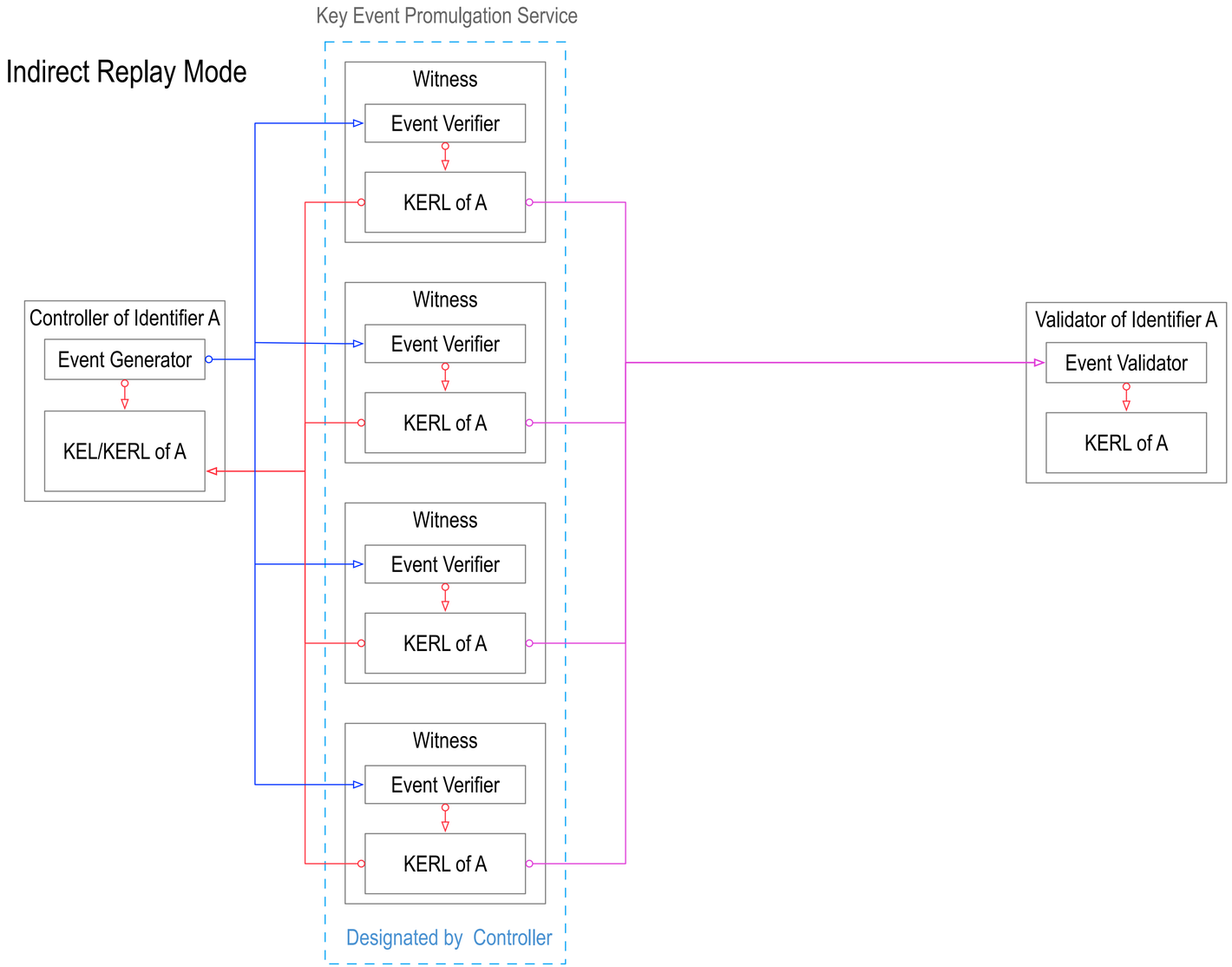

Verifiable Algorithm
DADi Block-Graph (Chain, Tree, Forest)
Decentralized Autonomic Data item
Entity A
Entity B
Entity C
Entity D
Entity E
Transform
DADi
AID
priors
payload
signature

