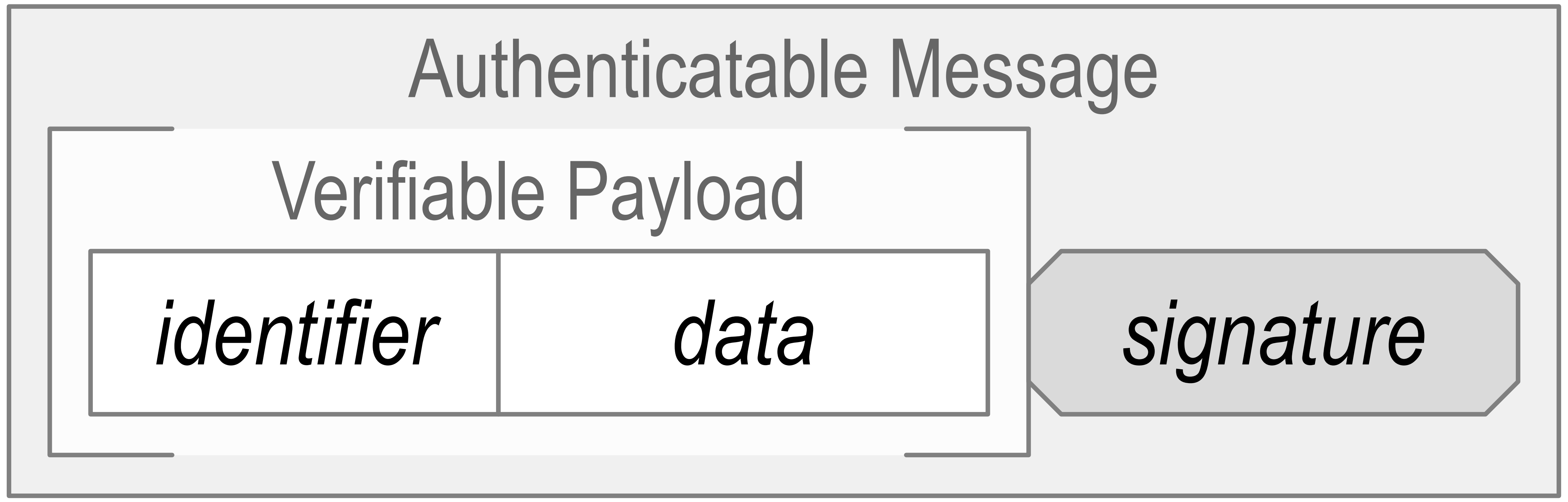
# ToIP
# Trust Spanning-layer Protocol (TSP) Proposal

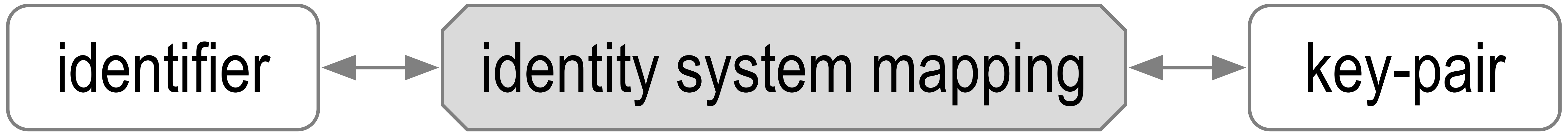
## A Secure Identifier Overlay for the Internet

*Samuel M. Smith Ph.D.*
*sam@keri.one*
*https://keri.one*
2023/02/08

# Identity (-ifier) System Security Overlay

Establish authenticity of IP packet’s message payload.

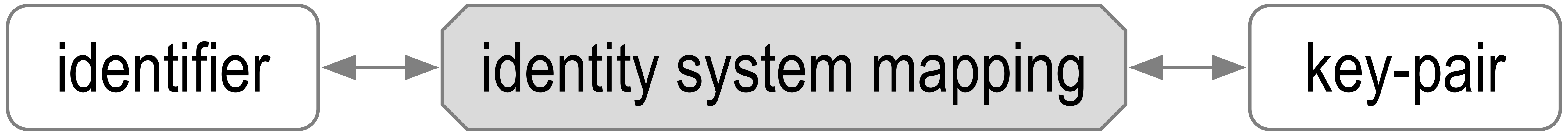




The overlay’s security is contingent on the mapping’s security.

# Identity (-ifier) System Security Overlay

persistent (transferable) mapping = verifiable data structure of key state changes

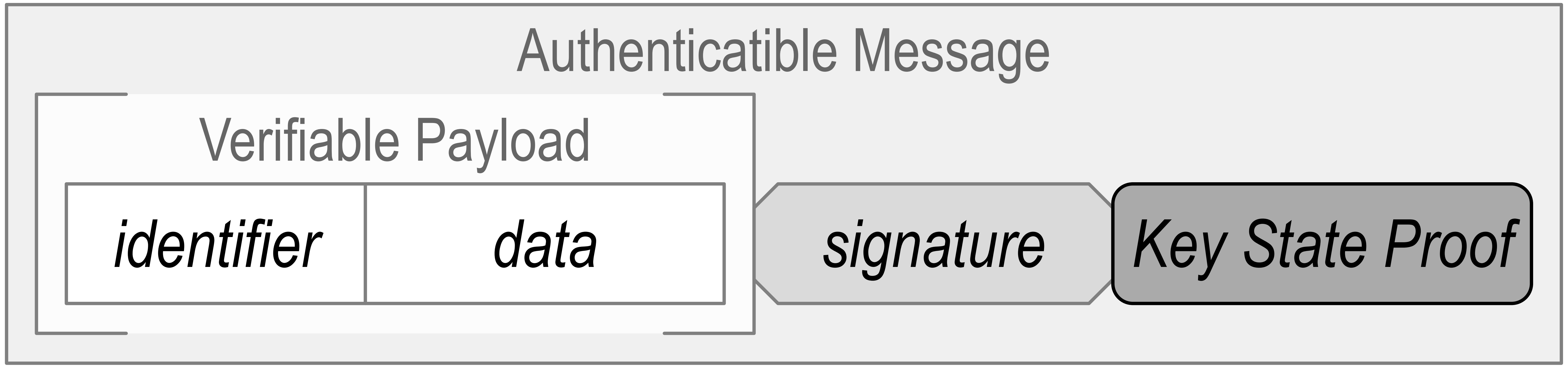


identifier

identity system mapping

key-pair

Establish authenticity of IP packet's message payload.



Authenticatible Message

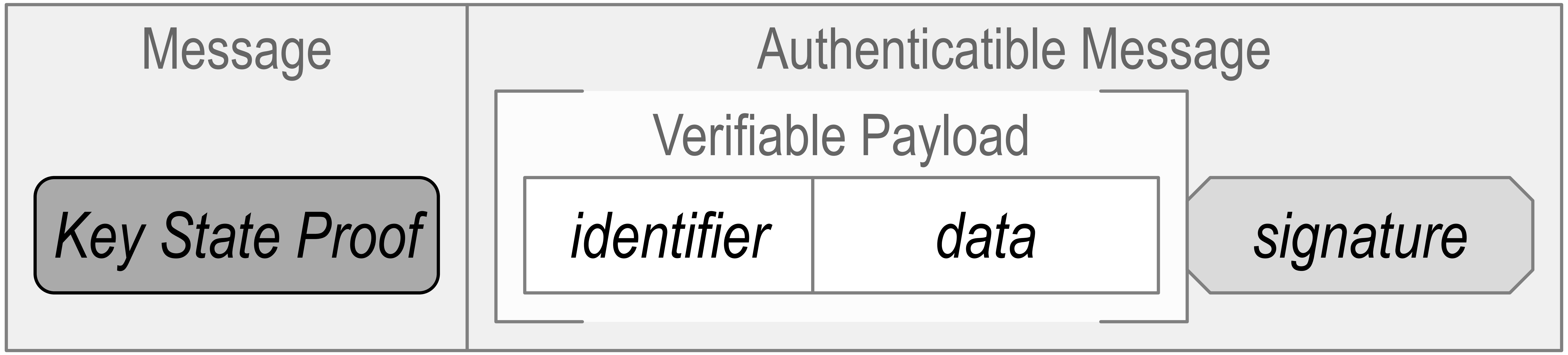
Verifiable Payload

*identifier*

*data*

*signature*

*Key State Proof*



Message

*Key State Proof*

Authenticatible Message

Verifiable Payload

*identifier*

*data*

*signature*

# What is an overlay?

Definition: *something laid over something else; covering; a layer, or decoration of something applied*

*Network Overlay (Gartner):* https://www.gartner.com/en/information-technology/glossary/overlay

An overlay is an installation of a networking component that is noninvasive to the wired infrastructure. Overlays generally employ tunneling techniques that connect end-point functionality to a central controller offering a variety of data, management, and control plane functions.

*Security Overlay (NIST):* https://csrc.nist.gov/glossary/term/security_control_overlay

A fully specified set of security controls, control enhancements, and supplemental guidance derived from tailoring a security baseline to fit the user's specific environment and mission.

*Identity System Security Overlay (KERI):* https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/KERI_WP_2.x.web.pdf

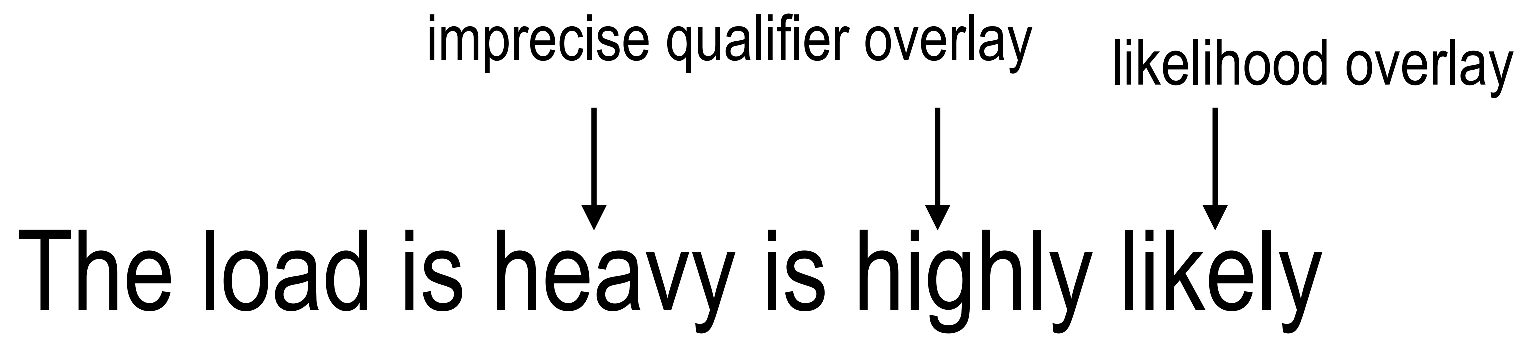
Provides a trust basis that binds controllers, identifiers, and key pairs. Provides persistent control over autonomic identifiers via the cryptographically verifiable provenance of each identifier's controlling key state. Provides verifiable control over supporting infrastructure, including other key pairs, that may be used to securely facilitate the authenticity, confidentiality, & privacy of interactions between controllers.

# Why an overlay?

- Efficiently imbue primitives (identifiers) in a statement with a given overlay's property(s) either singularly or in combination with other overlays.
- Modular flexible application to complex expressions.
- High expressive power.

Example: Uncertainty Overlays in Automated Reasoning (making computers think like people)

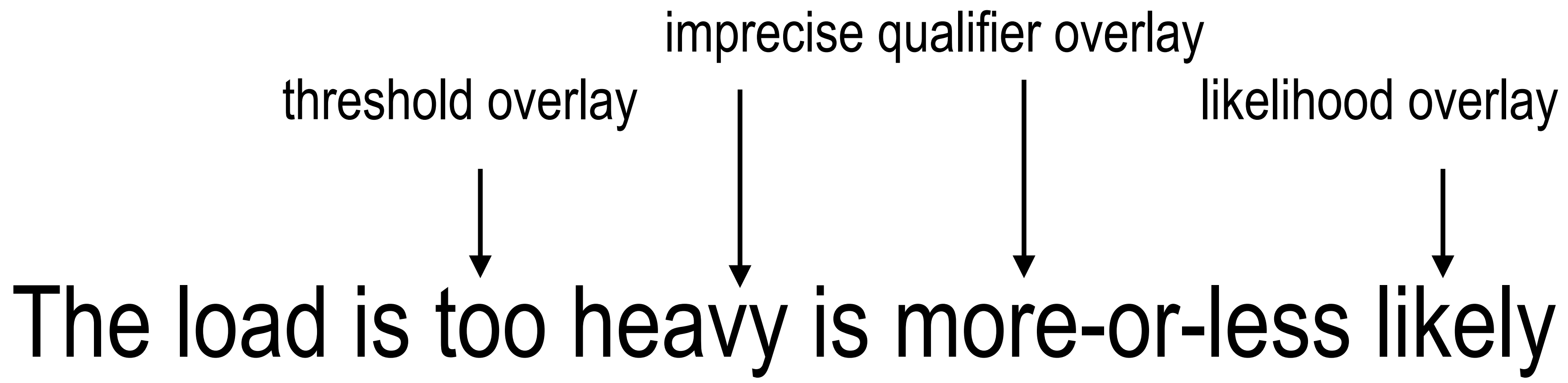
Three main types of uncertainty (each with its own overlay): imprecision, likelihood, and ambiguity.



imprecise qualifier overlay

likelihood overlay

The load is heavy is highly likely

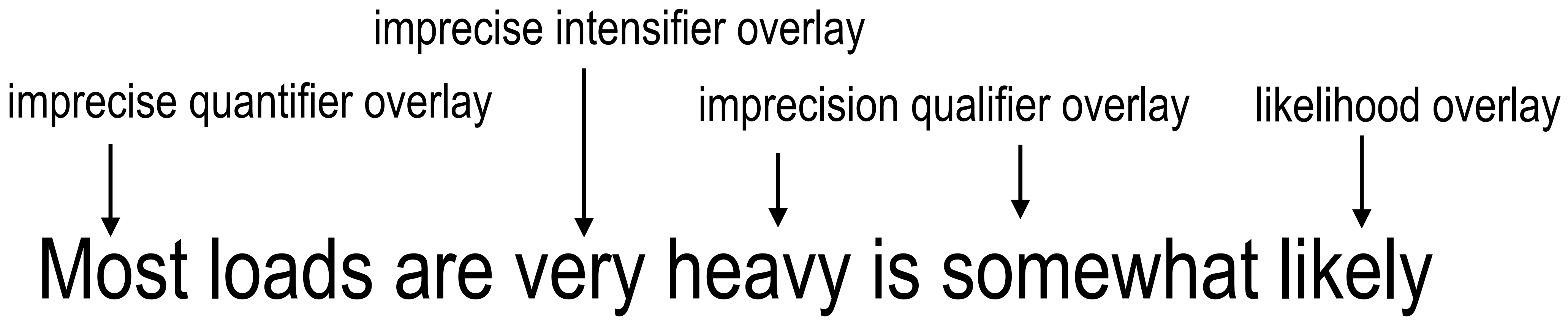


imprecise qualifier overlay

threshold overlay

likelihood overlay

The load is too heavy is more-or-less likely



imprecise intensifier overlay

imprecise quantifier overlay

imprecision qualifier overlay

likelihood overlay

Most loads are very heavy is somewhat likely

## Contextual Ambiguity

Loads for ships

Loads for dumptrucks

Loads for wheelbarrows

Loads for a plate of food

Loads for wheelbarrows when pushed by a construction worker

Loads for wheelbarrows when pushed by two power lifters

Loads for wheelbarrows when filled with gravel

Loads for wheelbarrows when filled with perlite

Loads for old rusty wheelbarrows with a flat tire

# Authenticity Overlay Example

In the yard, Ned the neighbor, Bob the builder, and Cam the construction worker who is pushing a wheelbarrow.

The load is heavy is ? likely given Ned said it.

The load is heavy is ? likely given Bob said it.

The load is heavy is ? likely given Cam said it.

The load is heavy is ? likely given anyone, including those not in the yard, could have said it.

The usefulness of any information in any decision process is dependent on the authoritativeness of its source.

If one can’t securely attribute the source one can’t ascertain the usefulness of the information.

The first property of received information that must be established by any decision maker is authenticity before any further processing can proceed.

# Authority Overlay Example

In the yard, Ned the neighbor, Bob the builder, and Cam the construction worker who is pushing a wheelbarrow.

The load is too heavy said Ned to Bob

The load is too heavy said Bob to Cam

The load is too heavy said Cam to Bob

Ned said to Bob, the load is too heavy and then Bob said to Cam, I concur with what Ned said.

The response of a party to a command (authorization) depends on the authority of the commanding party with respect to the commanded party which requires authentication of the commander by the commanded.

The load is too heavy said Cam to Ned

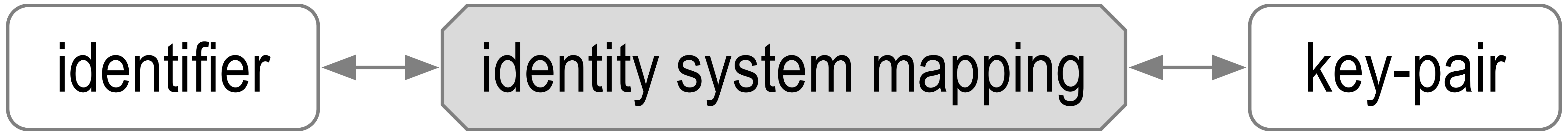
The load is too heavy said Bob to Ned

The load is too heavy said Bob to some unknown entity.

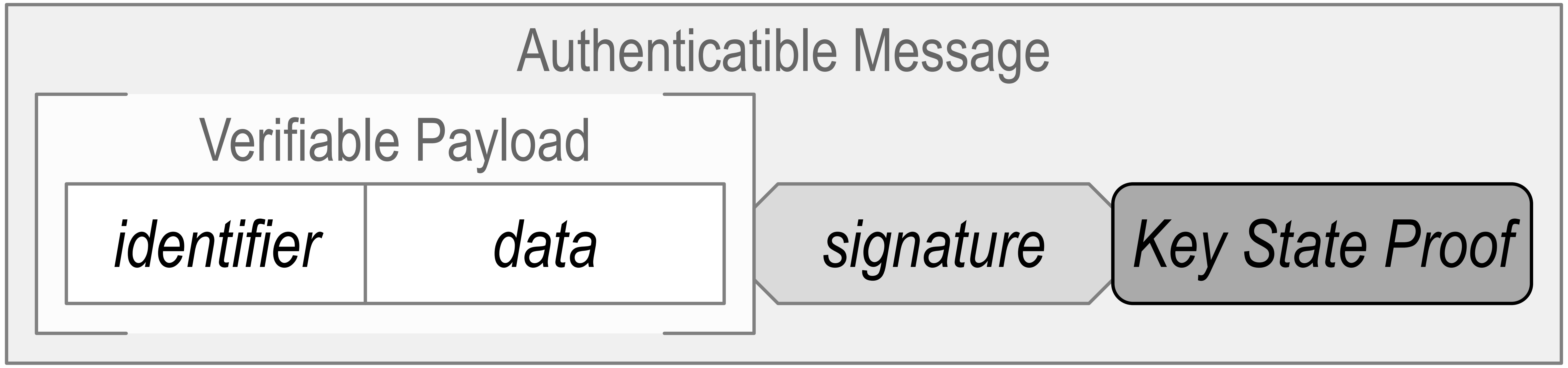
The effectiveness of a command is dependent on the authentication of the commanded by the commander.

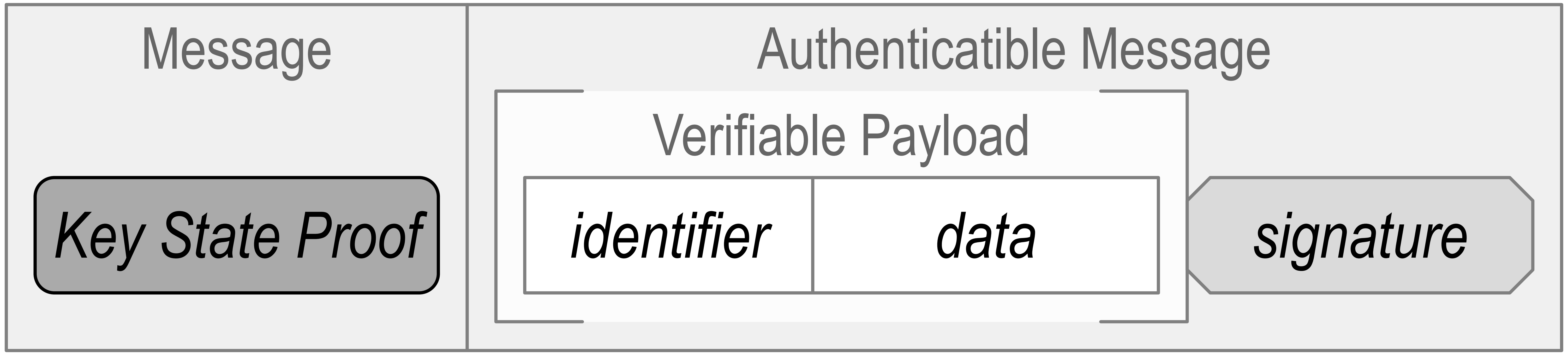
# Identity (-ifier) System Security Overlay

persistent (transferable) mapping = verifiable data structure of key state changes

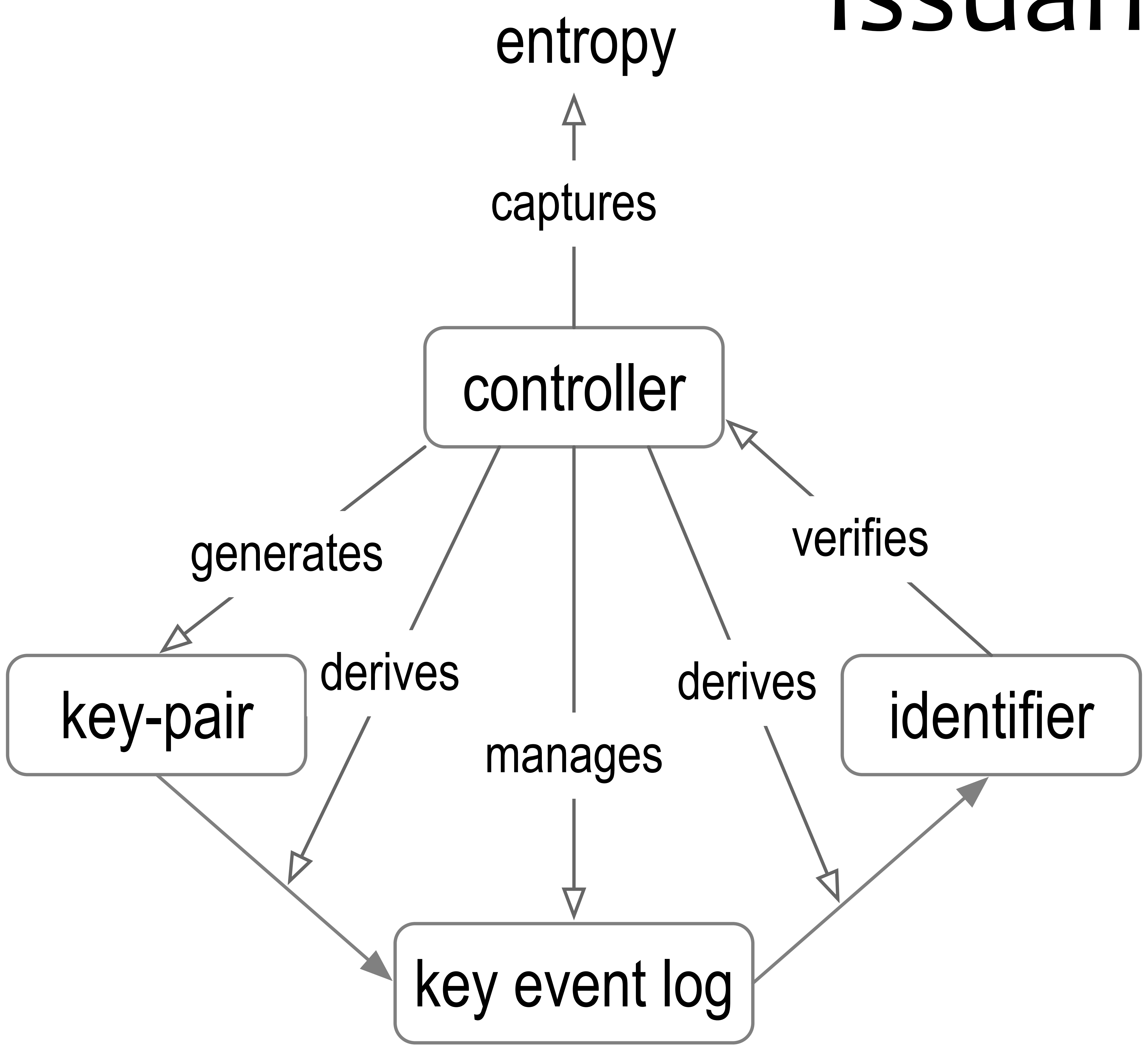


Establish authenticity of IP packet’s message payload.