Smith’s Identifier System Properties Triangle

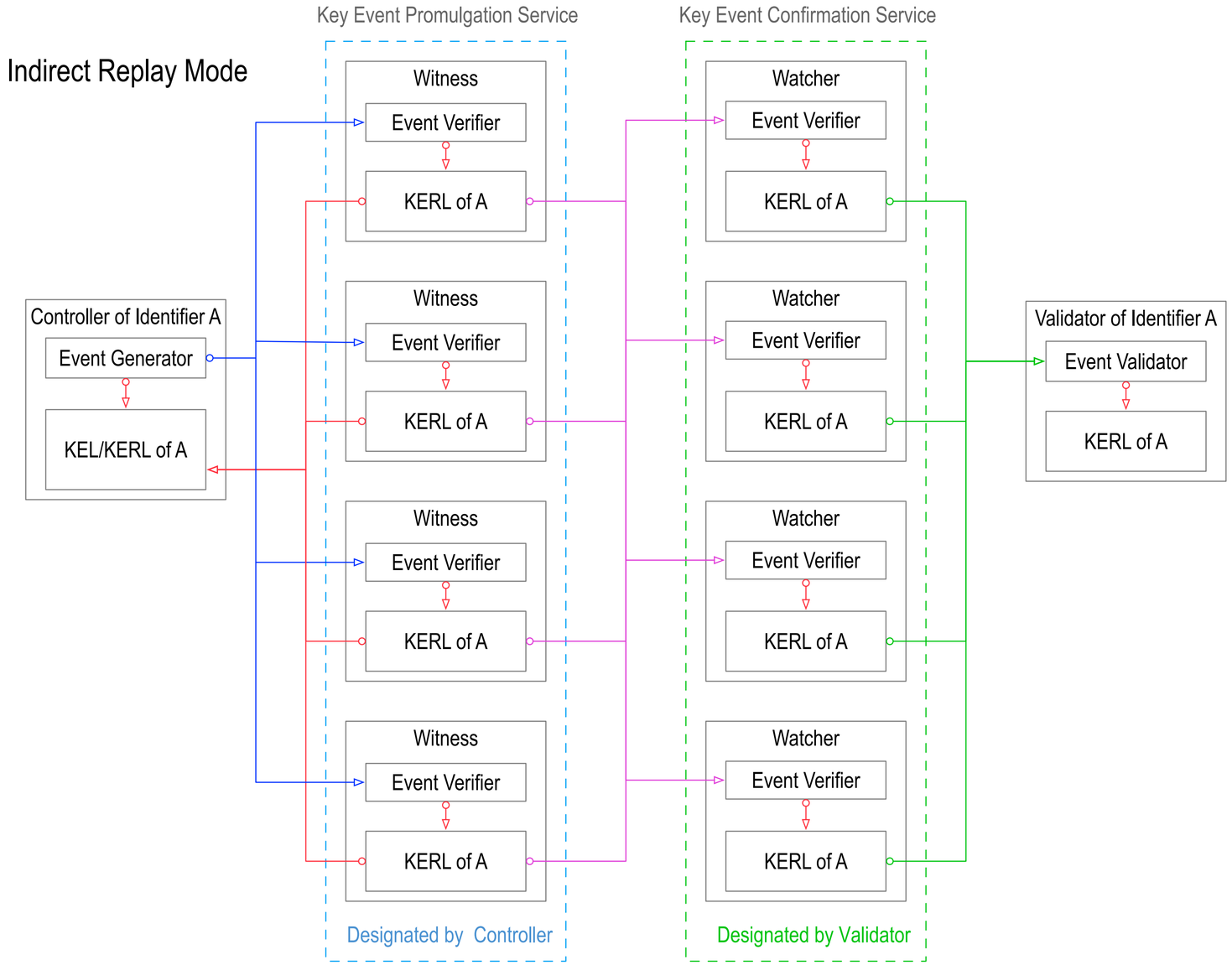


May exhibit any two at the highest level but not all three at the highest level

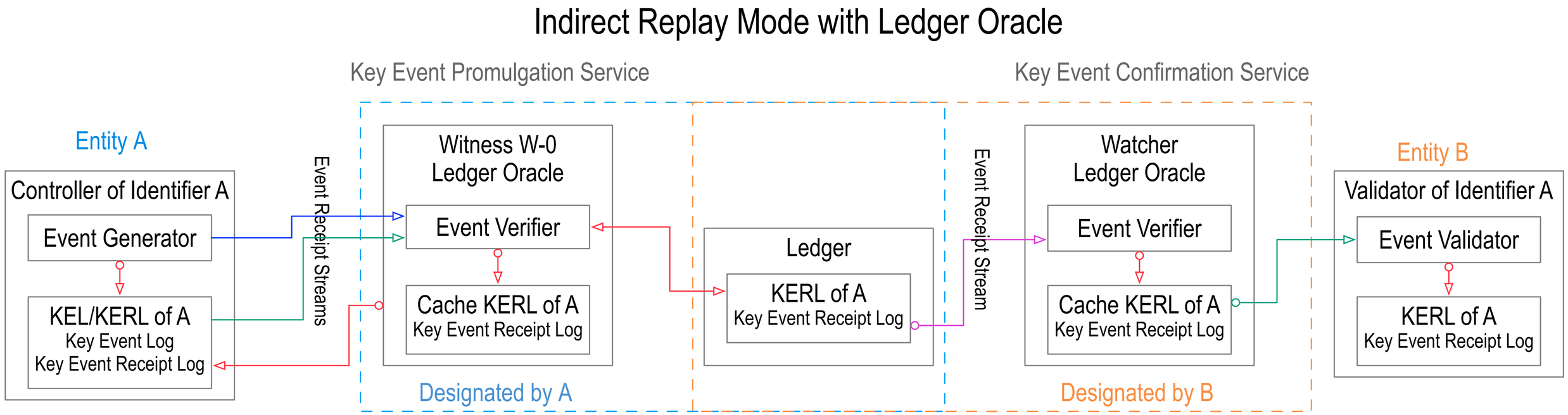
Indirect Mode Promulgation Service



Indirect Mode
Promulgation and Confirmation Services



Indirect Mode with Ledger Oracles



Separation of Control

Shared ledger = *shared control* over *shared data*.

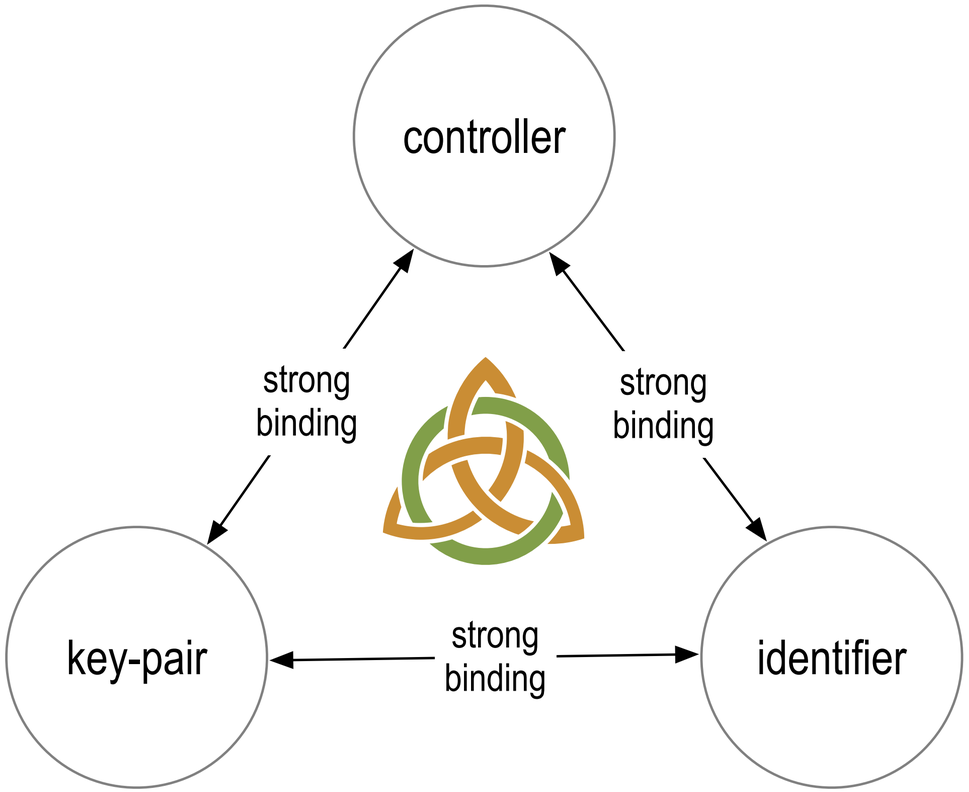
Shared *data* = *good*, shared *control* = *bad*.

Shared control between controllers and validators may be problematic for governance, scalability, and performance.

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

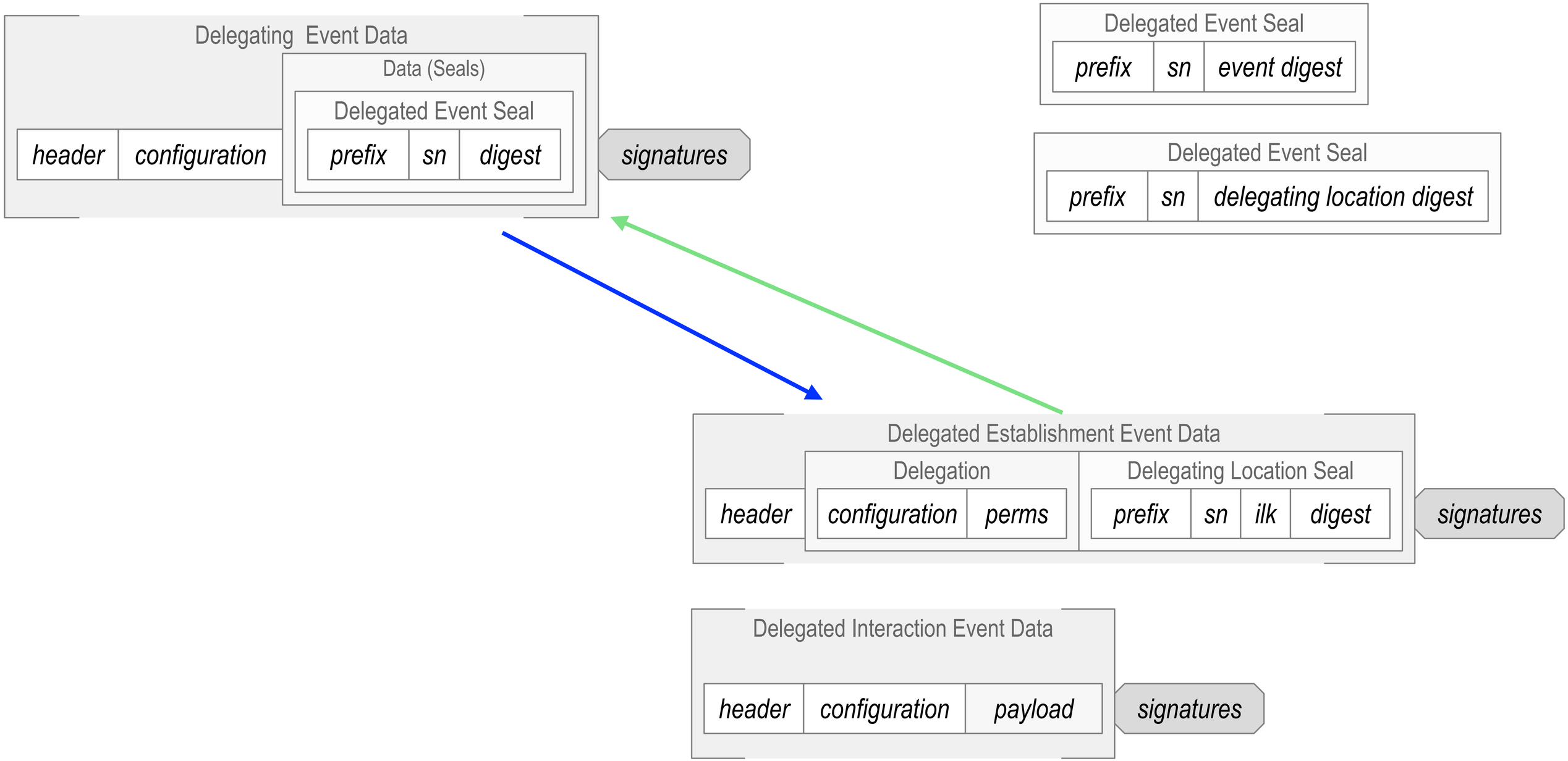
Separated control between controllers and validators may provide better decentralization, more flexibility, better scalability, lower cost, higher performance, and more privacy at comparable security.