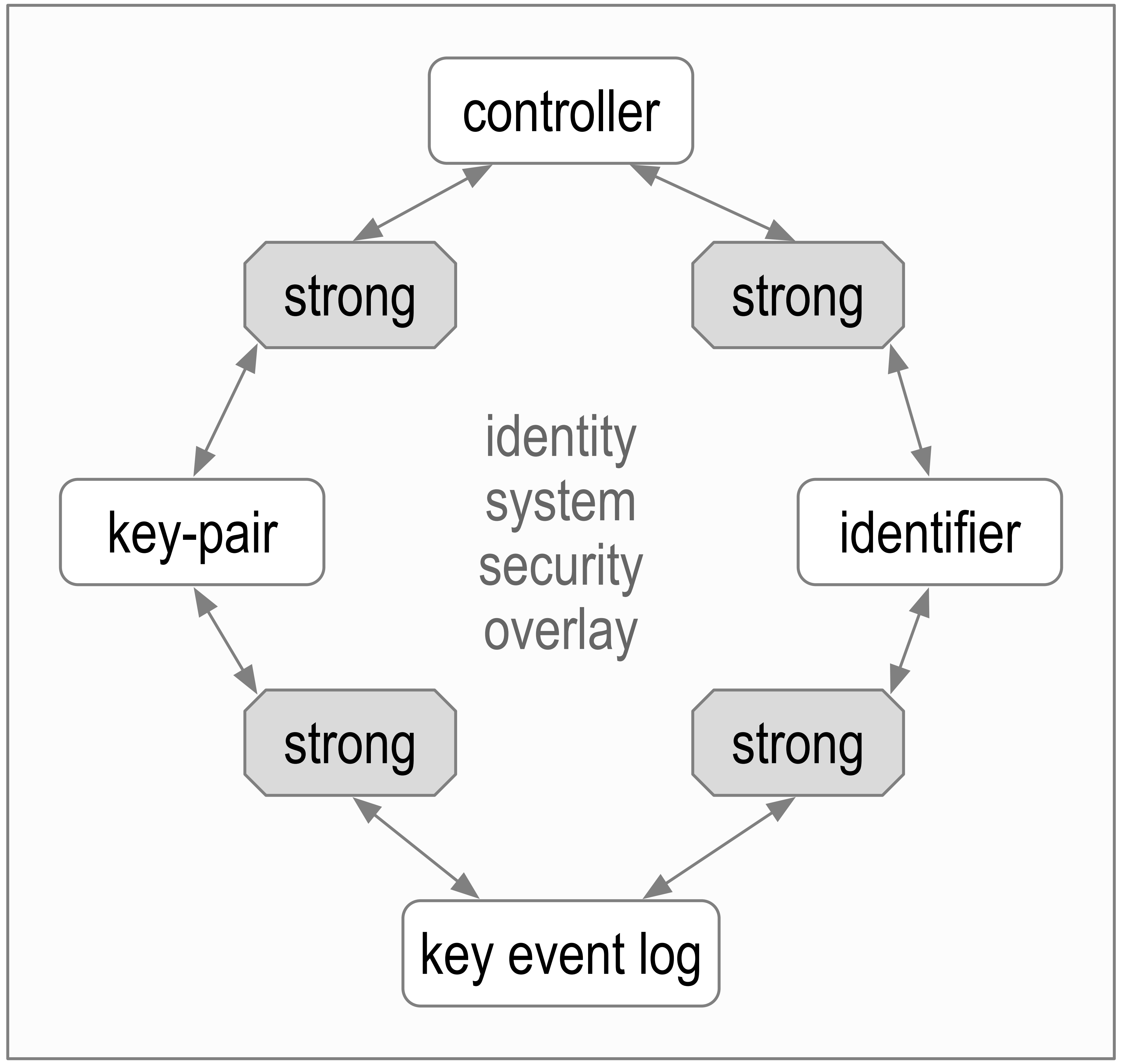




# Autonomic Identifier (AID) (Persistent): Issuance and Binding



autonomic, self-managing

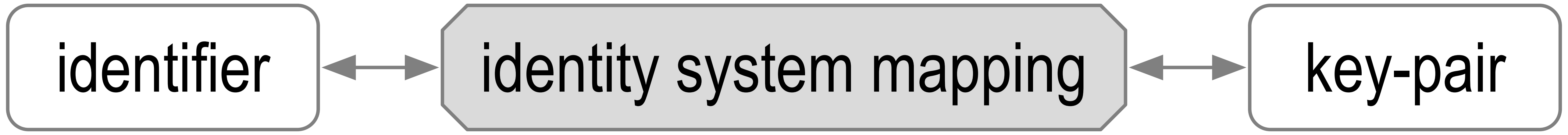


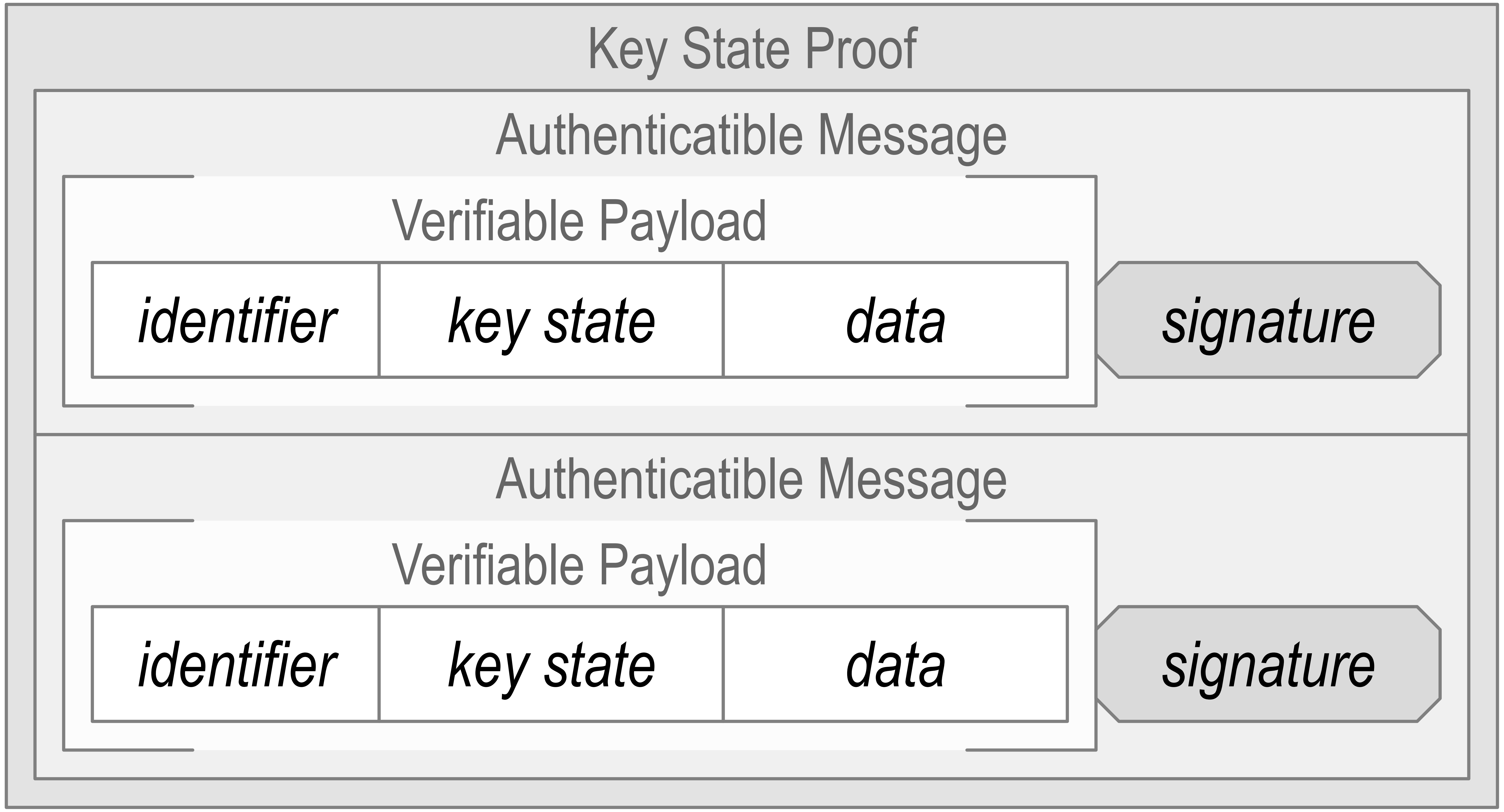
Autonomic Identifier Issuance Tetrad

cryptographic root-of-trust & verifiable persistent control

# Key State Proof is Recursive Application of Overlay

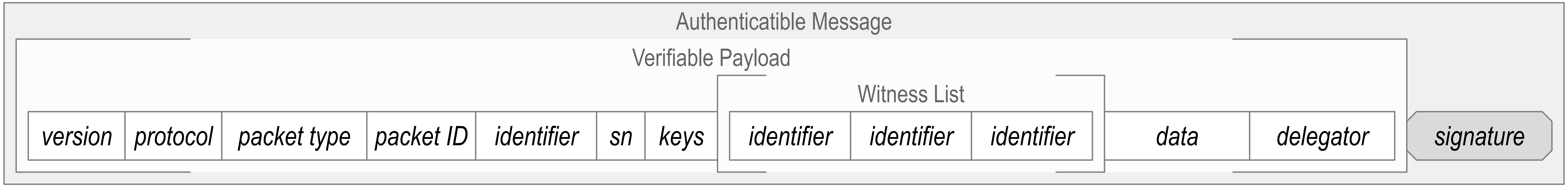
persistent (transferable) mapping = verifiable data structure of key state changes





# Wither *Destination* in Authentication Overlay

what counts as a *destination* identifier is contextual and may be entirely implicit

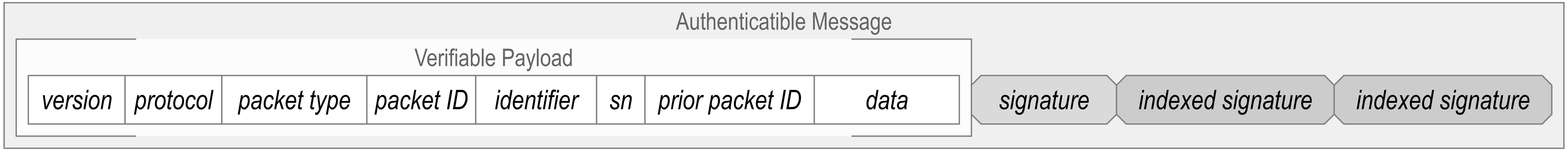


Authenticatible Message

Verifiable Payload

Witness List

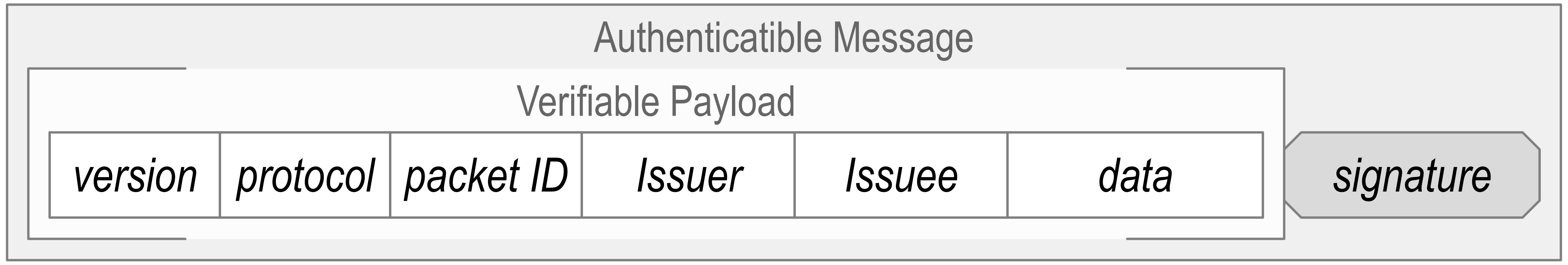
| version | protocol | packet type | packet ID | identifier | sn | keys | identifier | identifier | identifier | data | delegator | signature |
|---|---|---|---|---|---|---|---|---|---|---|---|---|



Authenticatible Message

Verifiable Payload

| version | protocol | packet type | packet ID | identifier | sn | prior packet ID | data | signature | indexed signature | indexed signature |
|---|---|---|---|---|---|---|---|---|---|---|



Authenticatible Message

Verifiable Payload

| version | protocol | packet ID | Issuer | Issuee | data | signature |
|---|---|---|---|---|---|---|

# Confidentiality Overlay Duality

A confidentiality overlay is the *dual* of an authenticity overlay

- *Authenticity*: Asymmetric key pair for signing, private key signs, any public key verifies
  - Only the key pair controller can sign with the private key, any recipient can verify with the public key.
- *Confidentiality*: Asymmetric key pair for en-de-cryption, public key encrypts, private key decrypts
  - Only the key pair controller can decrypt with the private key, any sender can encrypt with the public key

Any identifier can have a key state that includes both an asymmetric signing key pair and an asymmetric decryption key pair.

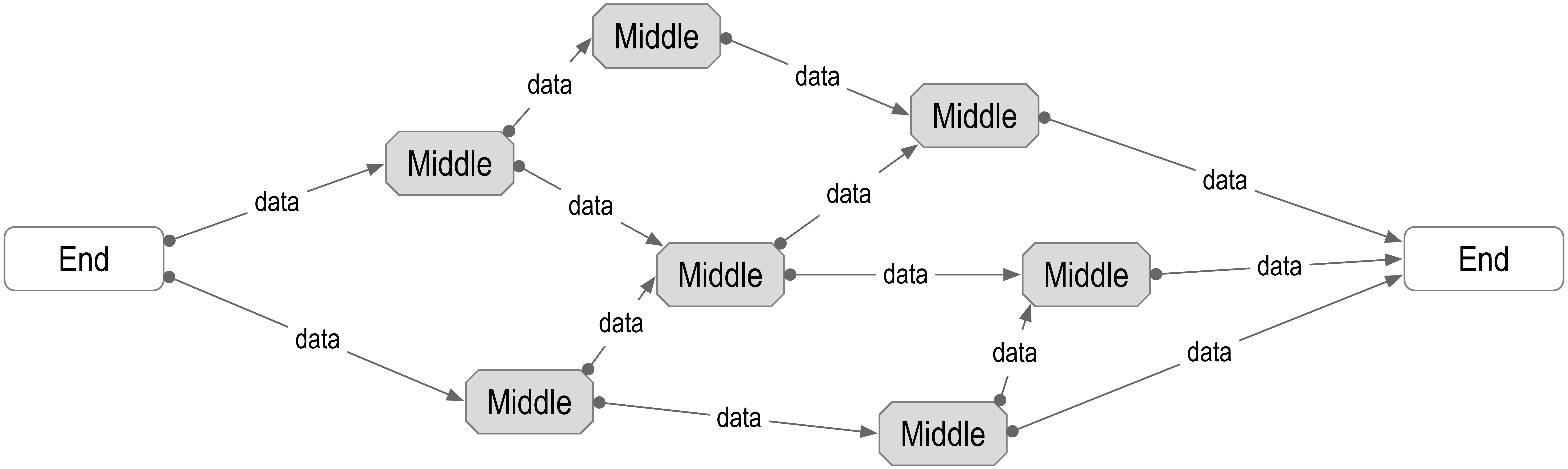
- Either the key pairs can be the same, or the decryption key pair can be derived from the signing key pair
- Thus an efficient approach is that only one key state, the signing key state, needs to be maintained. (with caveats)

Given the identifier and the security overlay's mapping to look up key state, any other party:

- Can verify with the signing public key that a message was non-repudiably sourced by the controller of the identifier (authenticity) (any-end-verifiable non-repudiable)
- Can encrypt a message using the encryption public key to ensure that only the controller of the identifier can view it (confidentiality) (only-end-viewable restricted)

# End Verifiability

*End-to-End* Verifiability *of* Authenticity



If the edges are secure, the security of the middle doesn’t matter.

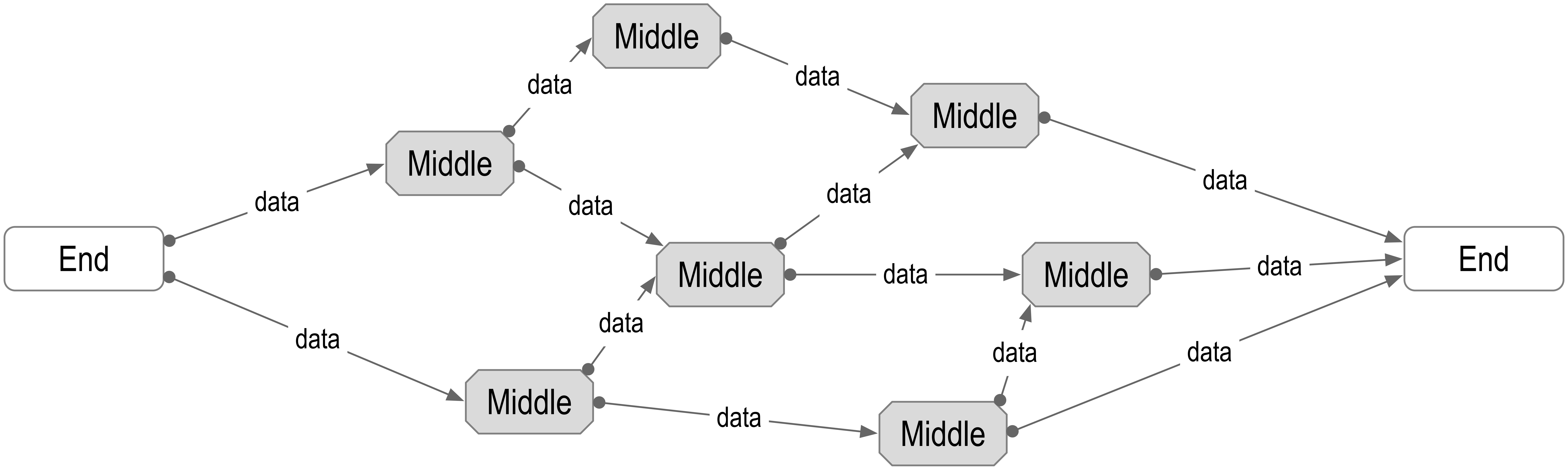
*Ambient Verifiability*: any-data, any-where, any-time by any-body

*Zero-Trust-Computing*

*Its much easier to protect one’s private keys than to protect all internet infrastructure*

# End Viewability

*Only-at-End Viewability* *via* Confidentiality



If the edges are secure, the security of the middle doesn’t matter.

*End only Viewability*: one-data, one-where, one-time by one-body

*Zero-Trust-Computing*

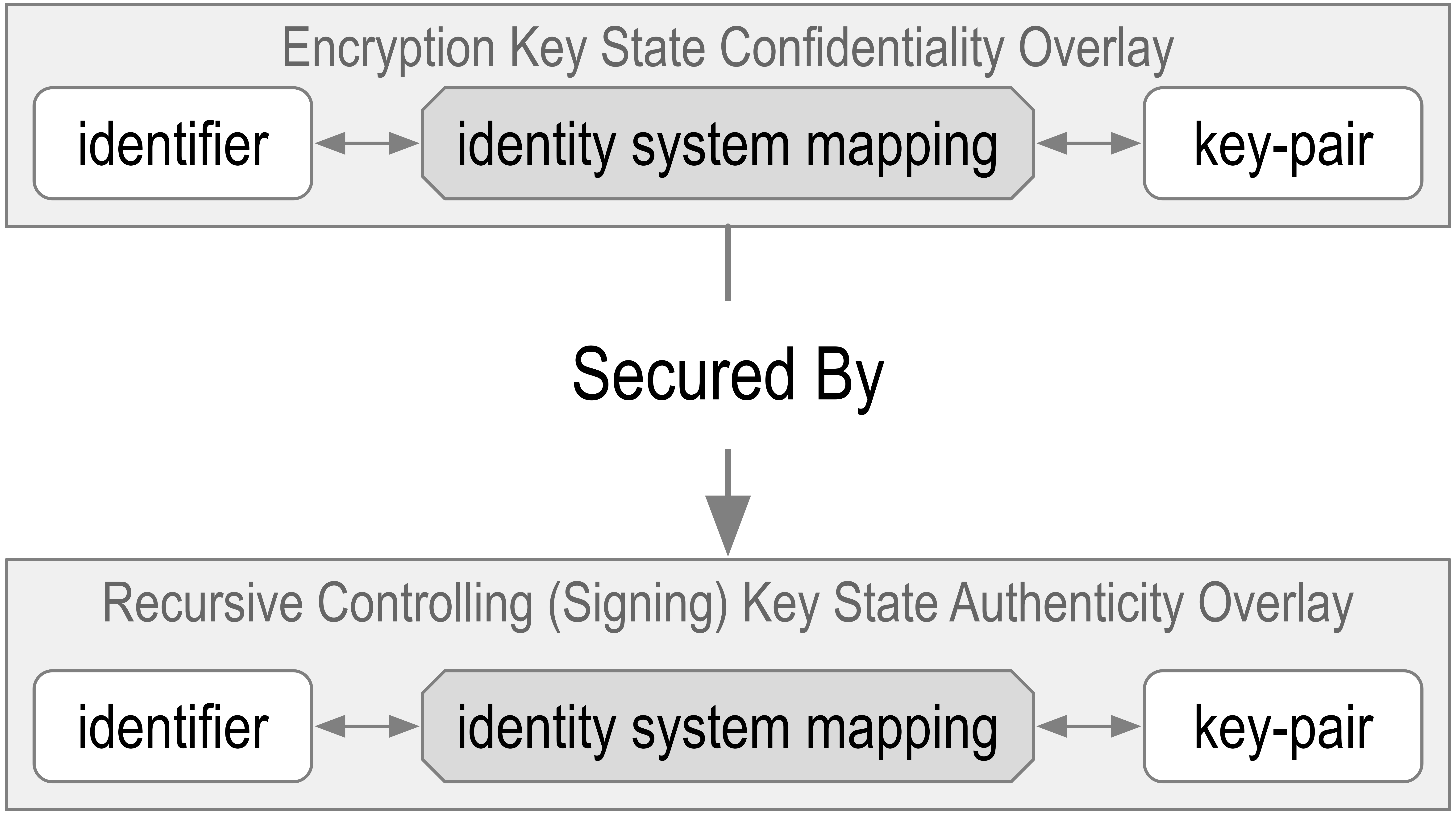
*Its much easier to protect one’s private keys than to protect all internet infrastructure*

# Confidentiality Overlay Dependency

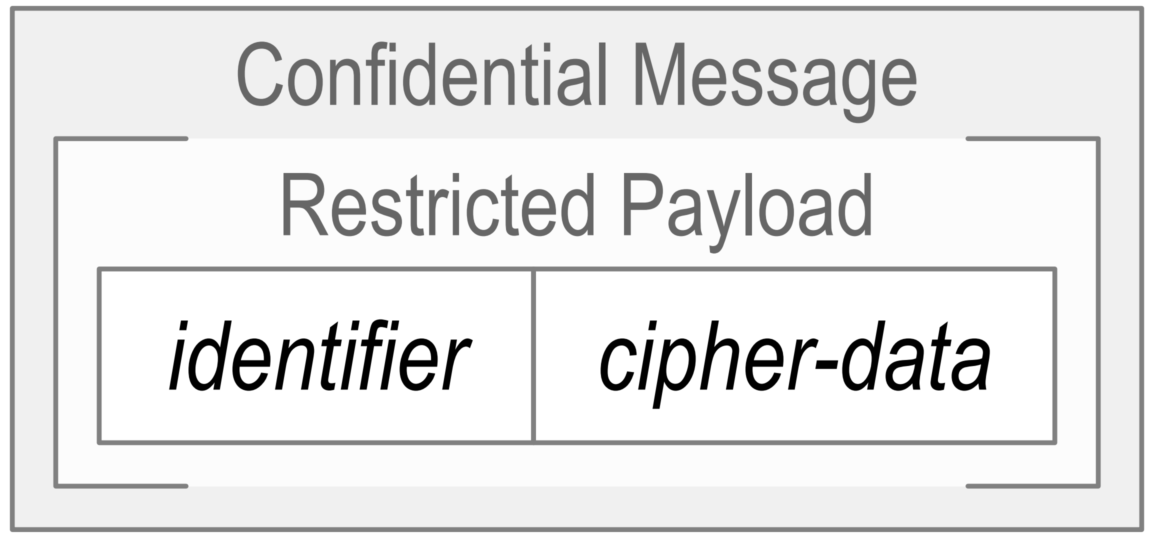
The encryption key state mapping can leverage the same persistent control mechanism as the signing key state mapping.

Indeed, the encryption key state must be securely attributable to the intended identifier for confidentiality to work.

In this strong sense, the confidentiality overlay depends on the authenticity overlay.


Encryption Key State Confidentiality Overlay
identifier
identity system mapping
key-pair
Secured By
Recursive Controlling (Signing) Key State Authenticity Overlay
identifier
identity system mapping
key-pair

# Confidentiality Overlay Elements

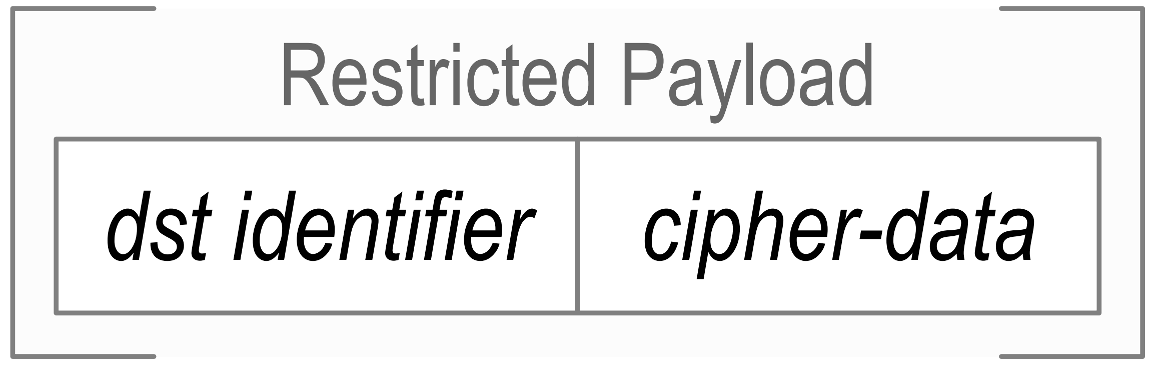


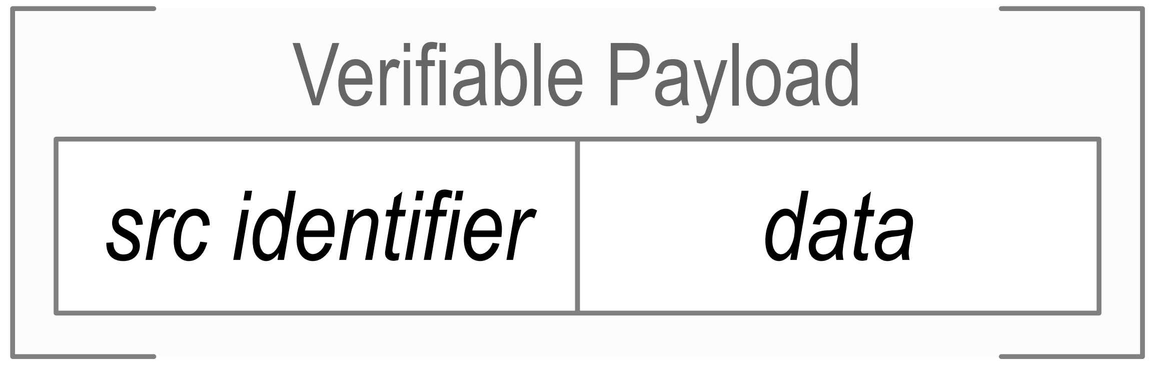
Strong Confidentiality Features:

3 party model: 1st party is sender, 2nd party is intended receiver, 3rd party unintended receiver

- 3rd party non-viewability
- 2nd party  partition-ability by 1st party
  - Detectable leakage (2nd party to 3rd party)
  - Detectable collusion (2nd party to 2nd party)
- 1st party non-view-ability
- 1st party non-collude-ability by 2nd party (1st party to 1st party or 1st party to 3rd party)

# Layering Confidentiality & Authenticatability Overlays

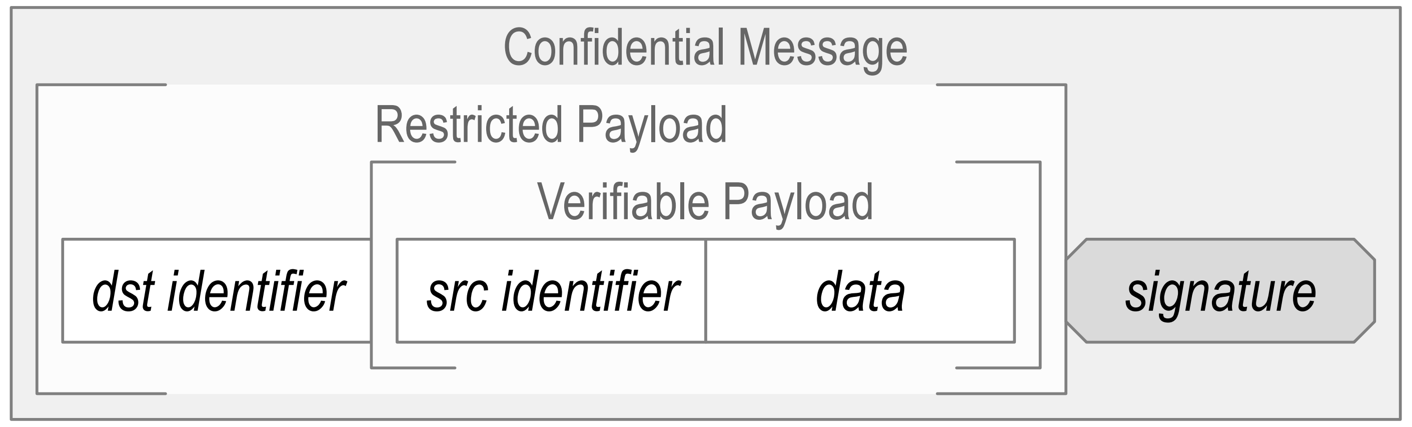


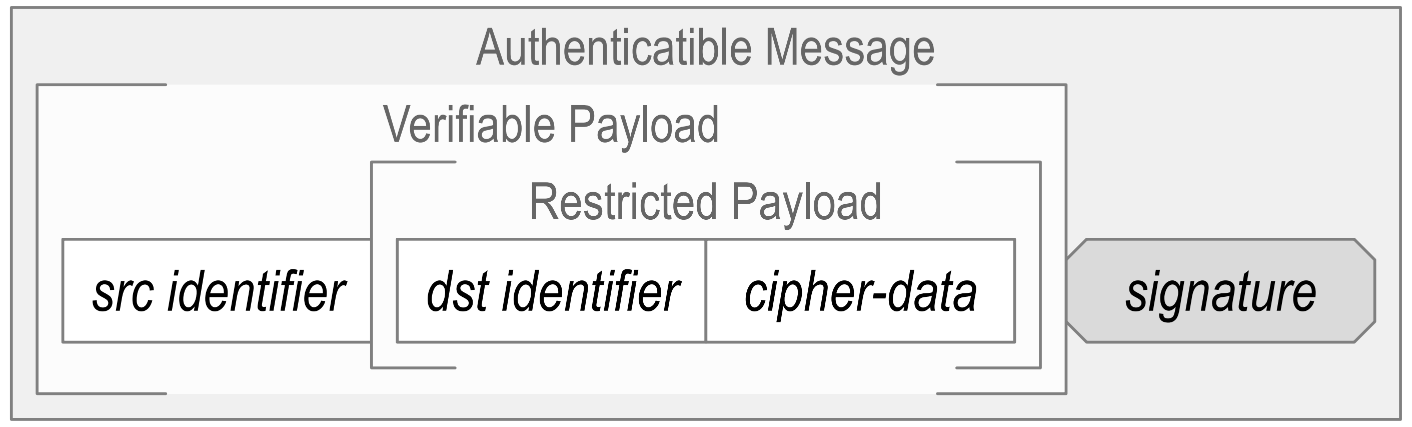


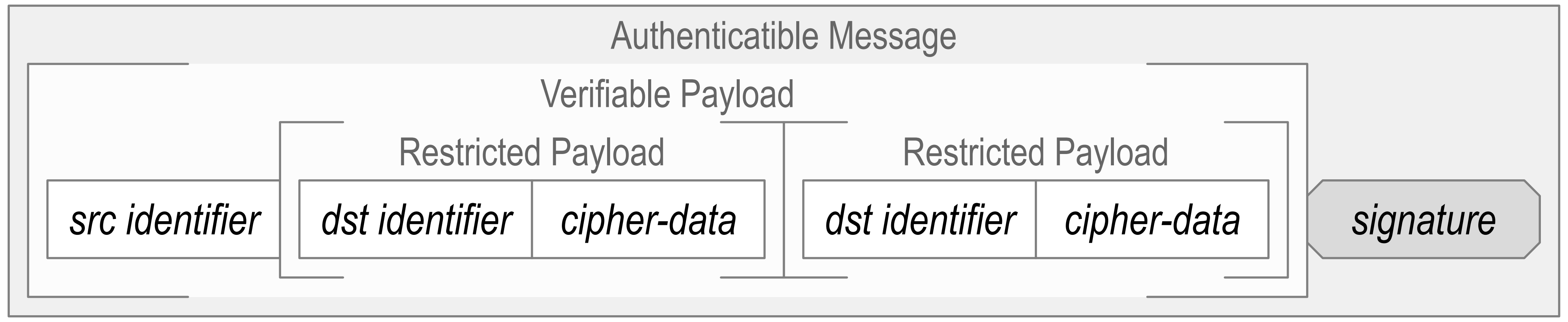
Setups matter!  Order of layering is setup dependent, including the setups of any trusted intermediaries

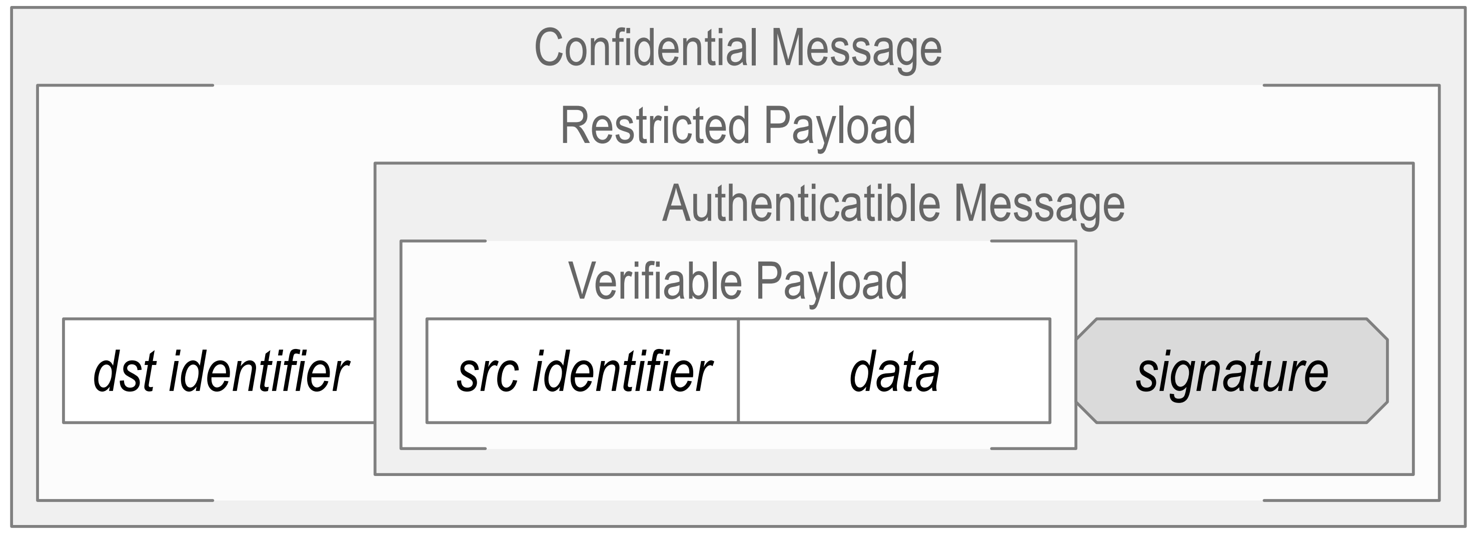
The presence or absence (explicit vs. implicit) of src and dst identifiers in a message is setup dependent

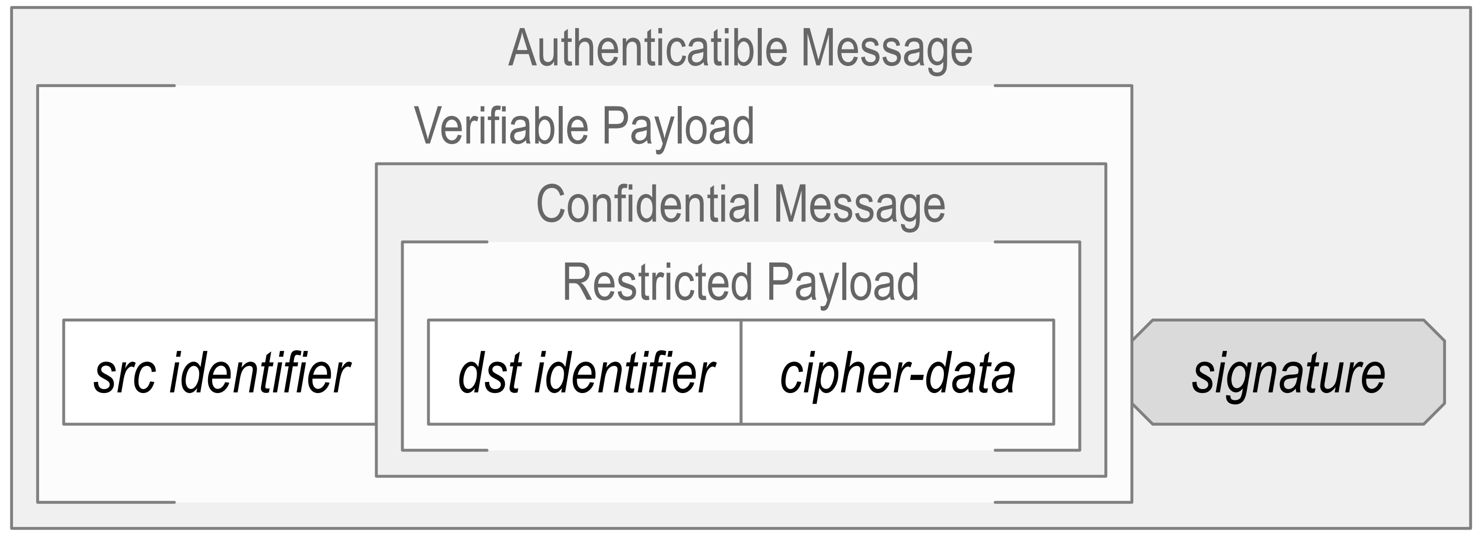
The hard problem of setups for persistent identifiers is to avoid repeating non-scalable manual setups like OOBA in order to ensure persistent control (solved by pre-rotated provenanced key state on cryptographic root-of-trust)











# Confidentiality Overlay Example

In the yard, Ned the neighbor, Bob the builder, and Cam the construction worker who is pushing a wheelbarrow.

Ned whispered to Bob, the load is too heavy and then Bob said to Cam the load is too heavy.

versus

Ned said to Bob, the load is too heavy and then Bob said to Cam, I concur with what Ned said.

versus

Cam whispered to Bob, the load is too heavy and then Bob said to Cam the load is too heavy.

versus

Cam said to Bob, the load is too heavy and then Bob said nothing.

Confidentiality enables a source of information to control the viewability of its information.

Control over confidentiality requires that a source can securely partition information meant for a given destination from other potential destinations which may require some level of authentication of the given destination prior to sending the information (setup context).

# Overlay: Trust Domain and Trust Basis

An *identifier system security* *overlay* controls the authenticity, confidentiality, & privacy of an interaction through recursive application of authenticity, confidential and privacy overlays (expanded consolidating definition).

Its *trust basis* binds controllers, identifiers, and key-pairs

Its *trust domain* is the ecosystem of *interactions* that relies on its *trust basis*

Each *trust basis* has one or more *roots-of-trust*

A *root-of-trust* is some component or process of the system upon which other components or processes are reliant. It has trustworthy security properties that provide a foundation of trust for components of a system.

A primary *root-of-trust* is irreplaceable

A secondary *root-of-trus*t is replaceable.

The application of an *identifier system security* *overlay* is to *map* its *trust basis* to its *trust domain* of *interactions*

# Overlay: Protocol Abstraction vs. Concrete Interaction

Overlay as a protocol abstraction where protocol determines type of overlay hence types of trust basis, trust domain, roots-of-trust, and interactions, …

versus …

Overlay as a concrete application of a specific identifier’s trust basis.

Concrete instantiation of any of the abstract types is per identifier not per identifier type!

Any interaction is concretized by a given message or set of messages wherein each message may involve multiple identifiers, each with a different controller and hence a different concrete trust basis.

The type of a given trust basis determines the type of its trust domain.

What happens when a given concrete interaction as overlaid message(s) mixes identifiers from trust bases of different types and hence trust domains of different types?

Is an ITDP even sensible in such a scenario?

# Trust Domain Appraisability

Trusted computing group uses the term *appraisal* to refer to the process of evaluating roots-of-trust for their security properties with regards the security policy of what is acceptable data secured by that root-of-trust (see also IETF RATS).

Generalizing, an *appraisable* trust-basis of some party to an interaction can be evaluated by any other relying party to that interaction with respect to the relying party's data acceptance policy.

How difficult is the appraisal of a given trust basis?

Some types of trust bases may have well-known easily appraisable characteristics that better facilitate trust transitivity in any given interaction relying on that trust basis versus other types of trust bases.

Maximal trust transitivity happens in an interaction when the trust bases of all parties are mutually appraisable with the same degree of trustability and at relative low appraisal cost.

# Types of Trust Bases

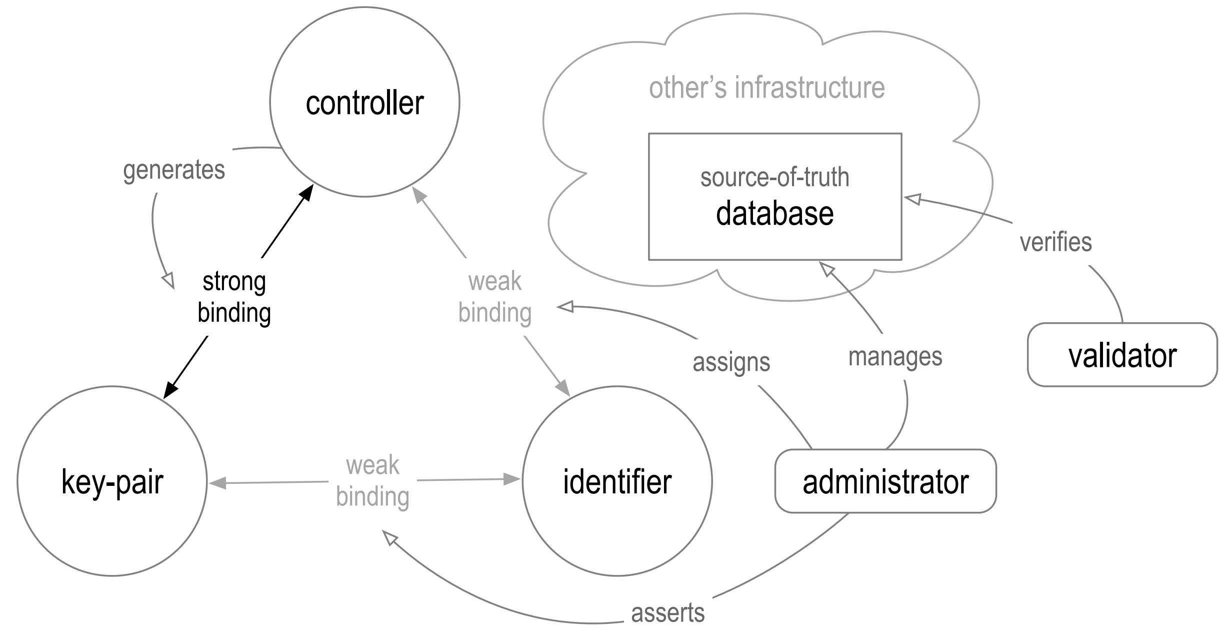
Given that maximal trust transitivity happens in an interaction when the trust bases of all parties are mutually appraisable with the same degree of trustability and at relative low appraisal cost …

Should we be fostering interoperability in interactions between poorly appraisable trust bases that either hide the security weaknesses or limit trustability to the lowest common denominator?

Analogy: Once non-repudiable digital signatures (using PKI) were invented, best practices deprecated HMACs for signing.

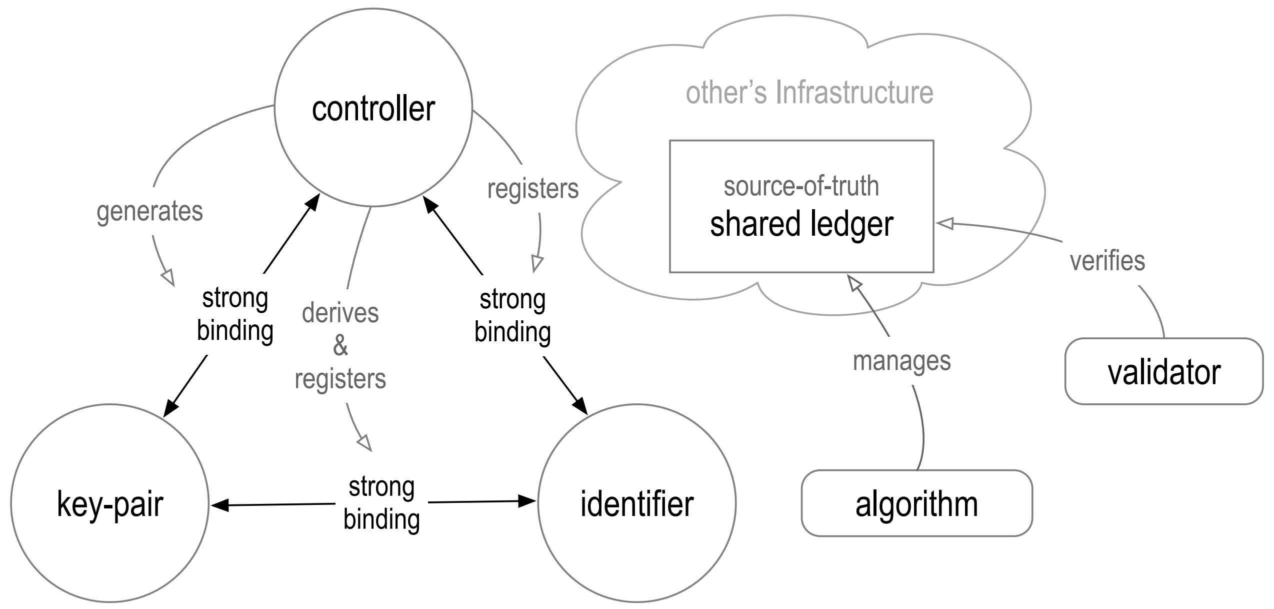
Is it not ludicrous to have a signed agreement where some parties use digital signatures, and some use HMACs?

## Administrative
## DNS/CA



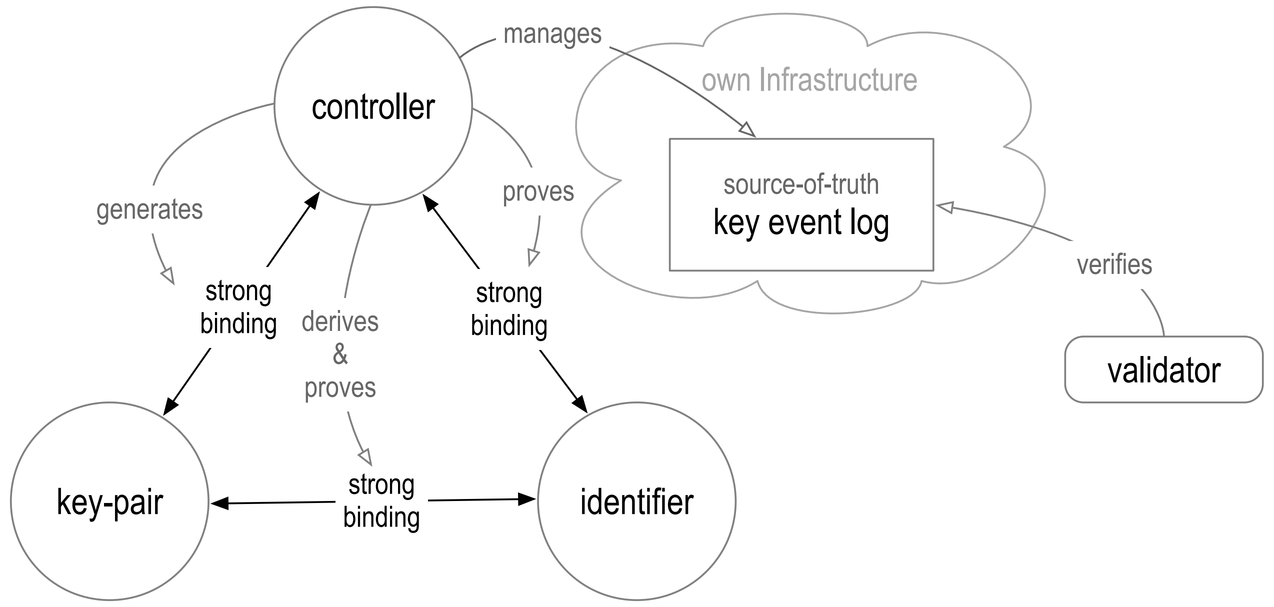
Opaque
Impractically High Appraisal Cost

## Algorithmic
## Shared Distributed Ledger



Duplicity Hiding
Non-monotonic (revisable)
Non-portable
Shared Control
Algorithmic Strength
More or Less Zero Trust
Relatively high appraisal cost

## Autonomic
## Cryptographically Verifiable



Duplicity Evident
Monotonic (non-revisable)
Portable
Non-Shared Control
Cryptographic Strength
Fully Zero Trust
Relatively low appraisal cost

# Zero-Trust Architecture: Data Protection

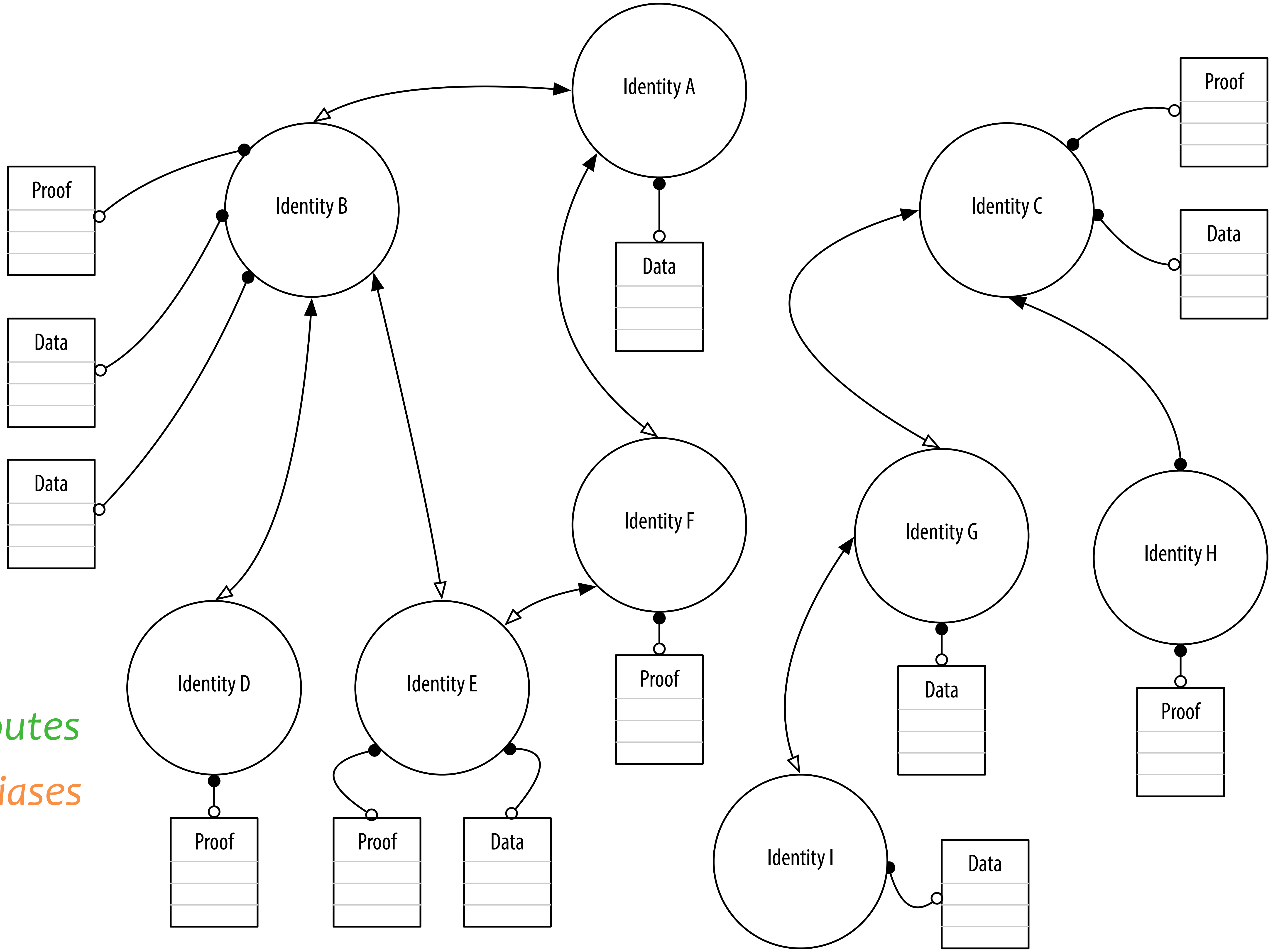
Never Trust, Always Verify

- Perimeter-less Security Model (necessary but not sufficient)
- Zero-Trust Spectrum = *ratio* of *trusted* surface to *verifiable* surface
- Trade-space axes of verifiable trust are *authenticity*, *confidentiality*, *privacy*
  - All protected data must have end-verifiable non-repudiable authenticity to its source (signing)
  - All protected data may have end-only viewable confidentiality to its destination (encrypting)
  - All protected data may have sufficient privacy amongst the parties to that data (correlating)

Data is signed and/or encrypted both *in motion* and *at rest* *(overlays should support both)*

- *at rest* means the storage mechanism is a *party* to the conversation
- *at rest* is an attack surface that *in motion* can’t protect against
- party identifiers may be *explicit* or *implicit* w.r.t protected data

# Self-Identity Graph



Identity = Identifiers + Attributes

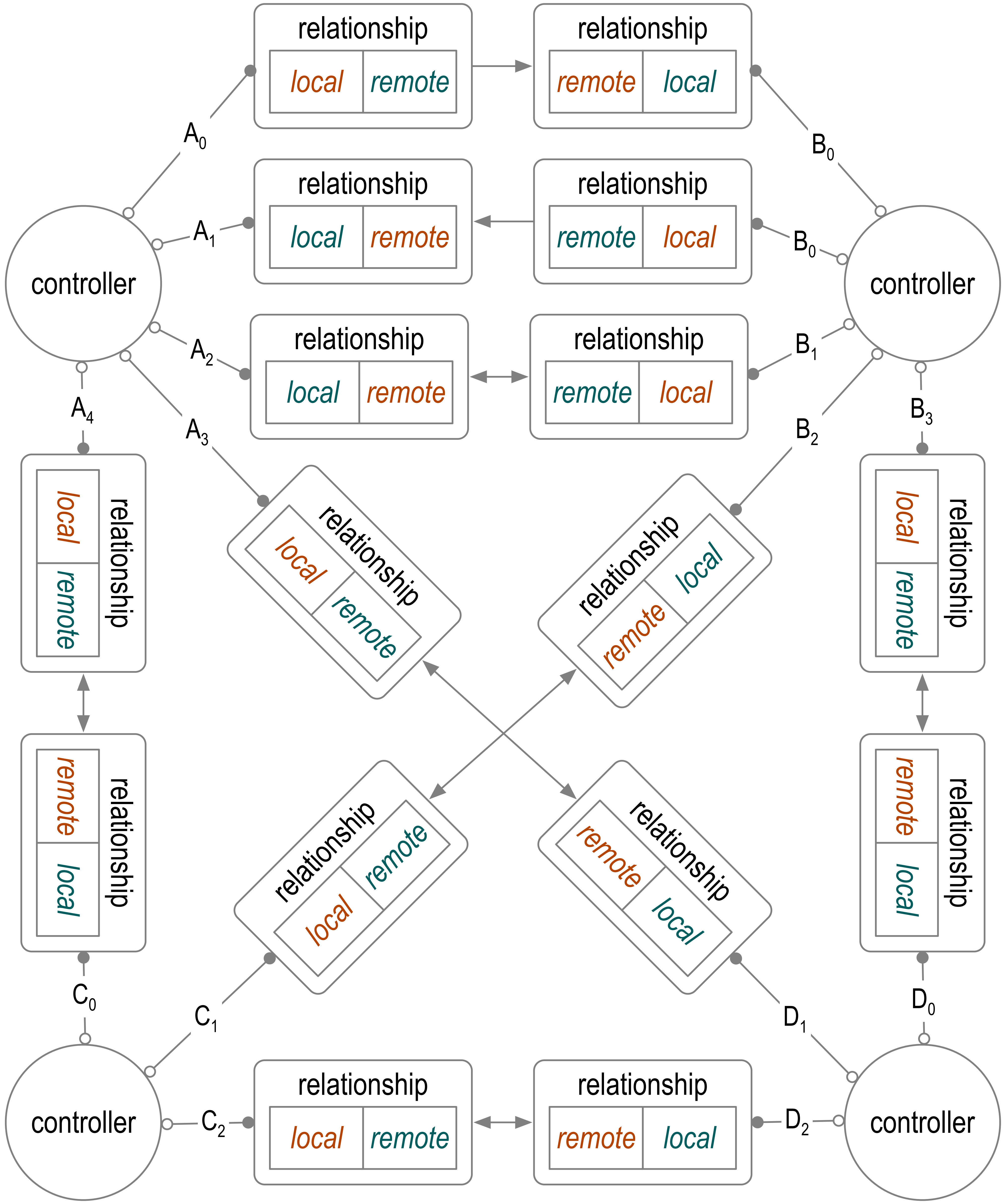
Identifiers = cryptonyms + aliases

Attributes = *data* + *proofs*

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

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

# Relationship Graph



A *relationship* is a pairing of two cryptonymous identifiers, one each from a different controller

- *relationships* are not communication channels
- *cryptonym* is a cryptographically derived pseudonym with at least 128 bits of entropy in the derivation
- *AID* is a cryptonym that is also securely attributable to a key pair
- two different cryptonyms are by themselves *uncorrelated* in an information-theoretic security sense in that knowledge of one by itself provides no information about the other
- *context* is a set of events. Two *contexts* are disjoint with respect to a *cryptonym* when that *cryptonym* appears in one or more events in one set but does not appear in any event in the other set.