BACKGROUND

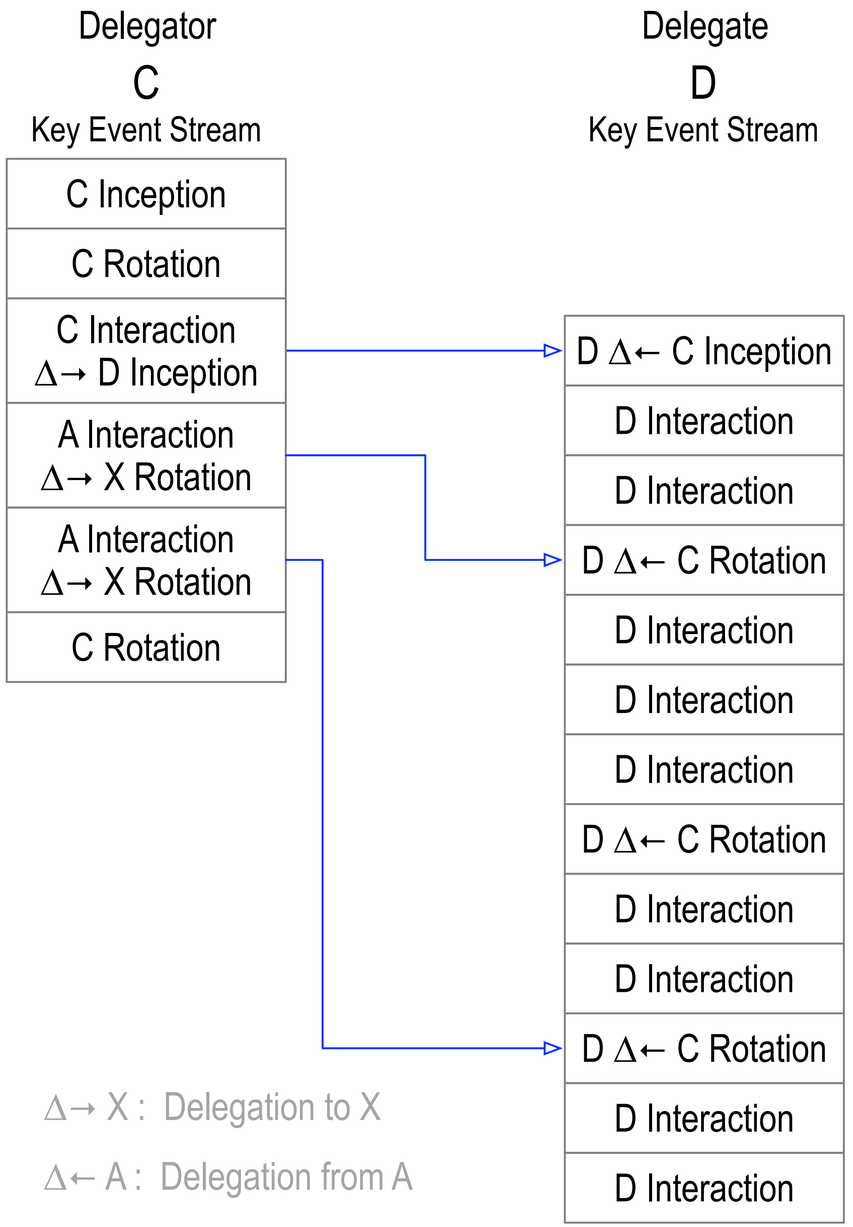


KERI

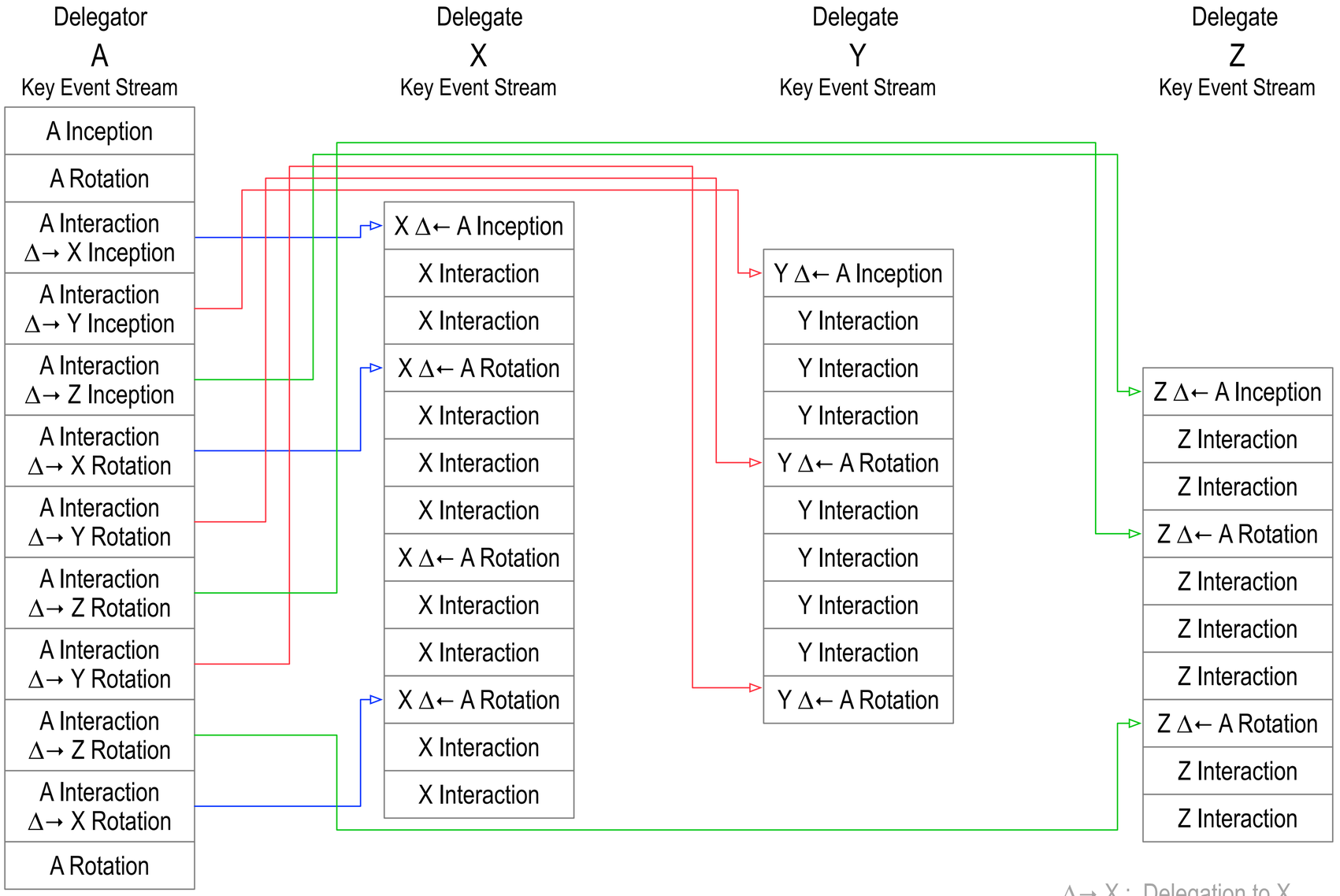
Delegation (Cross Anchor)