A *partition* is a set of *contexts* with mutually disjoint relationships

- partitioned contexts may be correlatable due to other information associated with a each context but the partitioned relationships by themselves provide no correlatable information i.e. partitioned relationships are by themselves not correlatable
- partitioned relationships enable both secure attribution and secure confidentiality

A cryptonym that is common to a context  provides a perfectly correlatable feature within that context

the secure attributability of an AID together with its perfect correlatability within a context enable secure reputational trust (good or bad) in that AID within that context

Partitioned Relationships using  ORIs (One Relationship Identifiers)  balance the concerns of strong authenticity and strong confidentiality within context with sufficiently strong privacy across contexts.

# ToIP Design Goals

***Authenticity***:

is the receiver of a communication able to verify that it originated from the sender in a tamper evident form?

***Confidentiality***:

are the contents of a communication protected so that only authorized parties may have access?

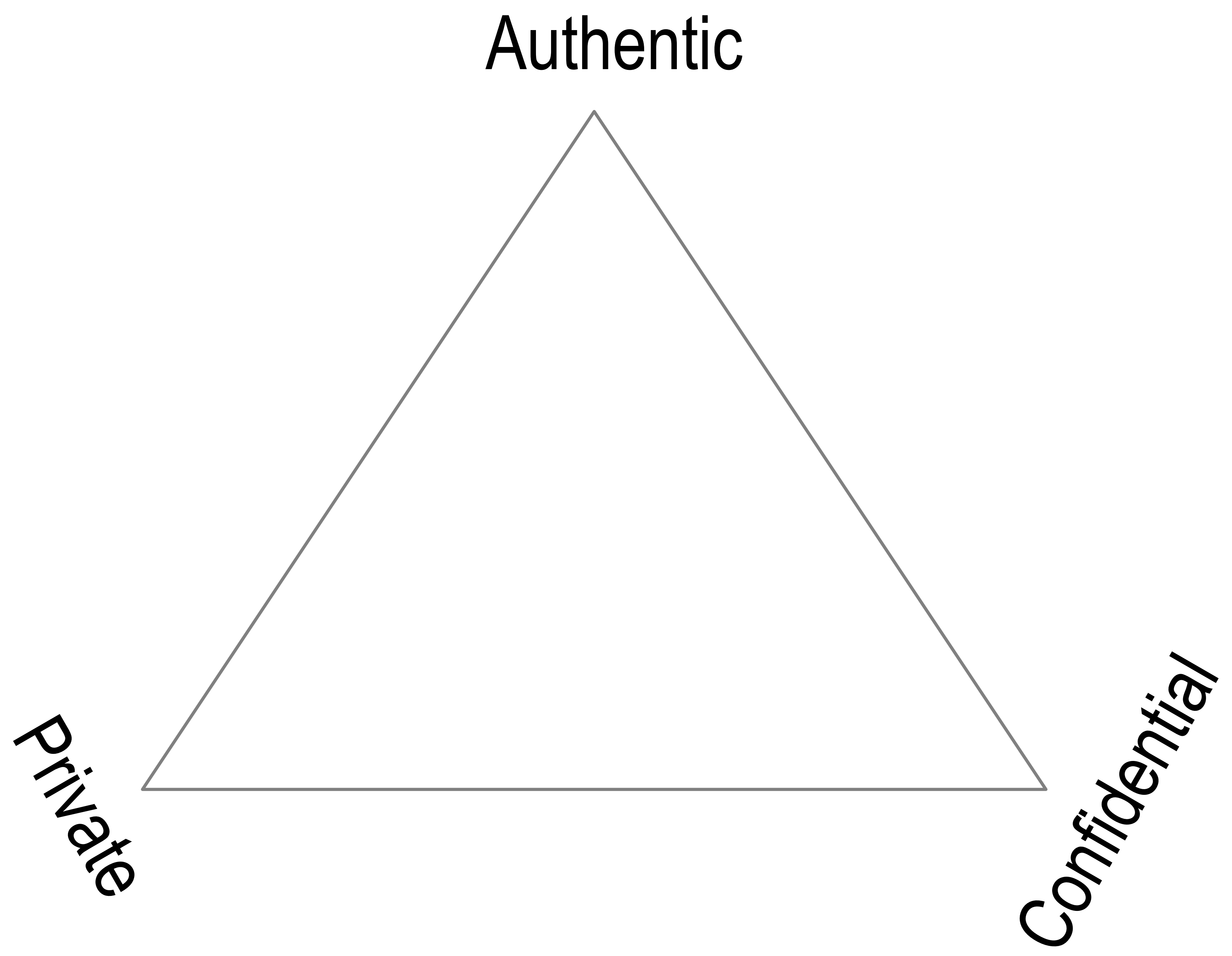
***Privacy***:

will the expectations of each party with respect to the usage of shared information be honored by the other parties?

*Note that, in some trust relationships, confidentiality and privacy may be optional. Thus the design goal is to achieve these three properties in the order listed.*

# PAC Theorem

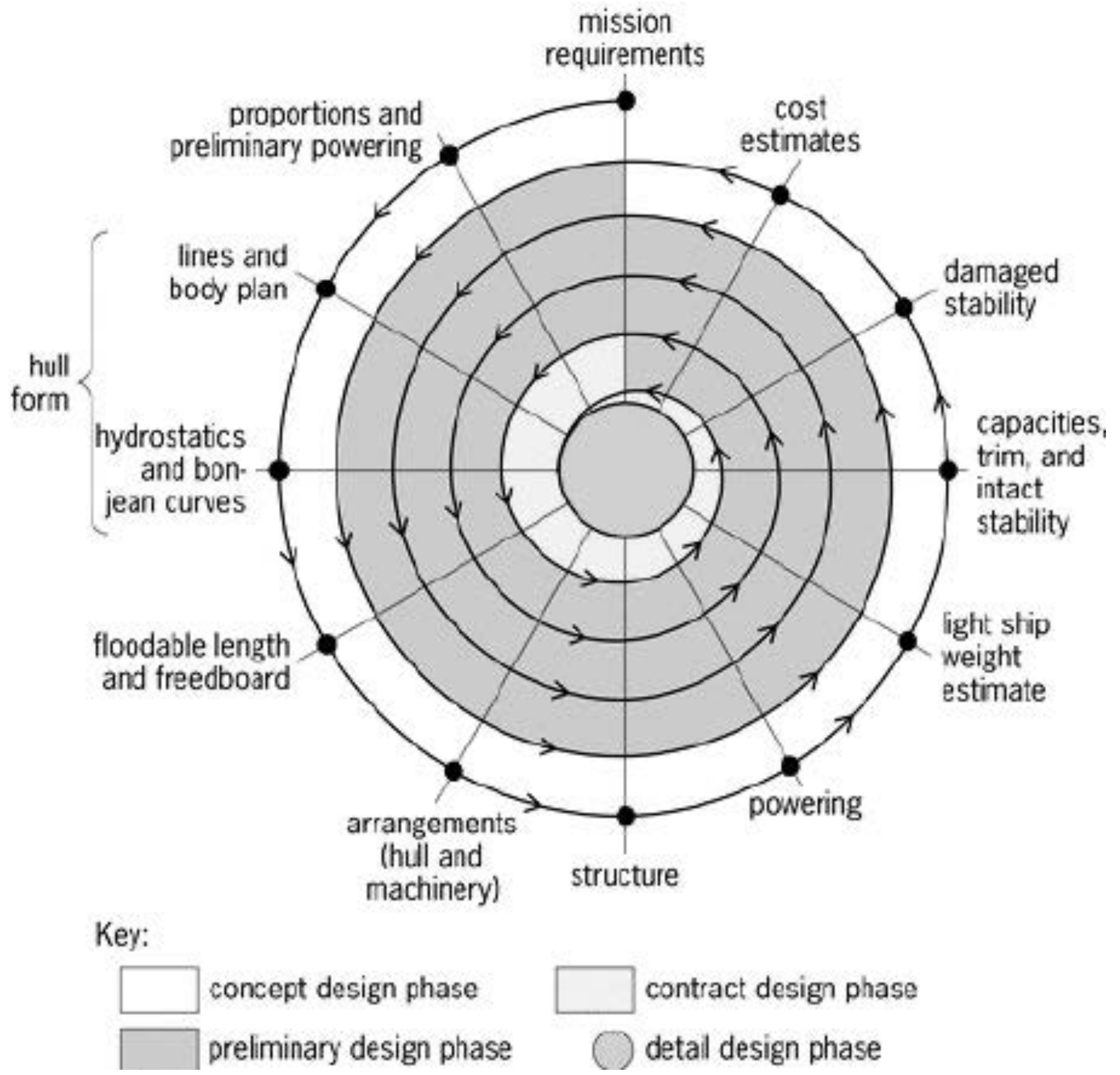
A conversation may be two of the three, *private*, *authentic*, and *confidential* to the same degree, but not all three at the same degree.



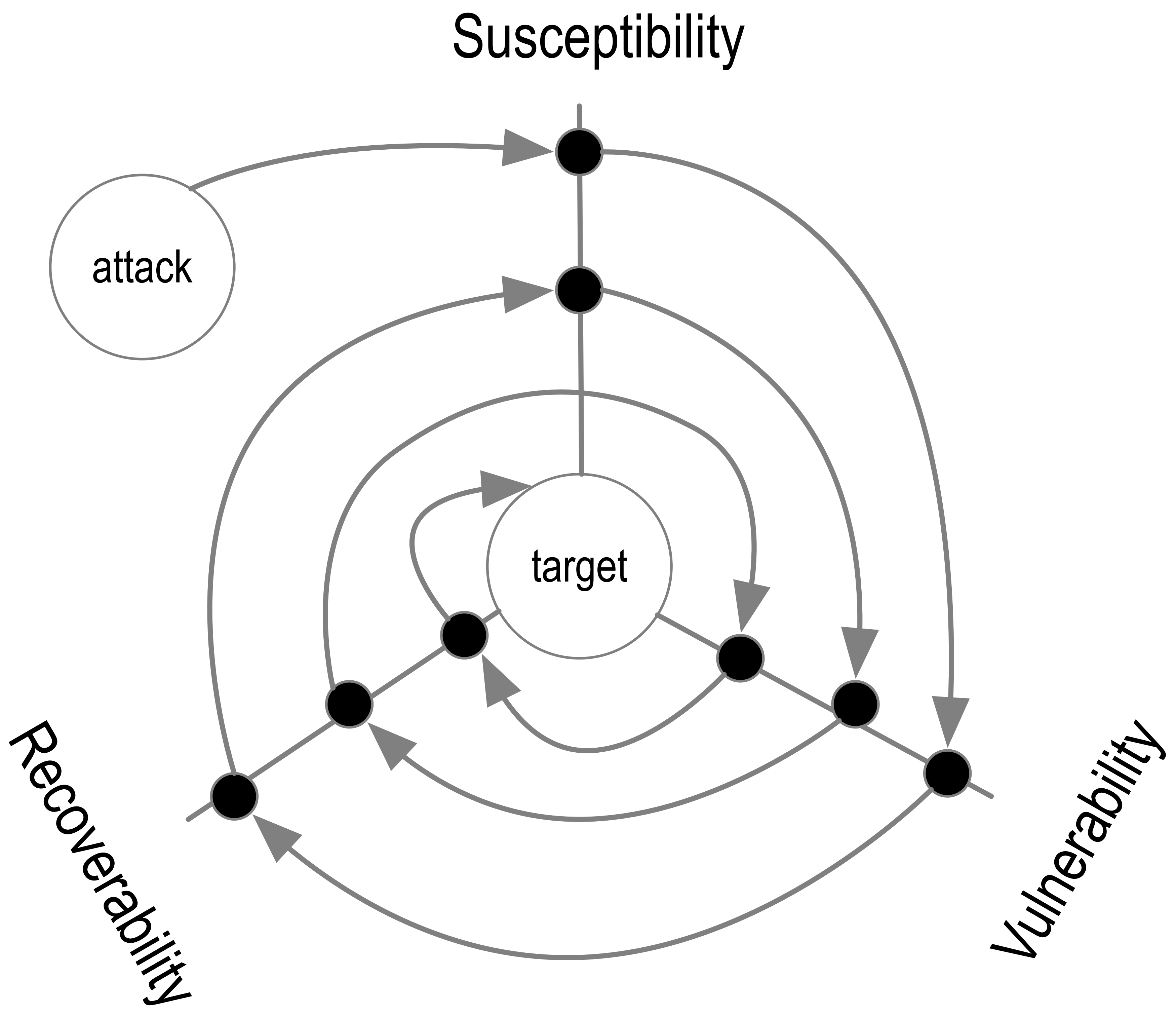
Trade-offs required!

# Multiple Overlay Spiral

Any set of mutually compatible overlays can be flexibly utilized in a spiral fashion
Where one enters, where one exits, where one stops, and how many times around the spiral is application dependent


mission requirements
cost estimates
damaged stability
capacities, trim, and intact stability
light ship weight estimate
powering
structure
arrangements (hull and machinery)
floodable length and freedboard
hydrostatics and bon-jean curves
hull form
lines and body plan
proportions and preliminary powering
Key:
□ concept design phase
▨ contract design phase
▨ preliminary design phase
○ detail design phase

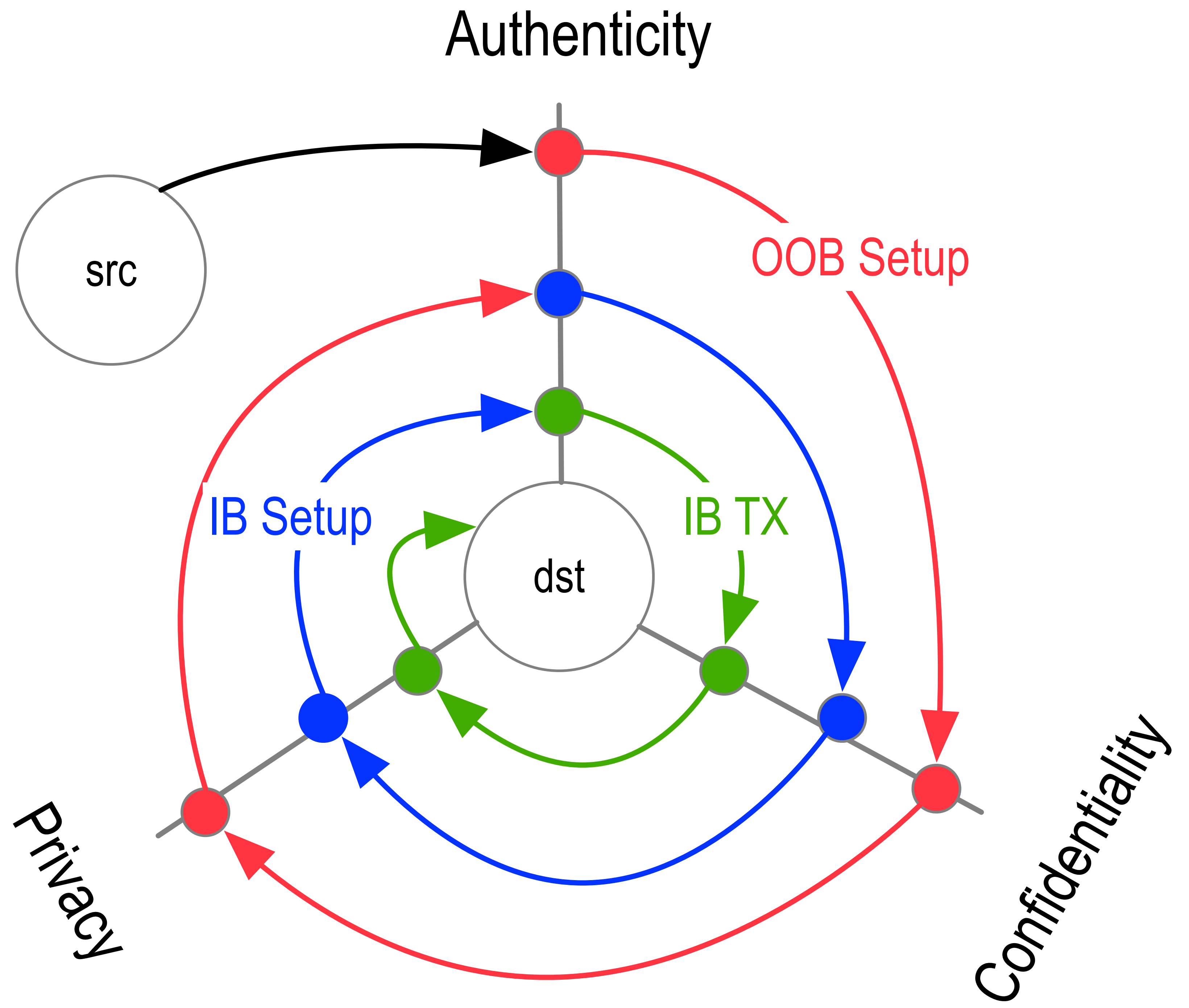
Naval Architecture Design Spiral



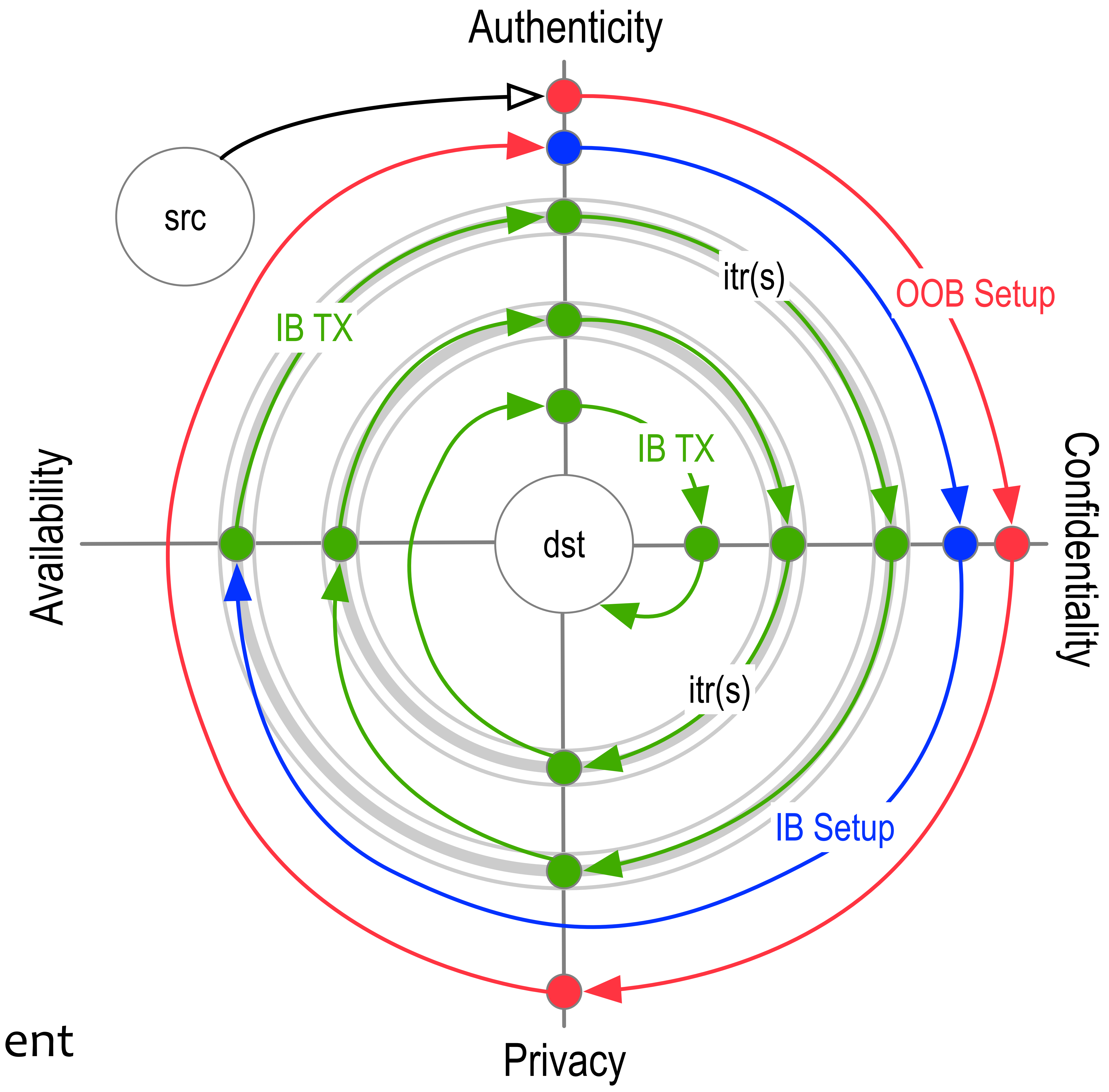
Layered Survivabilty Overlay Spiral

https://www.researchgate.net/publication/273026917_Ship_Design_and_System_Integration/figures?lo=1

# Setups Matter in Layered Security Overlays



Layered Security Overlay Spiral



Layered Security Overlay Spiral

Setups may be application dependent & hence overlay ordering dependent
Each setup requires one OOBA factor to protect against MITM attack
Setups may be costly, especially those with repeated OOBAs
Interactive setups are not as scalable as non-interactive setups
Trade-offs required

# Privacy

Authenticity and confidentiality are a cold war …

Proposed authenticity/confidentiality security overlays provide arbitrarily strong protection even with minimal resources

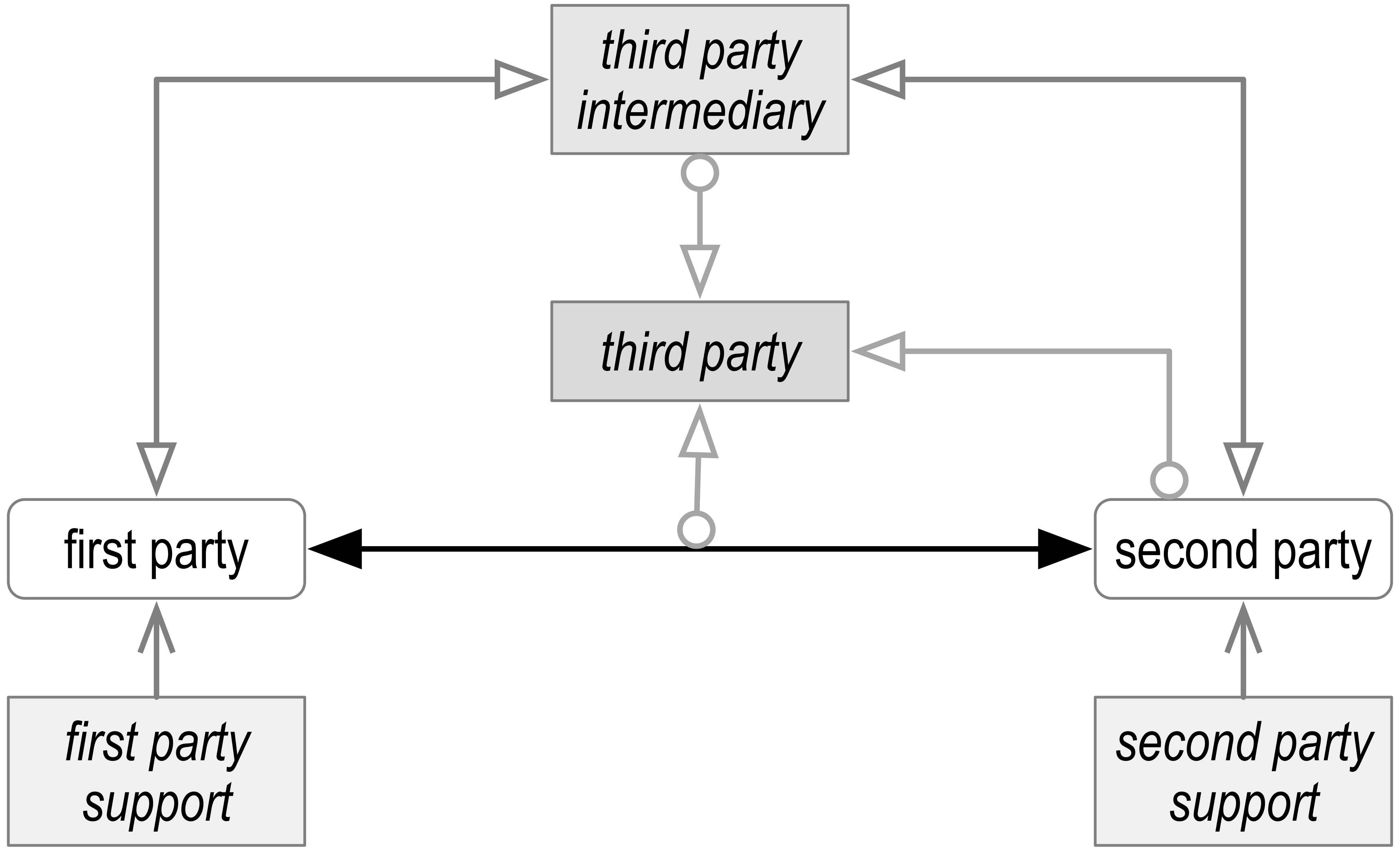
Privacy is a hot war …

Rapidly evolving tactics and tech, resource-constrained war of attrition against vastly superior opponents

Reduction in potential harm by decreasing privacy attack surface may net increase actual harm due to lost value capture opportunities

Focus on *exploitably correlatable identifiers*

Make risk mitigation trade-off between exploitable harm and protection cost



# Exploiters

State Actors (third party): (political incentives)

- Surveil, regulate, persecute all first parties, second parties and intermediaries with virtually unlimited resources and ability to cause harm  (out of scope for TSP for the time being)

Aggregators: (advertising incentive)

- Intermediaries (third party):
  - search as aggregator
  - cloud provider as aggregator
  - ISP as aggregator
- Second party as aggregator or colluding second parties as aggregators (platforms)
- Inadvertent first party as aggregator

Identity Thieves (third party): (asset theft incentives)

- access credentials
- ransomware)

# Third-Party Aggregator Threats and Mitigations

Monetized via advertising: (Assumes content data is confidentially encrypted between first party and second party)

Intermediaries as Aggregators:

Search as intermediary:

Mechanism:

First-party permissioned surveillance of metadata sent to second-party

Mitigations:

Use VPN with partitioned relationship identifier (random email)

Use a search engine that contractually protects your metadata

Cloud data storage provider as an intermediary

Mechanism:

First-party permissioned surveillance of first-party content data

Mitigations:

Use cloud storage providers that use end-to-end encryption (first-party confidential content data)

ISP as an intermediary:

Mechanism:

Non-content Metadata (routing and billing)

In USA, any anonymized data (both metadata & content) is aggregation permissible

Mobile device location tracking (no mitigation except to turn off mobile device)

Mitigations: (make non-content metadata confidential)

Use VPN (OVPN, better yet TSP VPN)

Provides herd privacy relative to other intermediaries

Contractual protection vis-a-vis the VPN itself

Use Privacy Protecting ISP

Contractual protection vis-a-vis the ISP itself

Use distributed decentralized confidential network overlay to make metadata harder to correlate;

onion or mix network routing;

TOR, DIDComm Routing, Elixxir, MainFrame

Use information-theoretic secure routing;

random walk routing (state actor resistant)

# Second- and First-Party Aggregator Threats and Mitigations

Monetized via advertising. (Assuming content data is confidentially encrypted between first party and second party)

Second Parties as Aggregators:

- Mechanism:
  - First-party content data (also non-content metadata when available), not as a third party surveillor (intermediary)
  - Statistical regression of anonymized content data by a large second party or by multiple colluding second parties
- Mitigation:
  - Don’t use second parties that aggregate without permission (anonymization is no protection)
  - Use contractually protected disclosure and/or selective disclosure

First Party as Inadvertent Aggregator:

- Mechanism:
  - Browser cookies makes first-party browser an inadvertent permissioned intermediary that can leak both non-content metadata and content data to second parties and/or intermediaries
- Mitigation:
  - Don’t use browsers in a way that supports tracking first-party data via cookies by second parties or intermediaries

# Identity Theft Threats and Mitigations

Monetized by stealing identities or ransomware: (Assumes content data is confidentially encrypted between first-party and second-party)

- Ultimate target is either treasure trove of credentialing information held by some second party that enables first party impersonation or high-value first party impersonation
- Does not monetize aggregated content data or metadata (anonymized or not)
  - "Hackers can use data stolen from companies with weak security to target employees and systems at other companies, including those with strong security protocols. ... the data ecosystem has become so vast and interconnected that people are only as safe as the least secure company that interacts with any company that has access to their data. That "least secure company" does not even need to have access to the consumer's data. … There were 5,212 confirmed breaches in 2021, which exposed 1.1 billion personal records across the globe" https://www.apple.com/newsroom/pdfs/The-Rising-Threat-to-Consumer-Data-in-the-Cloud.pdf
- A given trust domain is only as strong as the weakest connected trust domain (strong argument against non-autonomic trust bases)

Mechanism:

- Second-party impersonation attack (phishing, smishing) to steal first-party credentials
- Multi-step recursive authorization attacks
  - Edge attack on weak first-party then elevate to better first-party; VPN access of vendor, … , of vendor, of treasure trove holder
  - Leverage first-party attack on an intermediary which aids first-party attack on other second-party (DSN Hijack to verifiable certificate of second-party)
  - Leverage surveillance of first-party to second-party traffic via a compromised intermediary to mount attack on second-party to elevate first-party
  - Code supply chain (intermediary) attack to elevate to first-party credentials

Mitigation:

- Use strong authentication by any second-party of any first-party for any exploitable use of first-party credentials; CC, bank account, health data, ransom
- Get rid of treasure troves through cryptographic root-of-trust for authentication by any second-party of any first party
- Use strong authentication (zero-trust) at any party holding a treasure trove (also state actor resistant)
- Use strong authentication by second-party of any first party at any step along a multi-step recursive authorization attack

# Risk Mitigation Prioritization

Risk Rating and Ranking:

Magnitude of harm multiplied by the likelihood of harm versus the relative cost of mitigation as a percentage of available resources to mitigate all harms

Identity theft has very high harm magnitude with moderate likelihood but potentially low cost of mitigation via strong authentication and strong confidentiality

Aggregation harm per exploiter is small with very high likelihood so pick from privacy mitigations that are cost-effective

The internet is broken primarily because of identity theft.

The infrastructure changes needed to fix identity theft will enable easier adoption of the changes needed to better protect against aggregators.

ToIP TSP can take mantle of teaching and leading in best practices for securing trust over the internet

# Questions

# Definitions

*Authentic and Authenticity*:

The origin and content of any statement by a party to a conversation is provable to any other party.

… is about control over key state needed to prove who said what in the conversation (secure attribution)

*Confidential and Confidentiality*:

All statements in a conversation are only known by the parties to that conversation.

… is about control over the disclosure of what (content data) was said in the conversation and to whom it was said (partitioning).

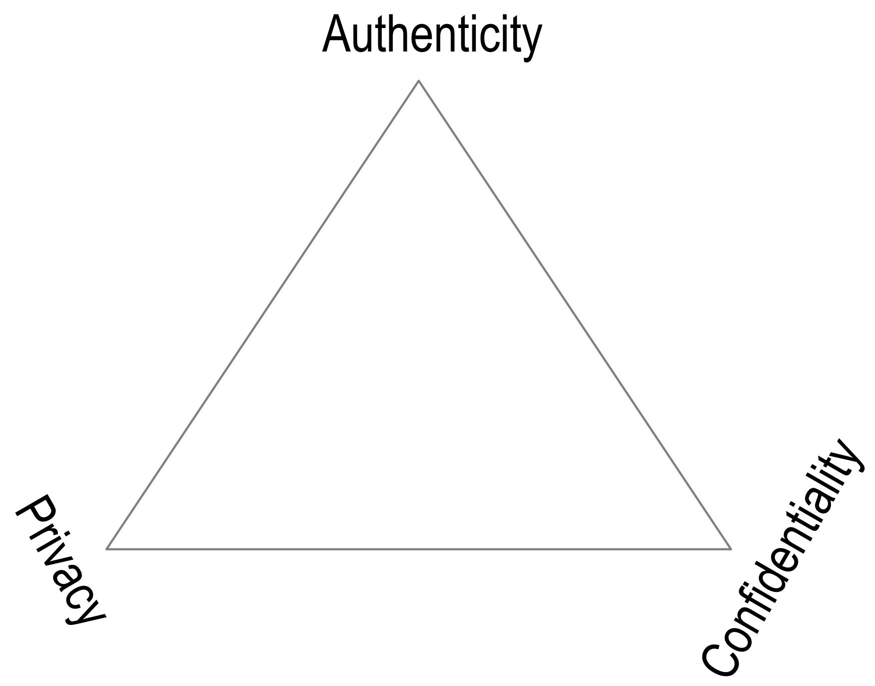
… is about control over key state needed to hide content via encryption vis-a-vis the intended parties

*Private and Privacy*:

The parties to a conversation are only known by the parties to that conversation.

… is about control over the disclosure of who participated in the conversation (non-content meta-data = identifiers)

… is about managing *exploitably correlatable identifiers*

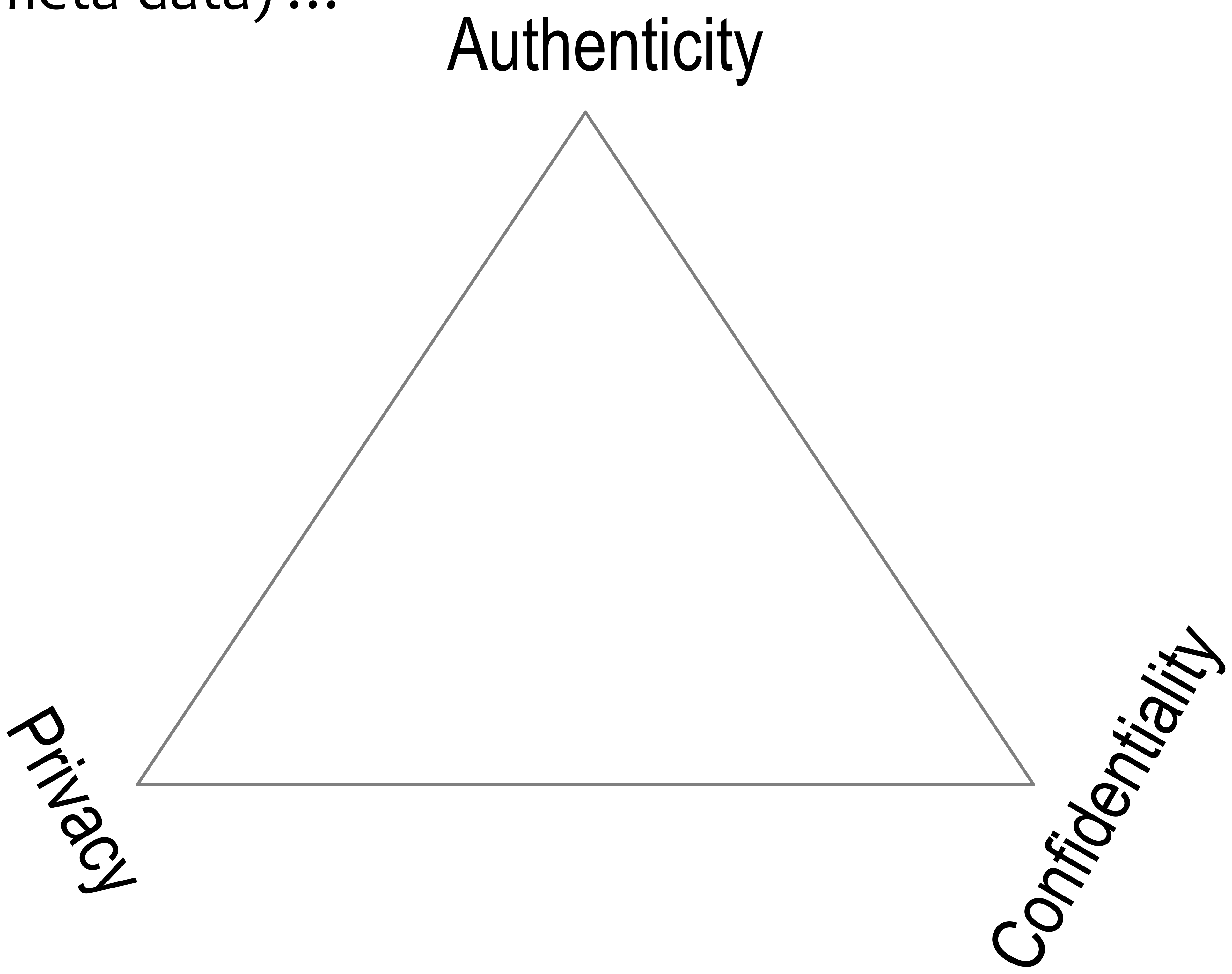


# Legal Relationship

Physical Search: (analogy to Constitutionally protected physical search)
Outside of home vs. Inside of home.
The *address* is a public identifier used for routing and taxing.
Viewing *who* enters the home from outside via public street is unprotected. (subpoena)
Viewing *what* happens behind closed doors inside the home is protected. (warrant)

Information Search:
Relatively weak legal protection (subpoena) for identifiers, (non-content meta-data) …
because identifiers are needed for public routing and billing.
Relatively strong legal protection (warrant) for content …
because the content is not needed for public routing and billing.

https://www.lawfareblog.com/relative-vs-absolute-approaches-contentmetadata-line
https://www.pogo.org/analysis/2019/06/the-history-and-future-of-mass-metadata-surveillance/



# Identifier System Security

Authentic transmission of data may be verified using an identity system security overlay.

Message authenticity is provided by verifying signatures to the authoritative keys pairs for the identifier included in the message.

This overlay maps identifiers to cryptographic key-pairs.

The security of the overlay is contingent on the security of the mapping.

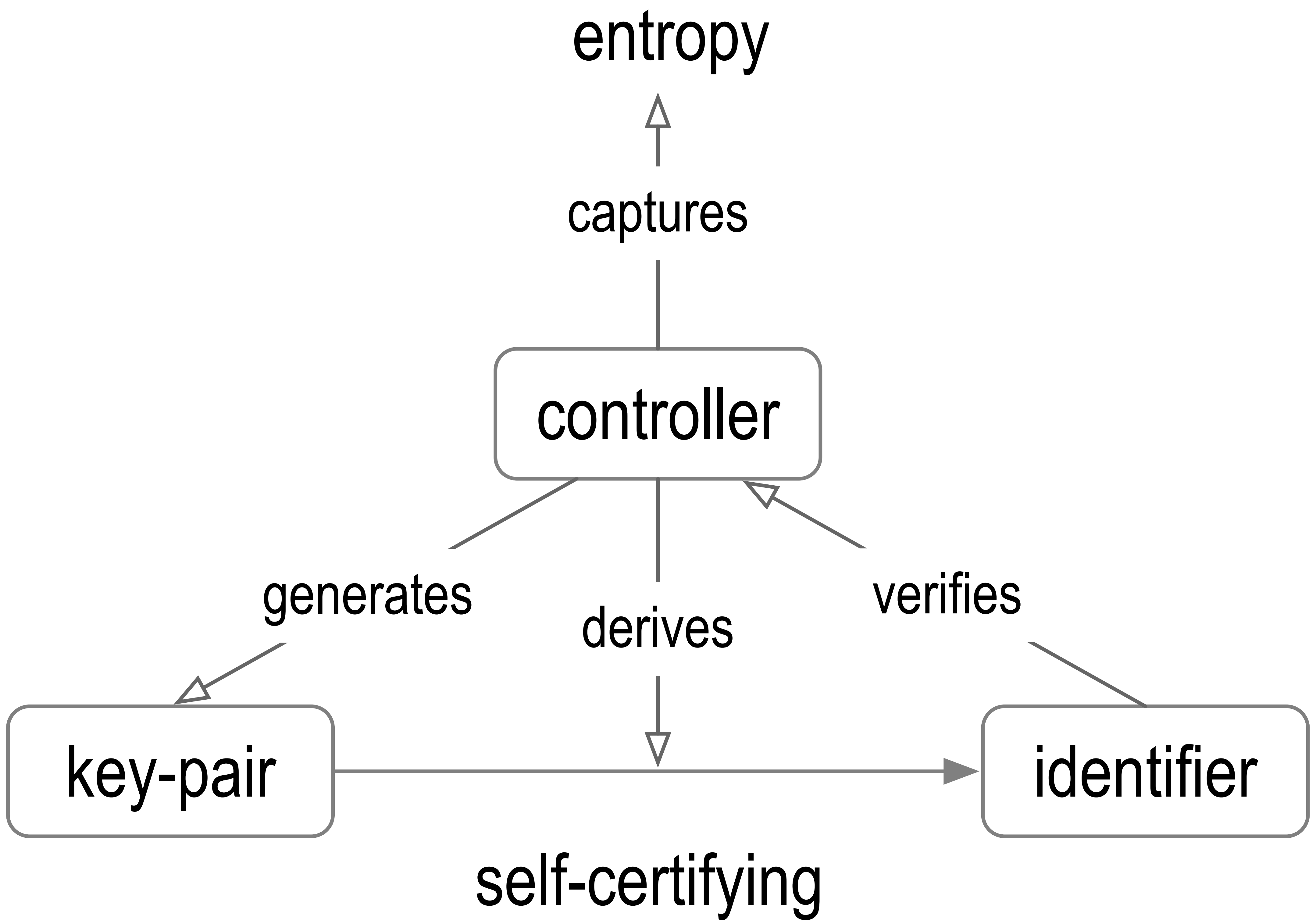
When those identifiers are self-certifying they are derived via cryptographic one-way functions from the key pairs.

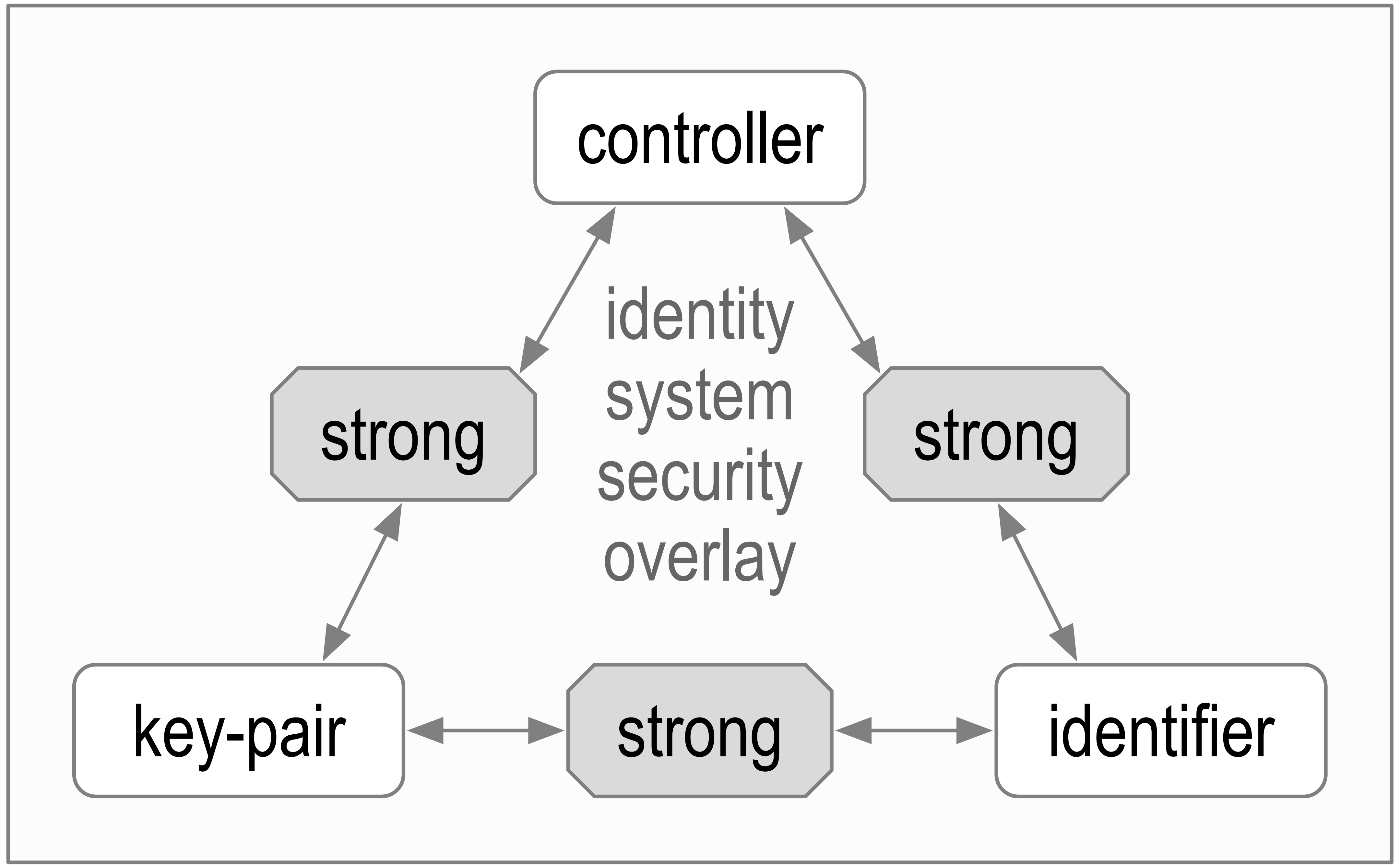
This strongly binds identifiers to key-pairs and makes control over the identifier equivalent to control over the key pairs.

This provides a self-certifying identifier with a cryptographic root-of-trust.

A key event log (KEL) provide support for secure key rotation (transferable control) without changing the identifier.

# Self-Certifying Identifier (SCID) (Ephemeral): Issuance and Binding



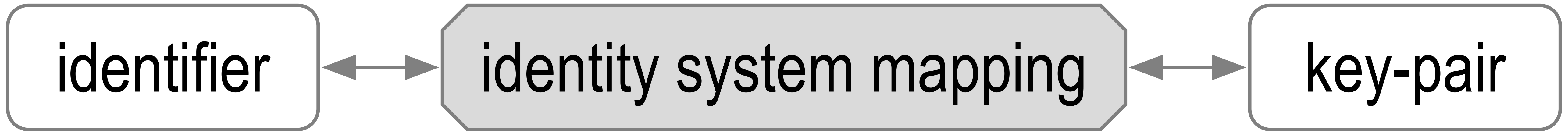


Self-Certifying Identifier Issuance Triad

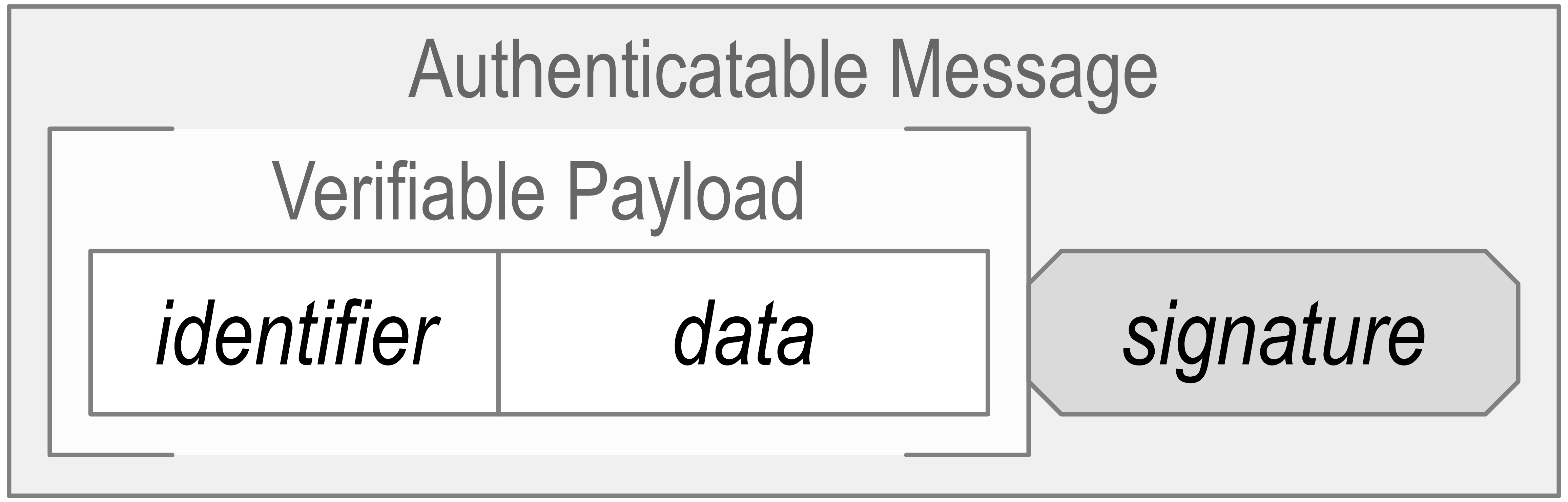
cryptographic root-of-trust

# Identity (-ifier) System Security Overlay

ephemeral (non-transferable) mapping = identifier is encoded public key



Establish authenticity of IP packet's message payload.



# PKI Then and Now

Who uses a password manager?

Who uses an authenticator app?

Who uses password-less login?

Then: Managing private keys impossible for users, federated identity.

Now: Mobile Devices with MFA & secure boot, password-less login.

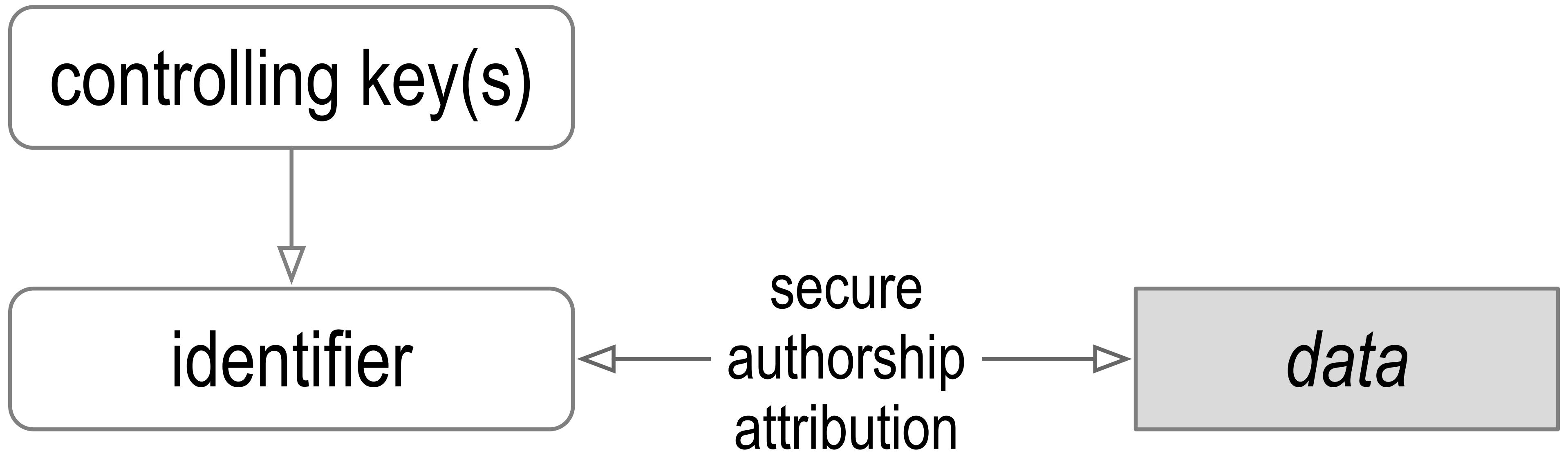
Then: Weak Crypto

Now: Strong crypto: ECC signing & asymmetric encryption.

Then: Perimeter security, no persistence of control over identifiers.

Now: authentic web and zero-trust architecture for identity.

# Flaw of PKI (DNS/CA)



Use of private keys exposes them to side-channel attack.

Over-time, exposure makes private keys weak.

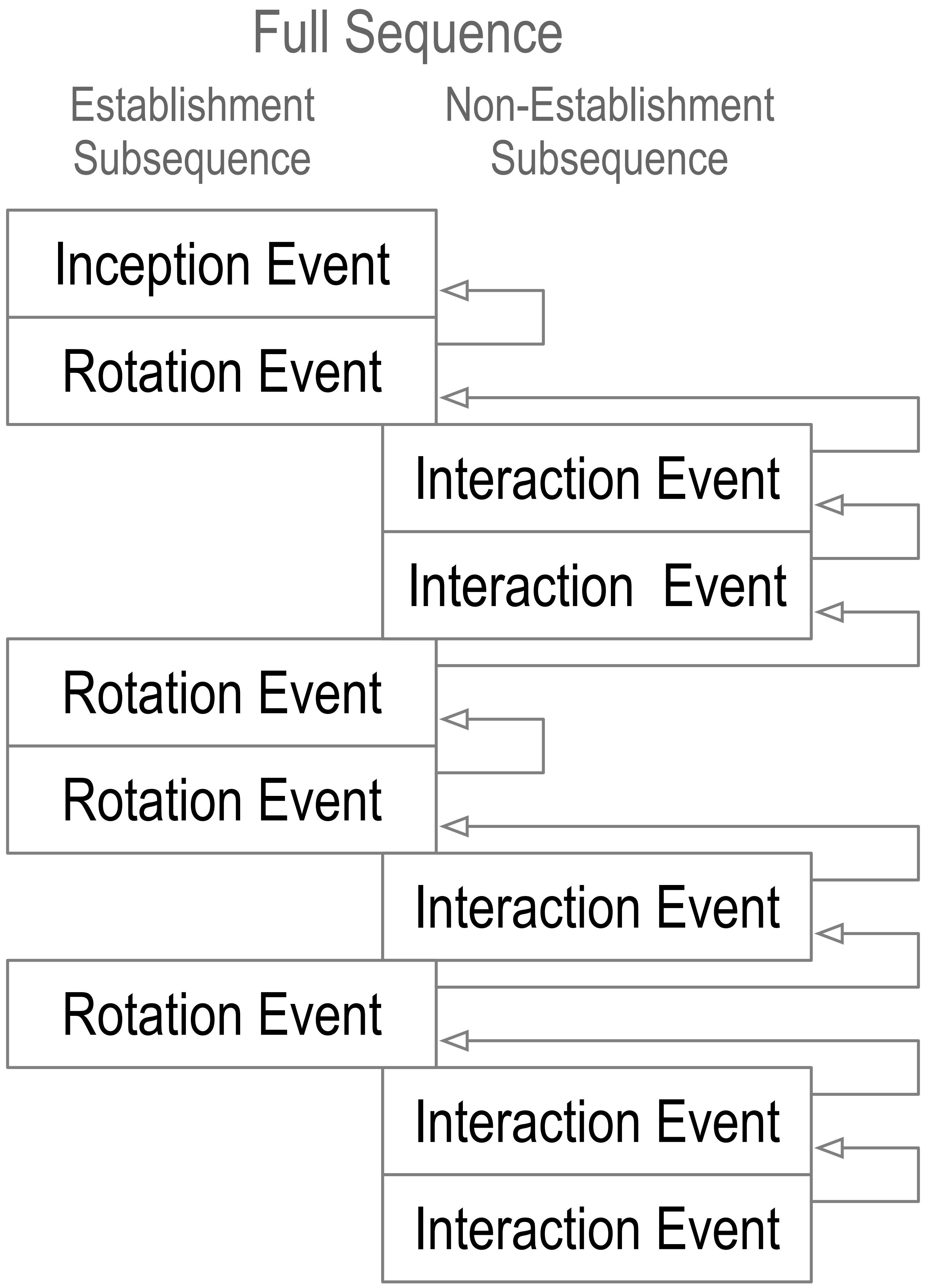
Thus, from time-to-time one must revoke and replace the controlling private keys for a given identifier