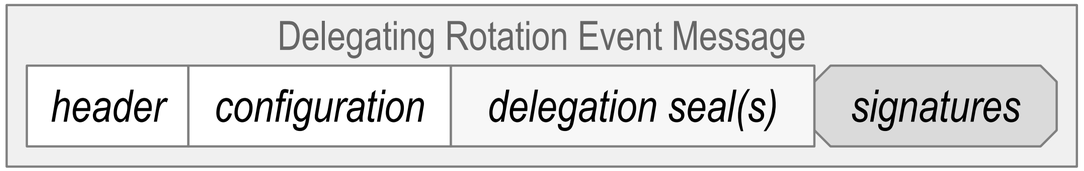
Interaction Delegation

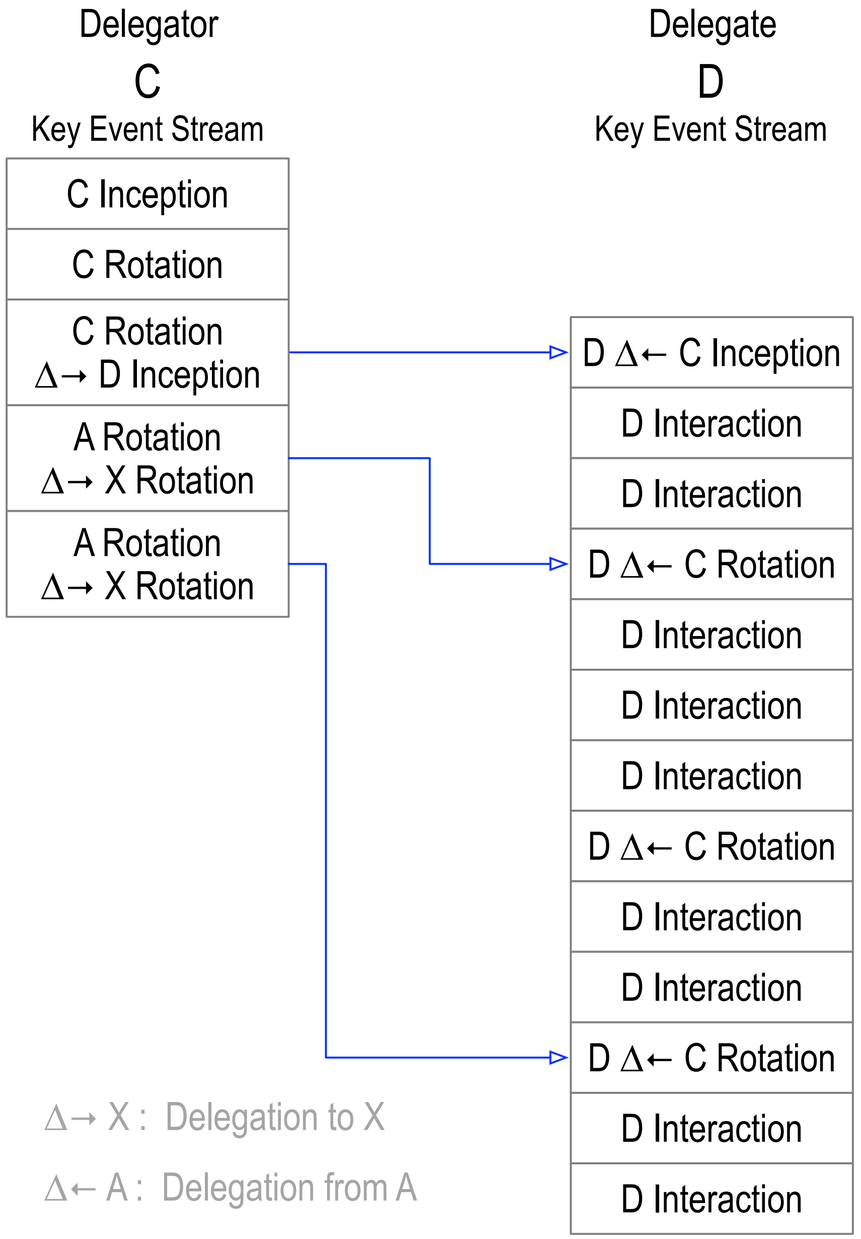


Scaling Delegation via Interaction

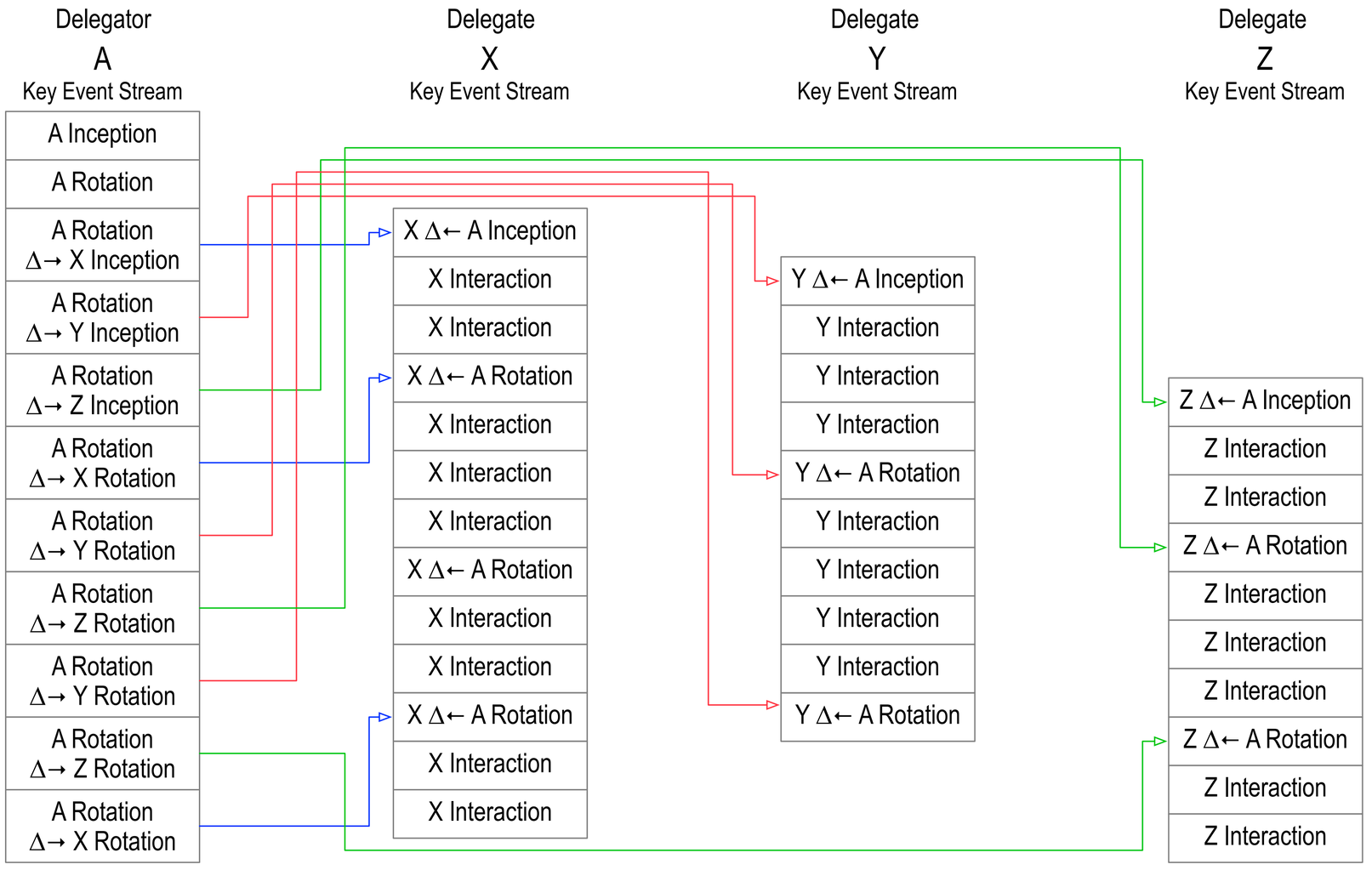


Rotation Delegation





Scaling Delegation via Rotation



Δ→ X : Delegation to X

Δ← A : Delegation from A

KERI for the *DID*ified

KERI non-transferable ephemeral with derivation code ~ did:key

KERI private direct mode (one-to-one) ~ did:peer

KERI public persistent indirect mode (one-to-any) ~ Indy interop, did:sov etc

KERI = did:un (did:uni, did:u) (all of the above in one method)

`did:un:`*`prefix`*`[:`*`options`*`][/`*`path`*`][?`*`query`*`][#`*`fragment`*`]`

KERI Agnosticism and Interop

KERI itself is completely agnostic about anything but the *prefix* !

`???:prefix[:options][/path][?query][#fragment]`

The KERI layer establishes control authority over a *prefix*

Any and *All* namespaces that share the same *prefix* may share the same KERI trust basis for control establishment over that *prefix* and hence that namespace.

Interop happens in a layer above the KERI layer

All we need for bootstrapping interop is some indication that the *prefix* inside identifier is KERI based (KERI trust basis).

Self-Certifying Identifier Prefixes

All crypto material appears in KERI in a fully qualified representation that includes a derivation code prepended to the crypto-material.

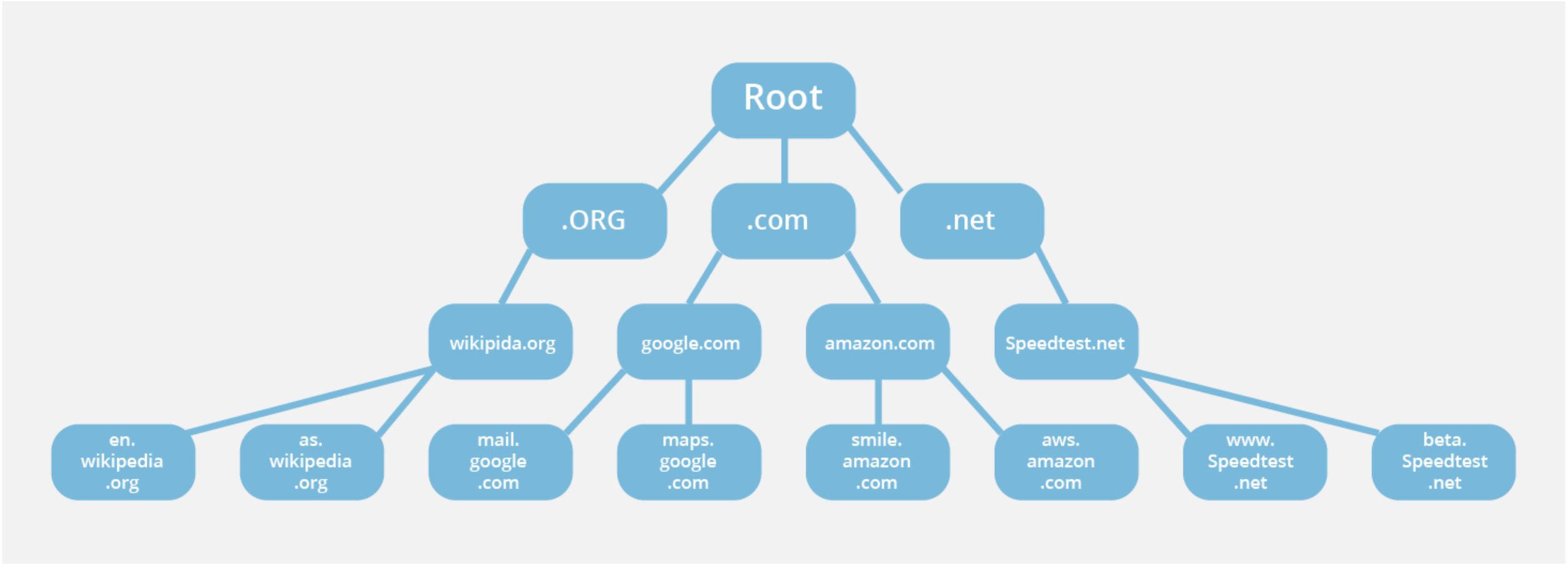
Identifier prefixes are fully qualified crypto-material.