Hence key rotation

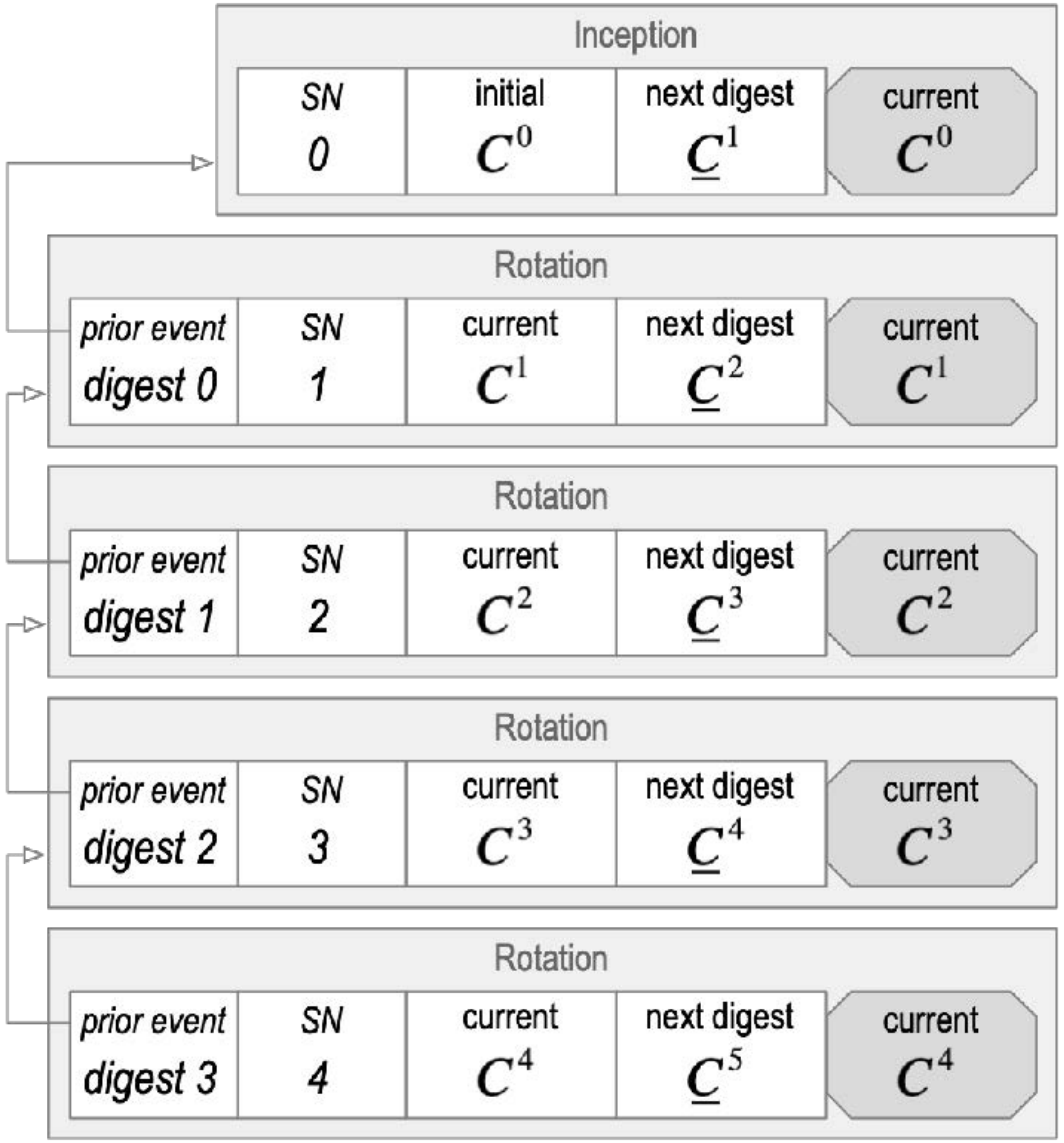
Existing PKI must re-establish the root-of-trust with each rotation thereby making it vulnerable to attack

Breaks the chain-of-trust-of-control over the identifier

# Solution: Key Pre-Rotation



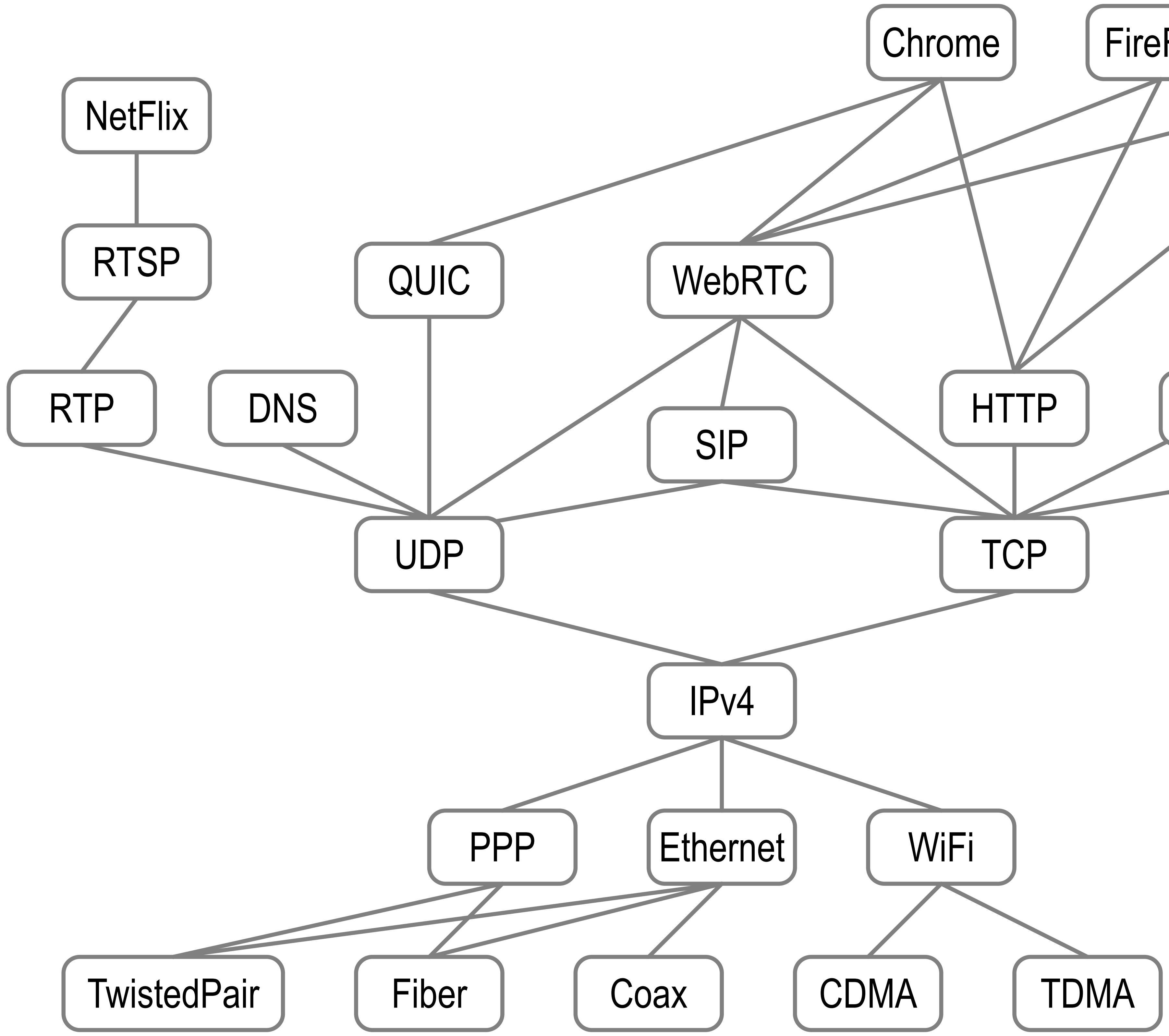
*duplicity evident*
*verifiable data structure*

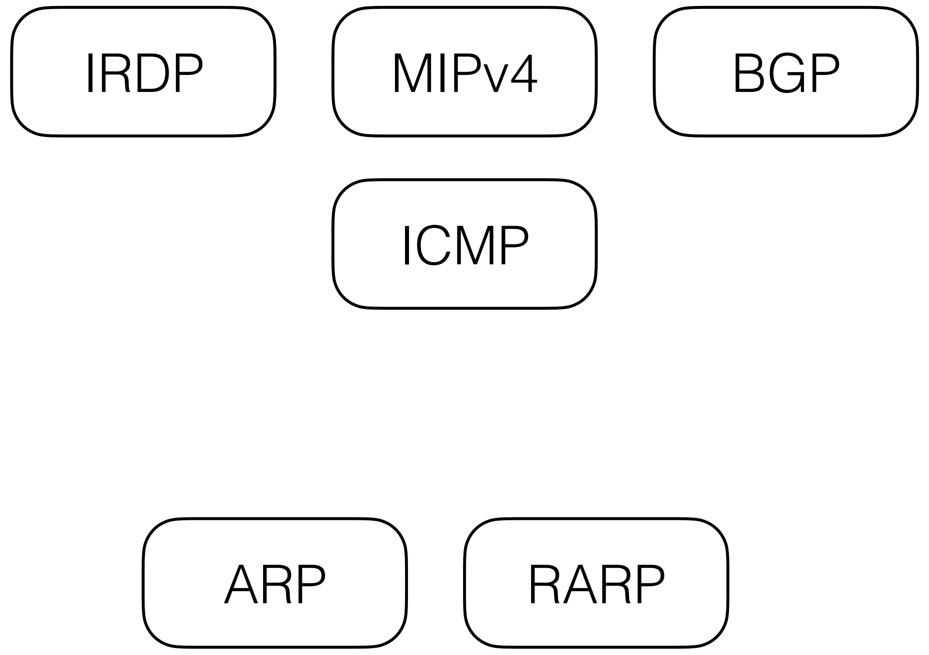


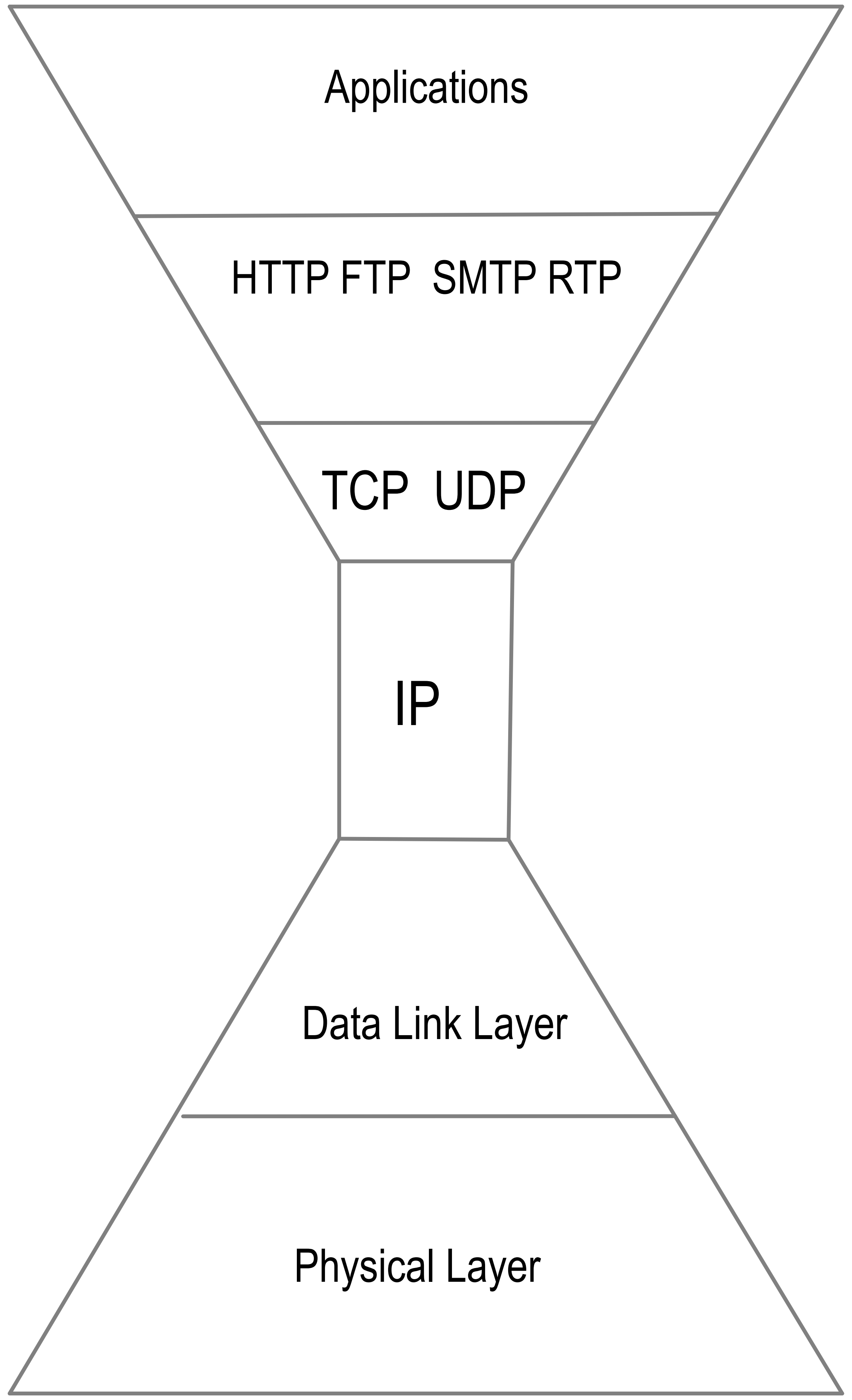
Digest of *next* key(s) makes pre-rotation post-quantum secure

D.J. Bernstein: https://cr.yp.to/hash/collisioncost-20090517.pdf

# Spanning Layer







https://web.archive.org/web/20050415042854/http://www.csd.uch.gr/~hy490-05/lectures/Clark_interoperation.htm

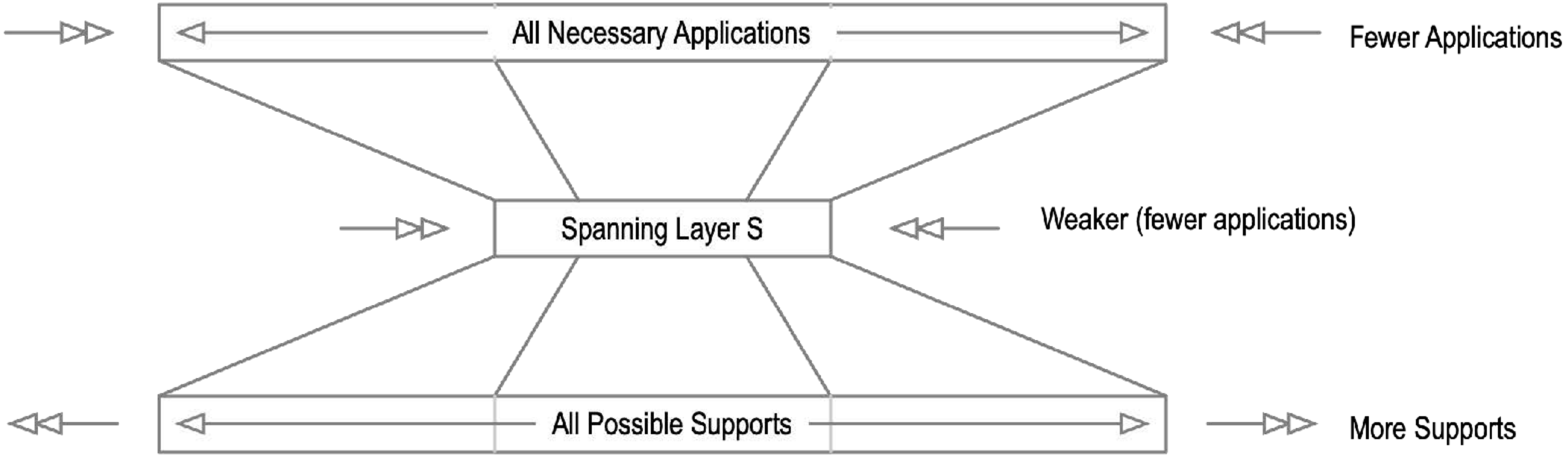
# Logical Weakness Property

A weaker spanning layer (less functionality) has fewer supported applications but more possible supports than a stronger spanning layer

Weaker means less functionality.

A stronger spanning layer (more functionality) has fewer possible supports but more supported applications

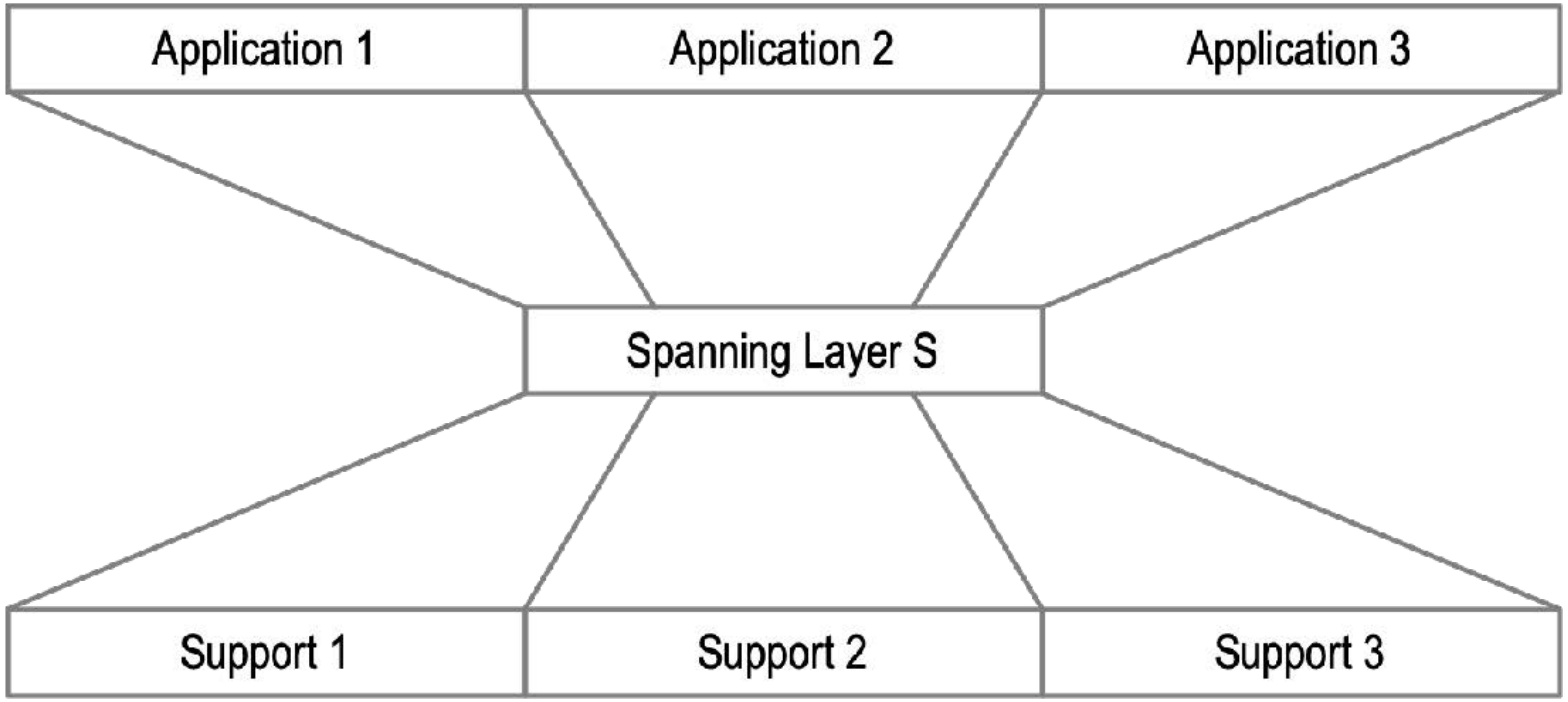
A minimally sufficient spanning layer is the weakest  possible layer that still supports the necessary applications



https://cacm.acm.org/magazines/2019/7/237714-on-the-hourglass-model/fulltext

# Features of a Spanning Layer

**Simplicity:** Only one way to access (orthogonality) any service or resource through the spanning layer
**Generality:** The spanning layer has the richest set of possible applications without increasing its logical strength
**Resource Limiting:** The spanning layer abstracts and limits resource access by supported applications
**Deployment Scalability:** widespread adoption though minimally essential functionality



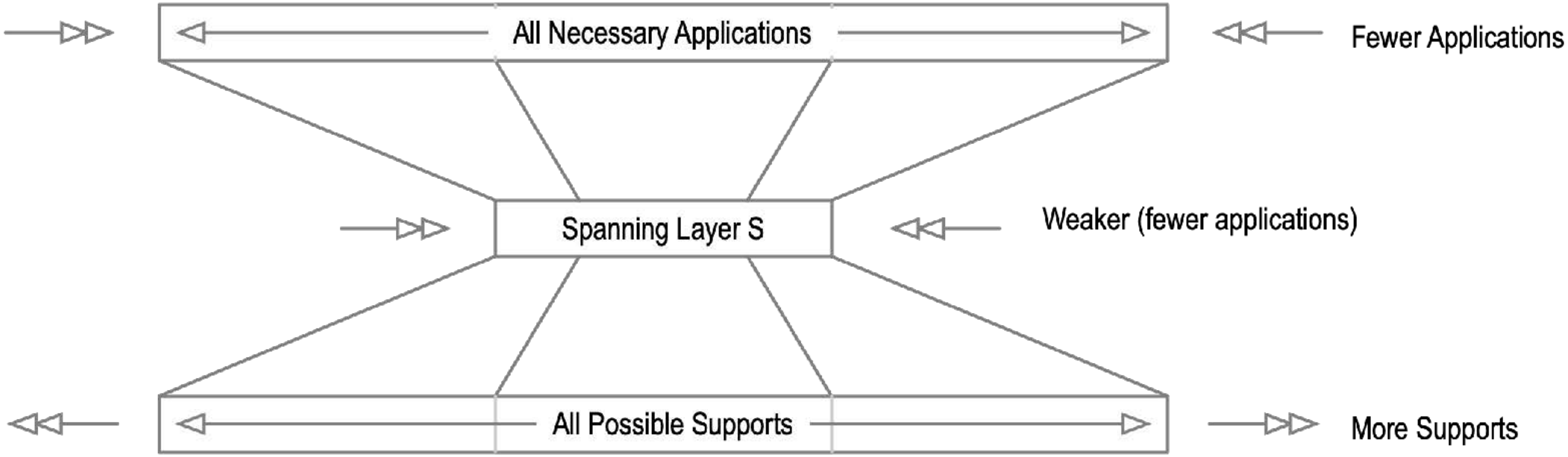
https://cacm.acm.org/magazines/2019/7/237714-on-the-hourglass-model/fulltext

# Implications of the Hourglass Theorem

Deployment scalability is enabled by a spanning layer that has implementations using as many different supports as possible

Broader adoption comes from increasing the support

A minimally sufficient spanning layer may not be defensibly monetizable because monetization disincentivizes broader support



# Insights

Adoption of a common service interface is a key to interoperability, portability, and “future proofing” in the face of rapid technological change.

The design of both the internet and Unix followed the hourglass principle leading to dominance while still allowing disruptive innovation.

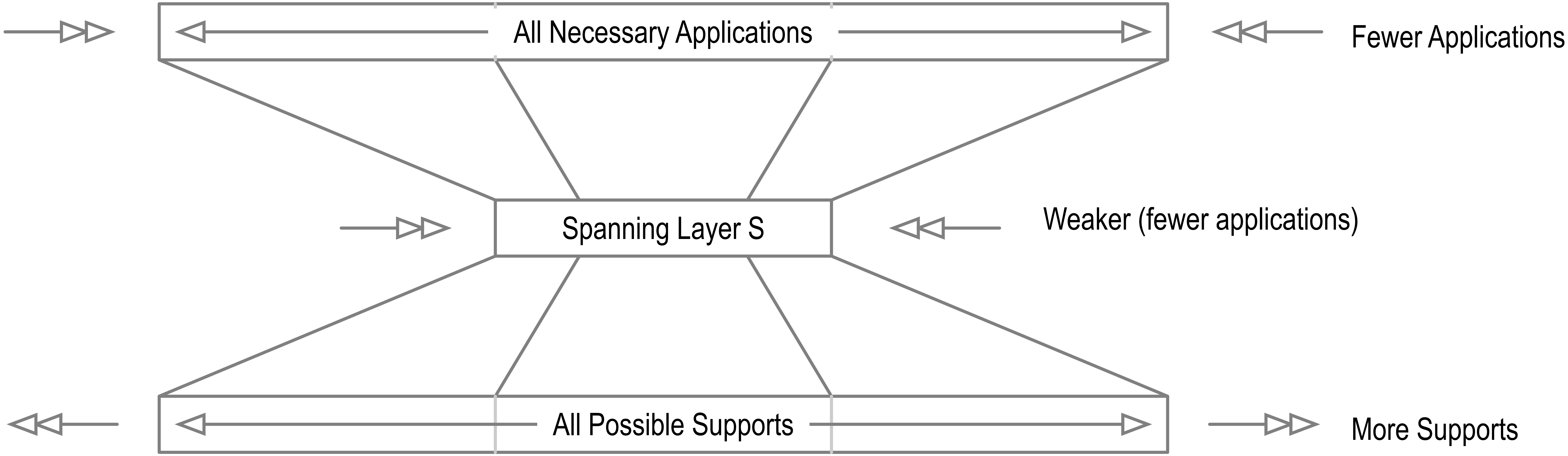
Successful interface design adheres to the discipline of simplicity, generality, and resource limitation of the common interface

The deployment scalability trade-off is that broader adoption comes from thinning or weakening the spanning layer

KERI design guidance from hourglass principle:  address only = namespace agnostic, portable, primary root-of-trust is portable verifiable data structure, secondary roots-0f-trust are replaceable