Discovery

Ledger Based

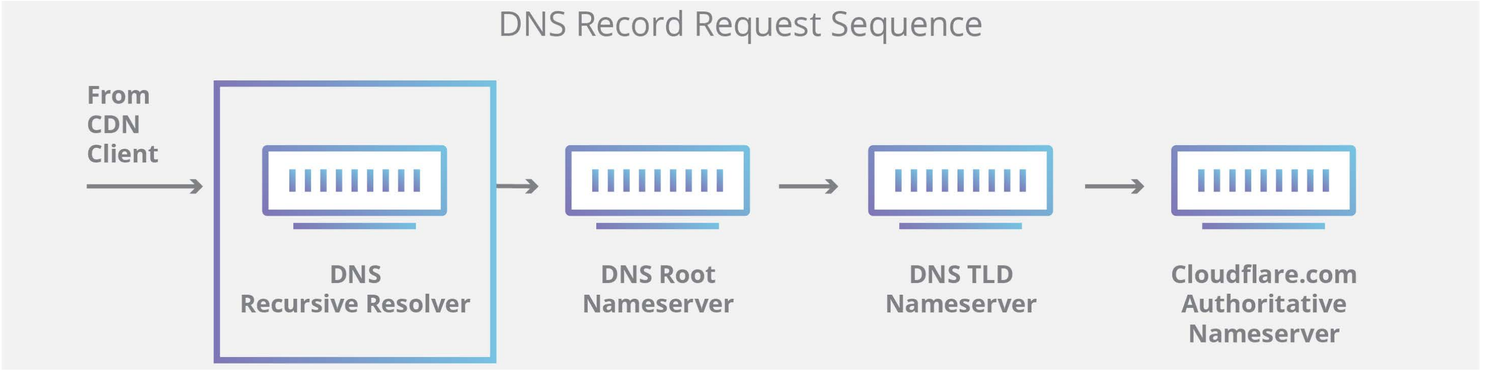
Non-Ledger Based

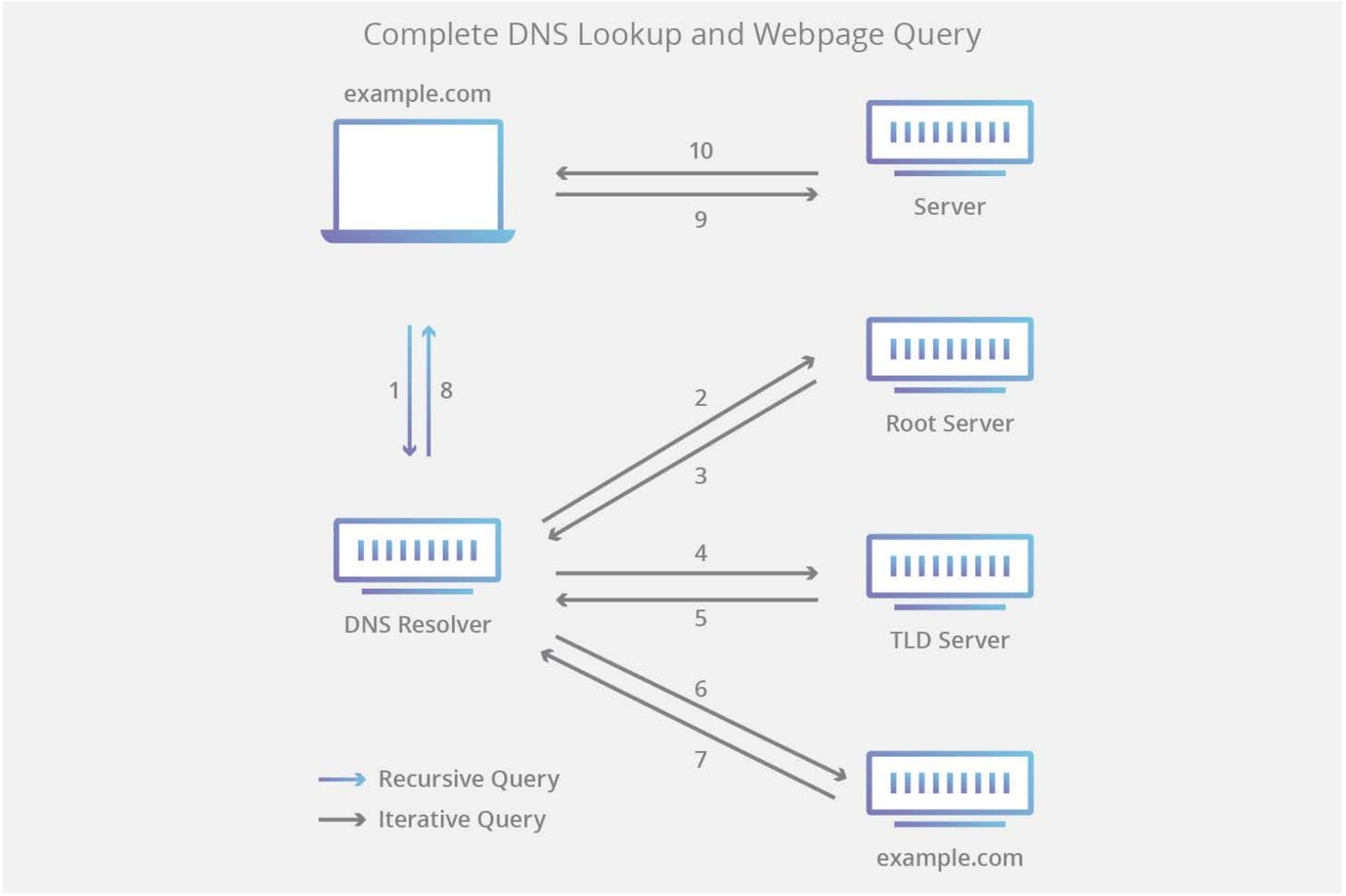
DNS “Hierarchical” Discovery



Diagrams from https://www.cloudflare.com/learning/dns/what-is-dns/

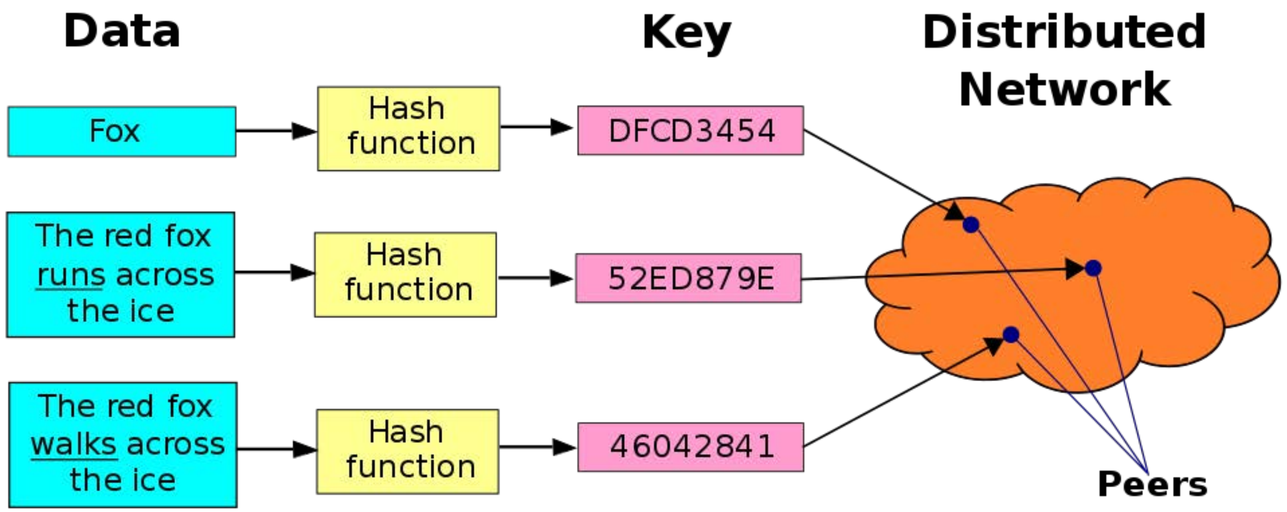
DNS “Hierarchical” Discovery





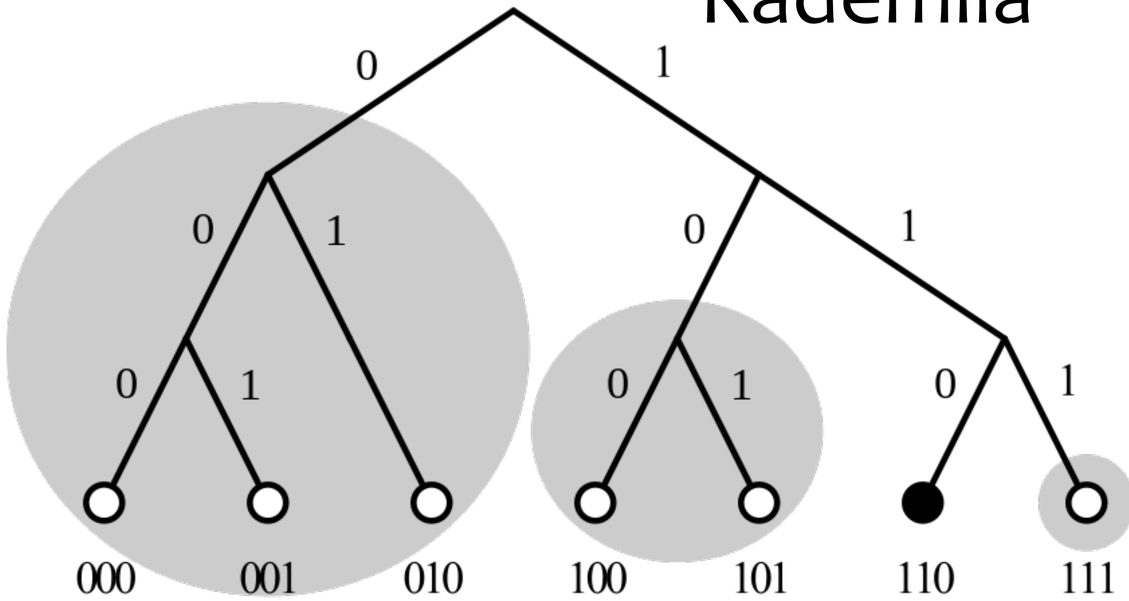

```
$ORIGIN example.com.
                        @               3600 SOA   ns1.p30.oraclecloud.net. (
                        zone-admin.dyndns.com.    ; address of responsible party
                        2016072701              ; serial number
                        3600                    ; refresh period
                        600                     ; retry period
                        604800                  ; expire time
                        1800                    ) ; minimum ttl
                        86400 NS    ns1.p68.dns.oraclecloud.net.
                        86400 NS    ns2.p68.dns.oraclecloud.net.
                        86400 NS    ns3.p68.dns.oraclecloud.net.
                        86400 NS    ns4.p68.dns.oraclecloud.net.
                        3600 MX    10 mail.example.com.
                        3600 MX    20 vpn.example.com.
                        3600 MX    30 mail.example.com.
                        60 A     204.13.248.106
                        3600 TXT   "v=spf1 includespf.oraclecloud.net ~all"
                        mail            14400 A     204.13.248.106
                        vpn               60 A     216.146.45.240
                        webapp            60 A     216.146.46.10
                        webapp            60 A     216.146.46.11
www                     43200 CNAME example.com.
```

DHT “Distributed” Discovery



Diagrams from https://en.wikipedia.org/wiki/Distributed_hash_table

Kademlia



Diagrams from https://en.wikipedia.org/wiki/Kademlia

DHT Discovery for KERI

Resolve Node Prefix to IP Mapping

Prefix to Inception/Latest Rotation Event Caching
-> Extract Witness Prefixes from Event

Witness Prefix to IP Mapping

KERL Query to Witness Node

Diagrams from https://en.wikipedia.org/wiki/Distributed_hash_table

Diagrams from https://en.wikipedia.org/wiki/Kademlia