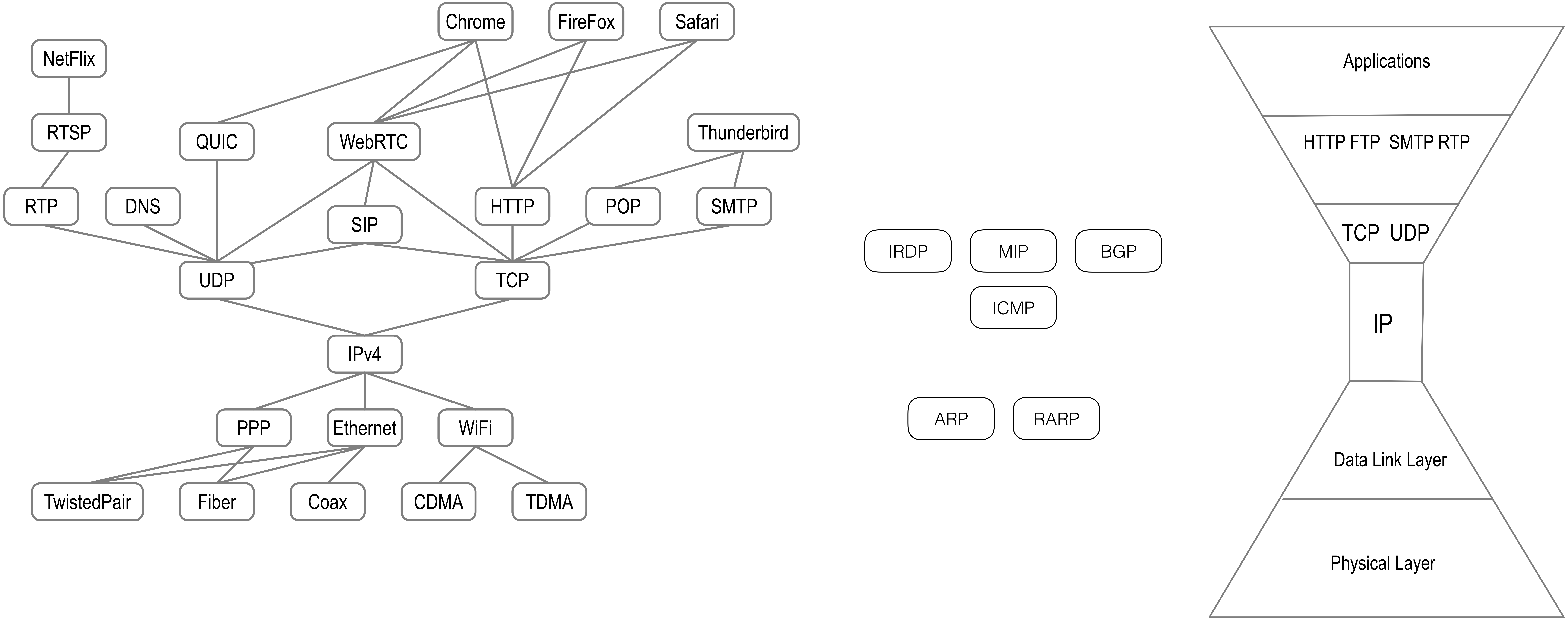
# Hourglass Theorem Summary

https://cacm.acm.org/magazines/2019/7/237714-on-the-hourglass-model/fulltext



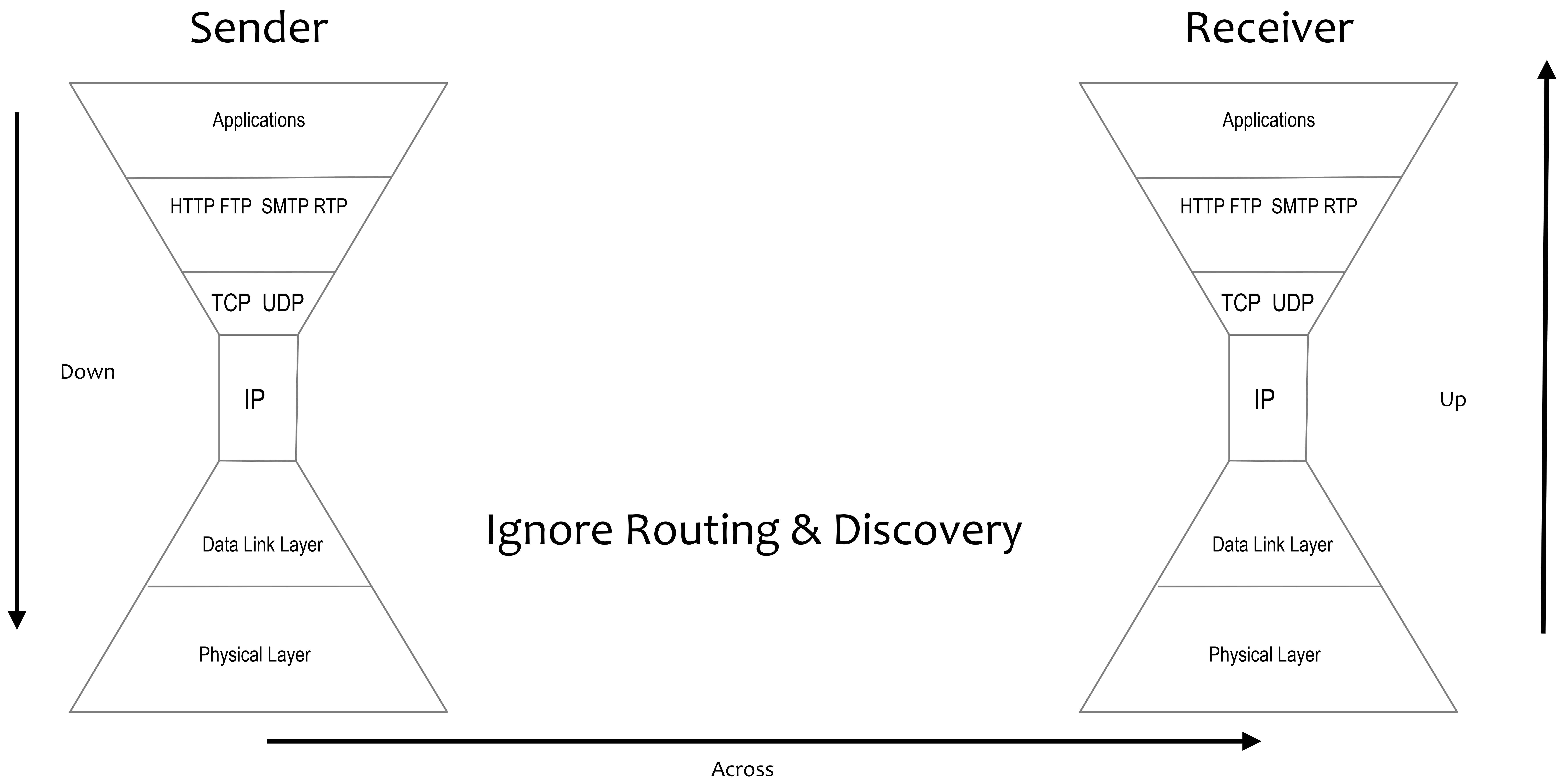
Minimally sufficient spanning layer singularly spans supporting protocols below it and applying protocols above it.
The degree of dominance of a spanning layer is eco-system validation of the preferred design

# Spanning Layer

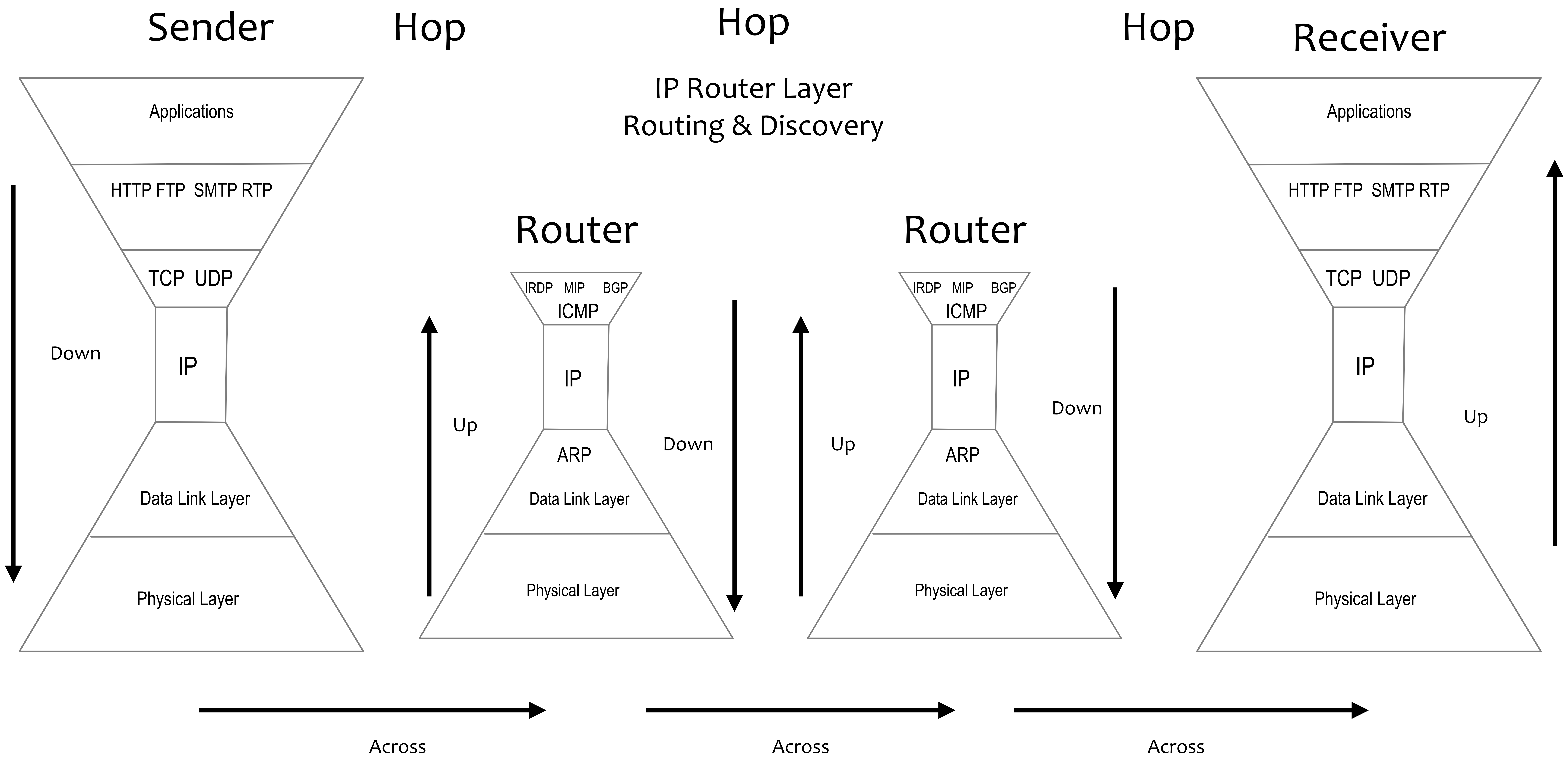


https://web.archive.org/web/20050415042854/http://www.csd.uch.gr/~hy490-05/lectures/Clark_interoperation.htm

# Application Layer Endpoint Spanning



# IP Router Layer Intermediation



Sender

Hop

Hop

Hop

Receiver

IP Router Layer
Routing & Discovery

Applications

HTTP FTP SMTP RTP

TCP UDP

IP

Data Link Layer

Physical Layer

Down

Router

IRDP MIP BGP

ICMP

IP

ARP

Data Link Layer

Physical Layer

Up

Down

Router

IRDP MIP BGP

ICMP

IP

ARP

Data Link Layer

Physical Layer

Up

Down

Applications

HTTP FTP SMTP RTP

TCP UDP

IP

Data Link Layer

Physical Layer

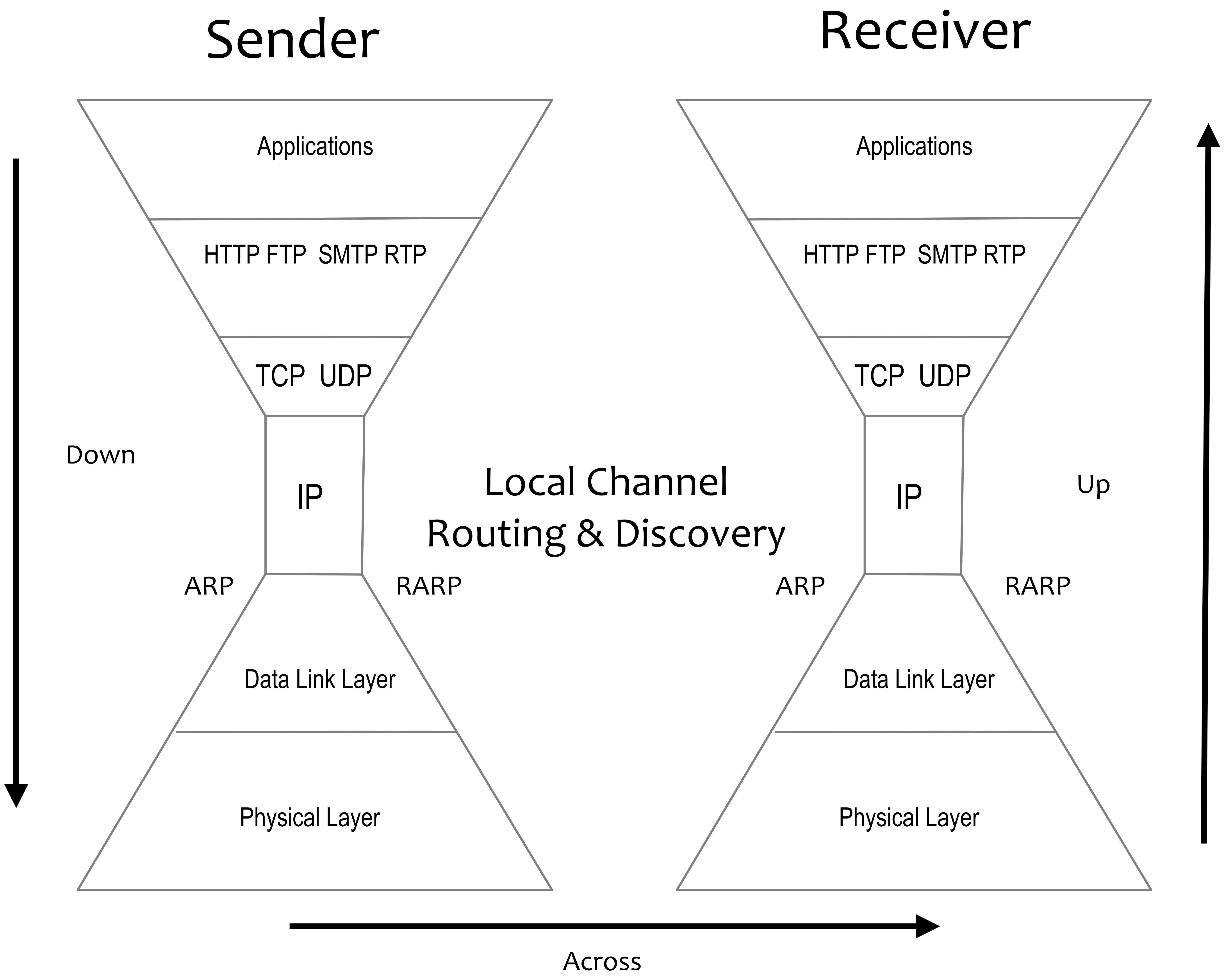
Up

Across

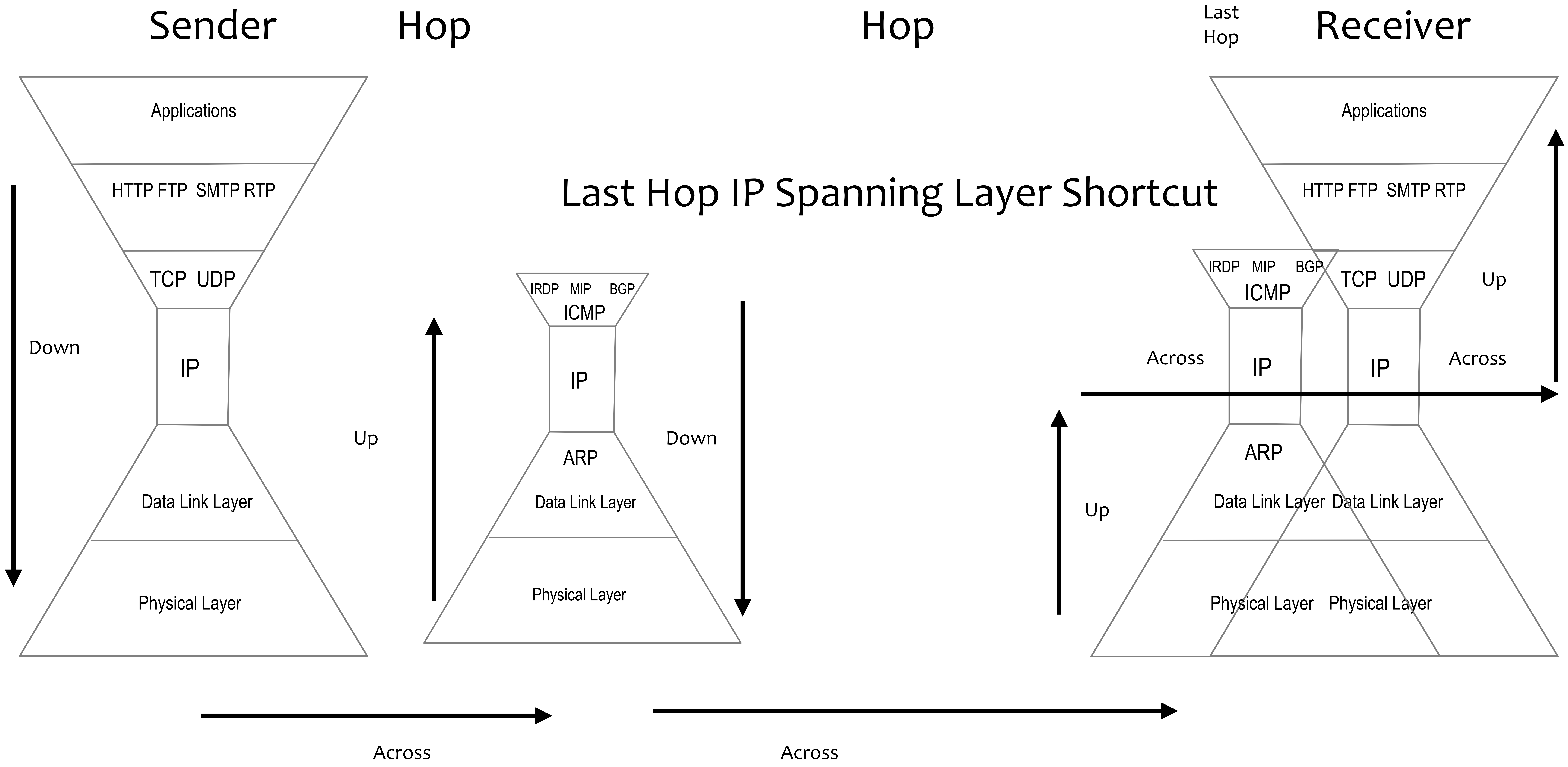
Across

Across

# Same Local Channel Routing Shortcut

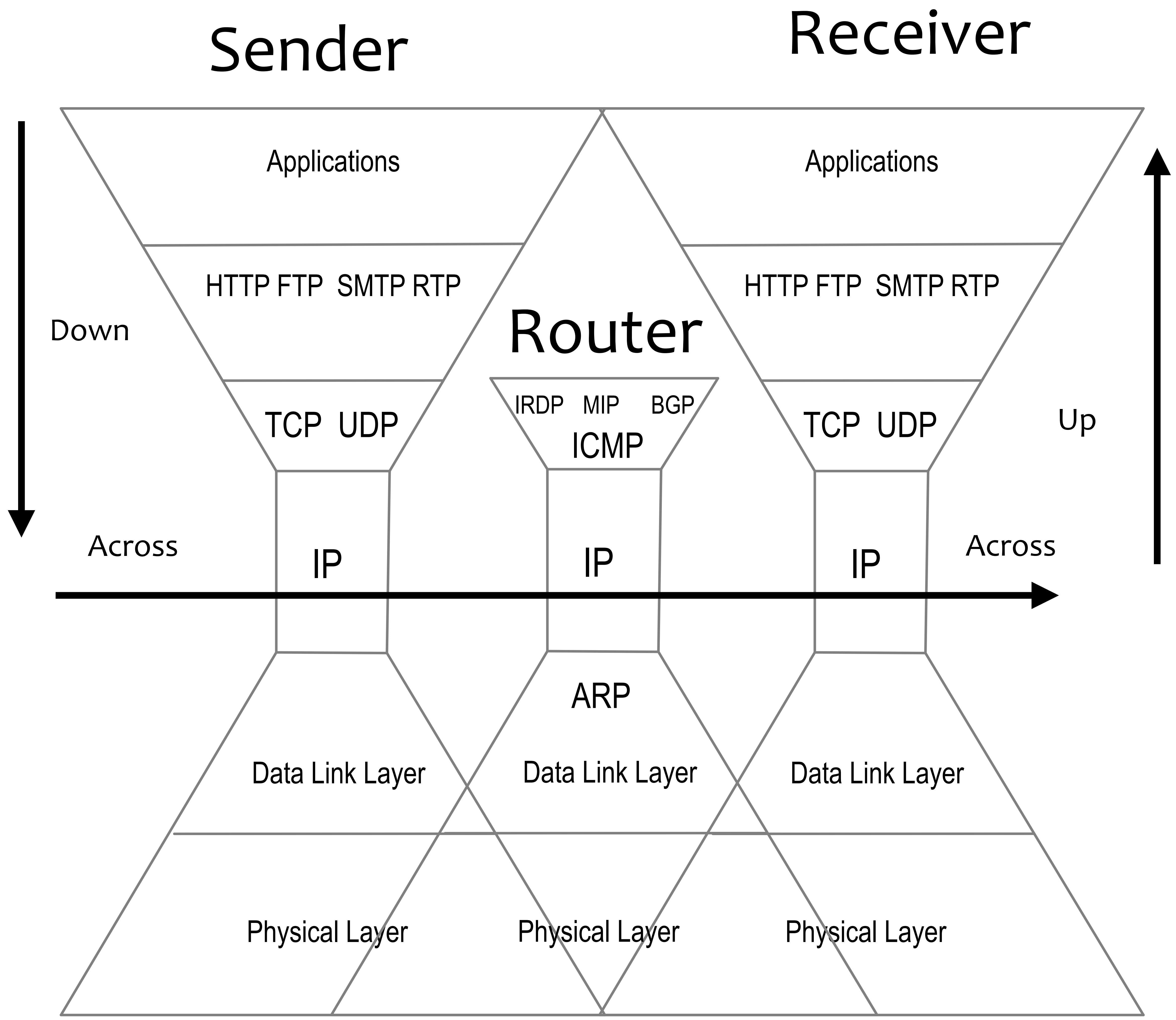


# Last Hop Routing Shortcut via Spanning Layer



# Local Hop Routing Shortcut via Spanning Layer

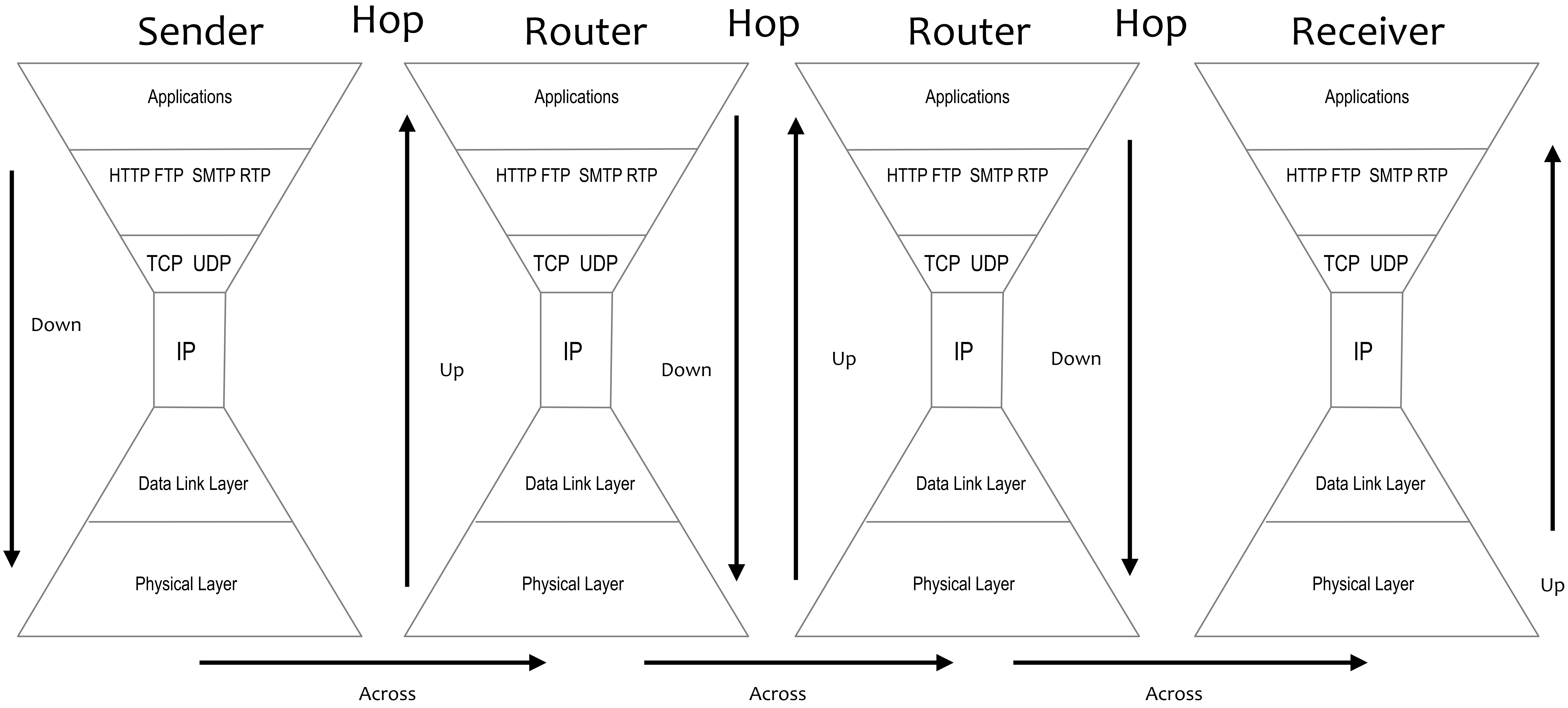
## Local Hop IP Spanning Layer Shortcut



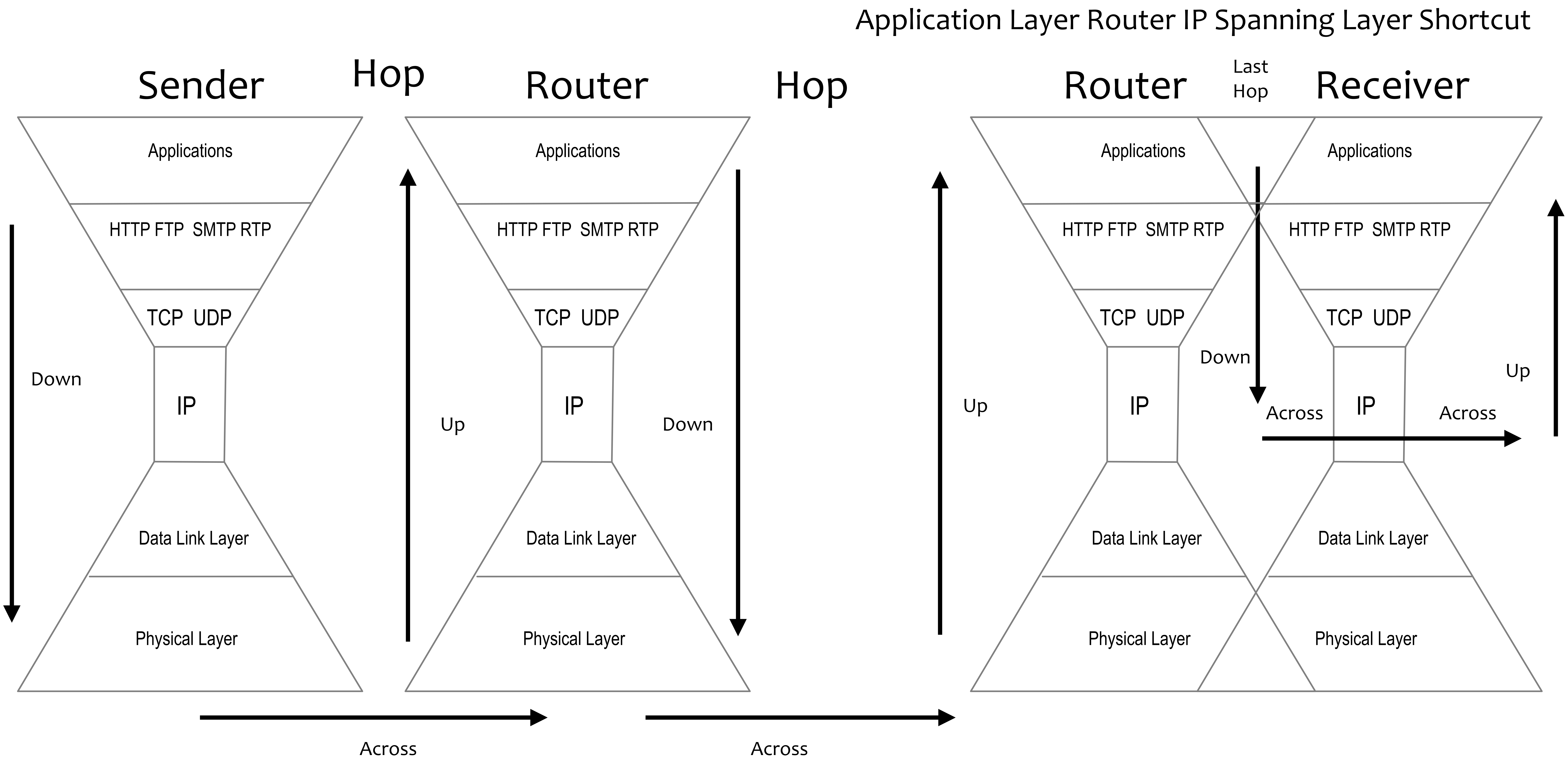
# Application Layer Router Intermediation

Application Layer Router

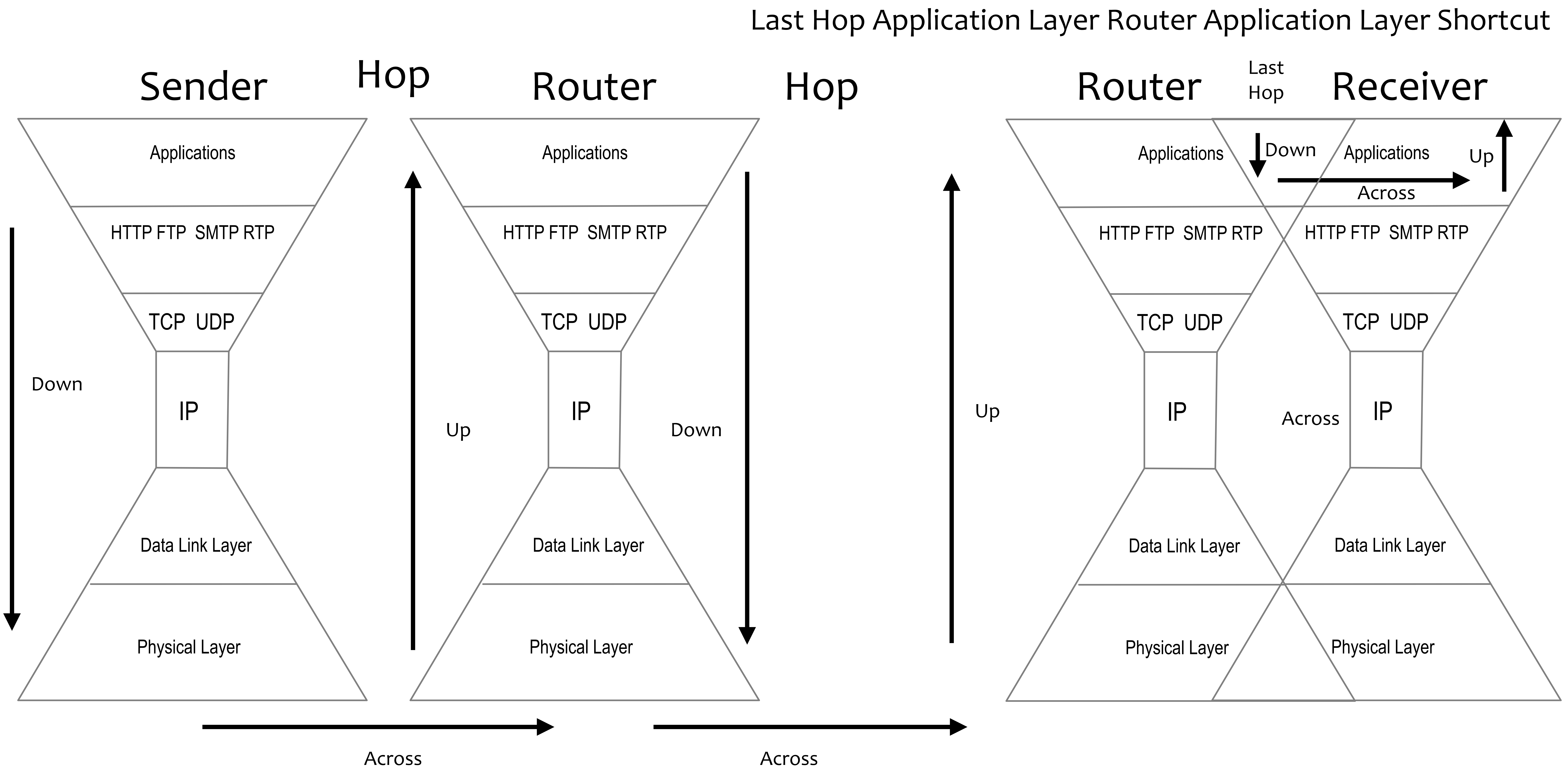
Routing & Discovery



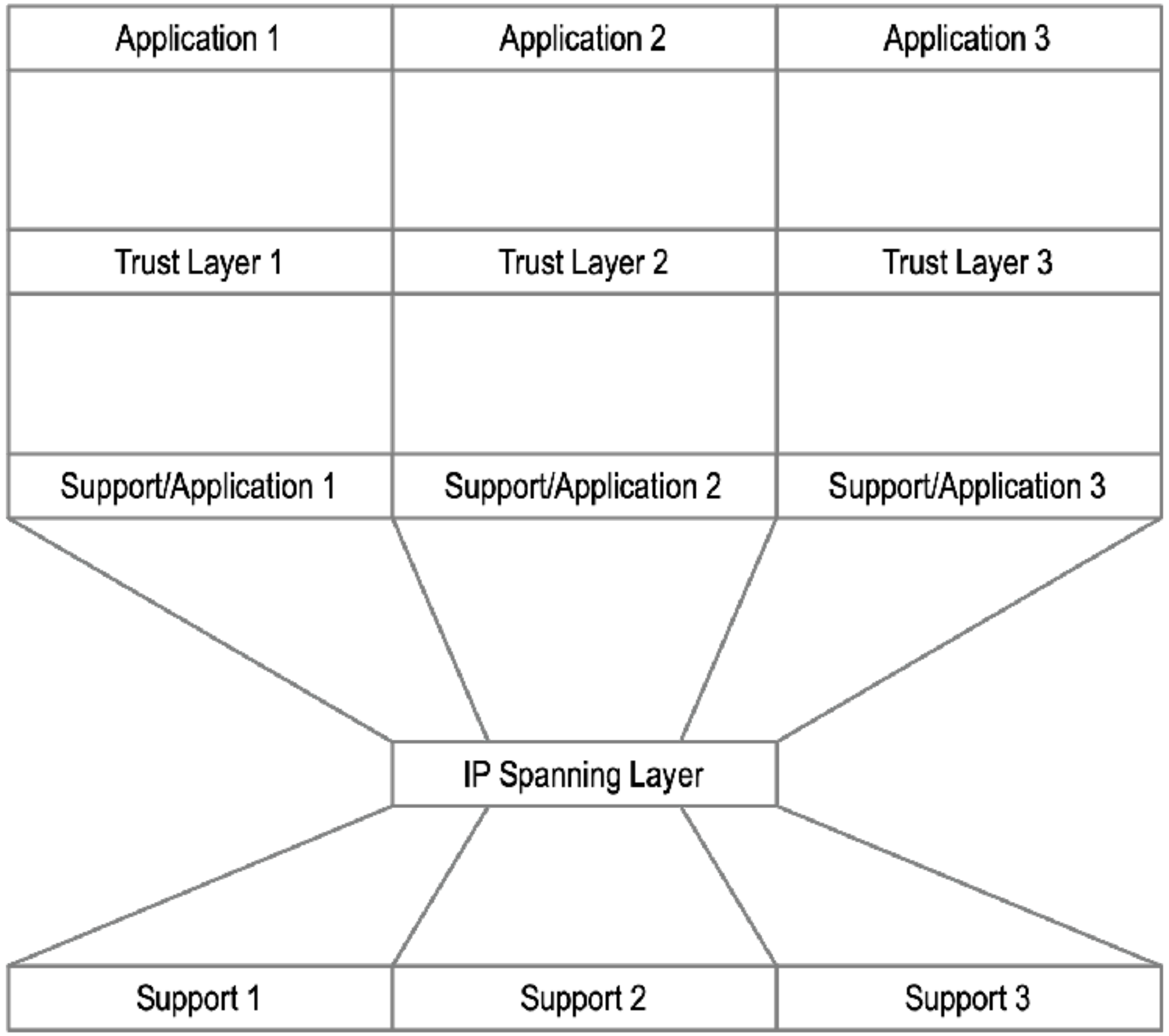
# Last Hop Application Layer Routing Spanning Layer Shortcut



# Last Hop Application Layer Routing Application Layer Shortcut

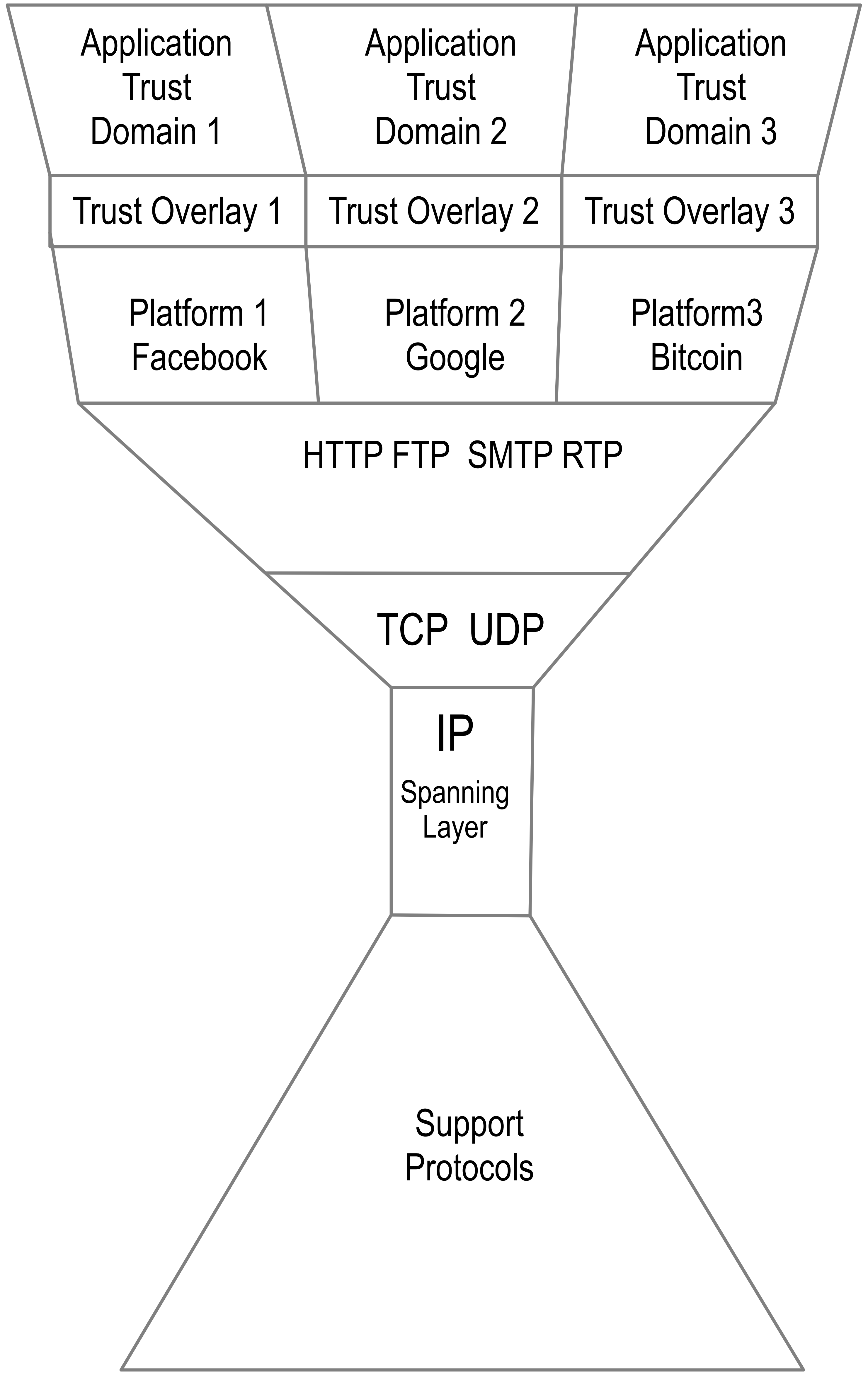


# Platform Locked Trust



Application 1
Application 2
Application 3
Trust Layer 1
Trust Layer 2
Trust Layer 3
Support/Application 1
Support/Application 2
Support/Application 3
IP Spanning Layer
Support 1
Support 2
Support 3

## Trust Domain Based Segmentation



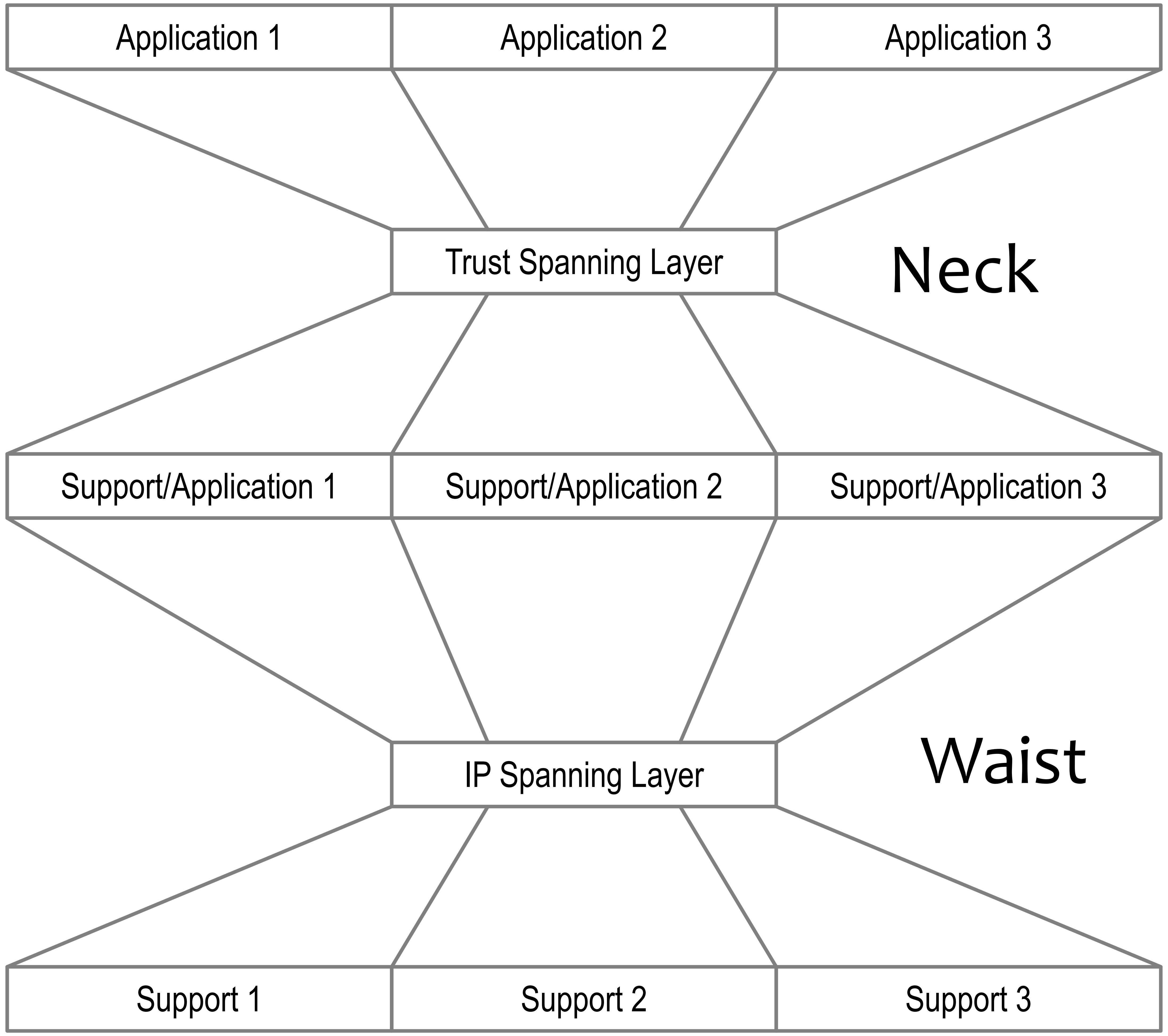
Application Trust Domain 1
Application Trust Domain 2
Application Trust Domain 3
Trust Overlay 1
Trust Overlay 2
Trust Overlay 3
Platform 1 Facebook
Platform 2 Google
Platform3 Bitcoin
HTTP FTP SMTP RTP
TCP UDP
IP
Spanning Layer
Support Protocols

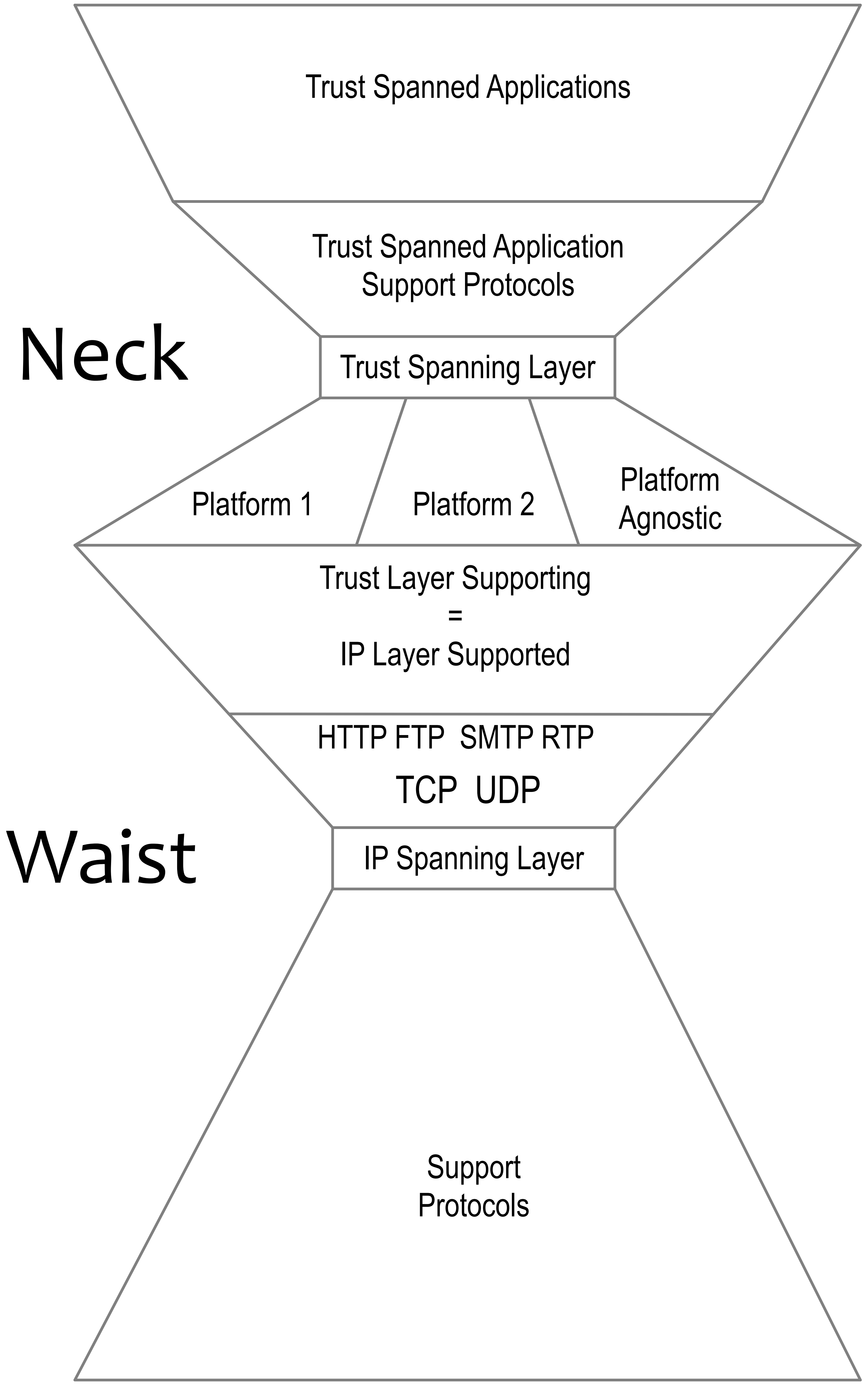
Each trust layer only spans platform specific applications

Bifurcated internet trust map

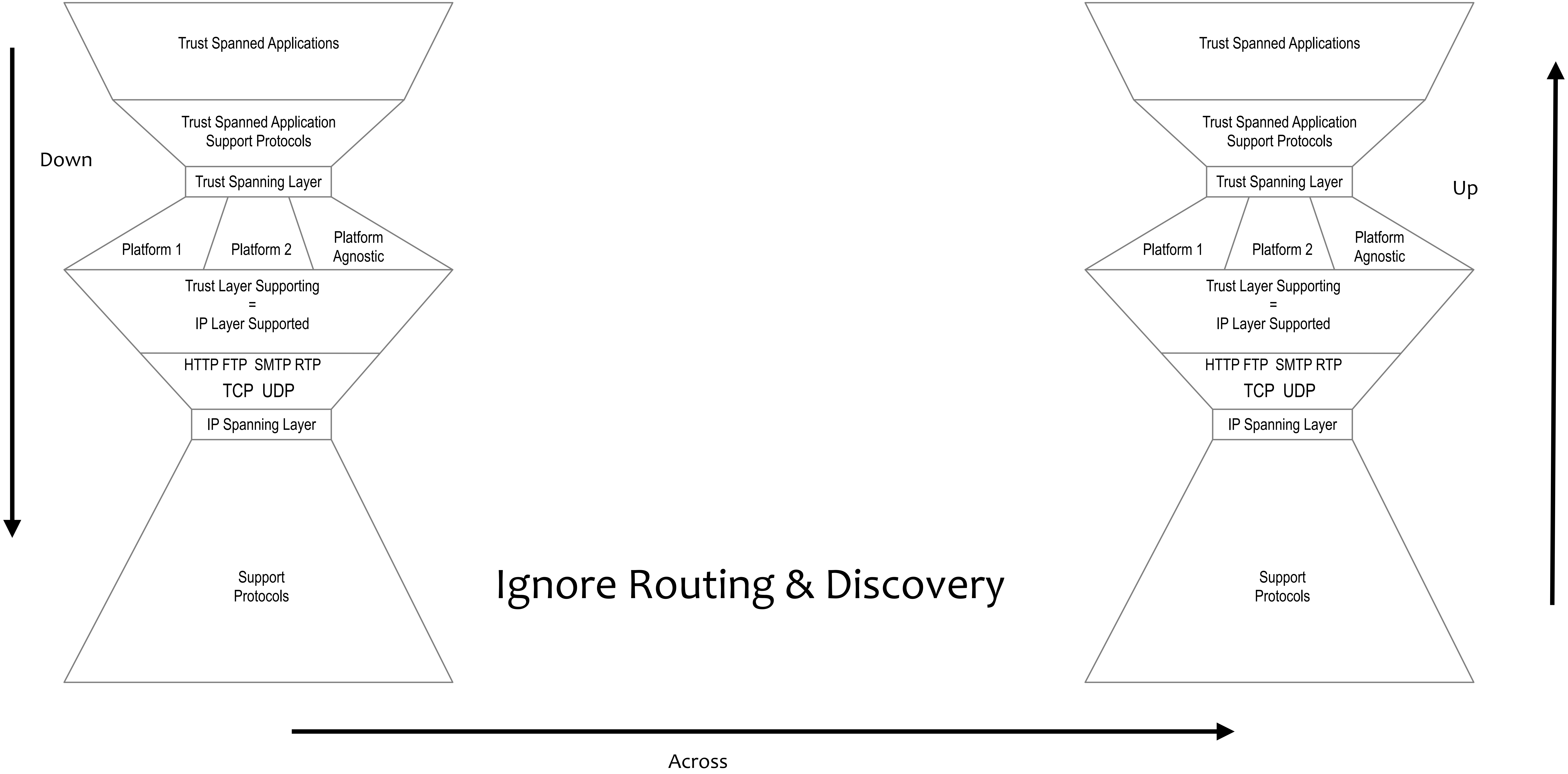
No *spanning* trust layer

# Solution: Trust Neck and IP Waist





# Trust Application Layer Endpoint Spanning


Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Down
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Up
Ignore Routing & Discovery
Across

# Untrusted Layer Router Intermediation

Sender
Hop
Hop
Hop
Receiver
KERI/ACDC Approach OOBI (IP)
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Down
Router
IRDP MIP BGP
ICMP
IP
ARP
Data Link Layer
Physical Layer
Up
Down
Router
IRDP MIP BGP
ICMP
IP
ARP
Data Link Layer
Physical Layer
Up
Down
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Up
Across
Across
Across

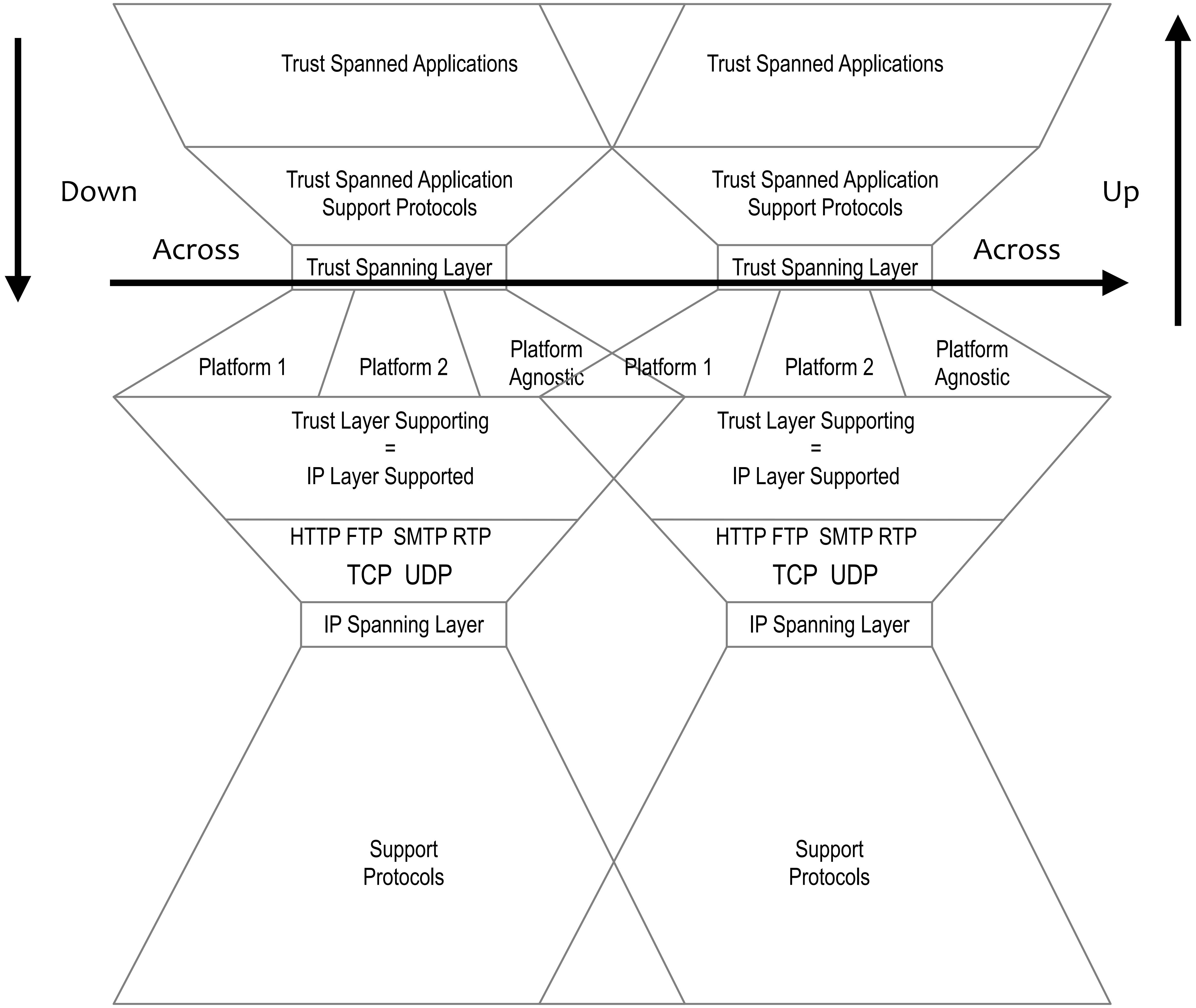
# Last Hop Shortcut via Untrused Layer

Sender
Hop
Hop
Hop
Receiver
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Down
IP Router Layer
Routing & Discovery
Router
Router
IRDP MIP BGP
ICMP
IP
ARP
Data Link Layer
Physical Layer
Up
Down
Up
Down
Across
Across
Up
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
IRDP MIP BGP
ICMP
IP
ARP
Data Link Layer
Physical Layer
Across
Across

# Untrusted Layer Router Intermediation

Sender
Hop
Router
Hop
Router
Hop
Receiver
KERI/ACDC Approach .well-known or web search engine
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Down
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Applications
HTTP FTP SMTP RTP
TCP UDP
IP
Up
Down
Data Link Layer
Physical Layer
Applications
HTTP FTP SMTP RTP
TCP UDP
IP
Up
Down
Data Link Layer
Physical Layer
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Up
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
IP Router Layer
Routing & Discovery
Across
Across
Across

# Shared Layer Routing Shortcut



# Trusted Layer Router Intermediation

Sender

Trust Spanned Applications

Trust Spanned Application Support Protocols

Trust Spanning Layer

Platform 1

Platform 2

Platform Agnostic

Trust Layer Supporting = IP Layer Supported

HTTP FTP SMTP RTP

TCP UDP

IP Spanning Layer

Support Protocols

Down

Hop

Up

Router

Trust Layer Routing & Discovery Protocols

Trust Spanning Layer

IP to Trust Routing & Discovery Protocols

Trust Layer Supporting = IP Layer Supported

IP Spanning Layer

Support Protocols

Down

Across

Hop

Trust Layer Router Routing & Discovery

Up

Router

Trust Layer Routing & Discovery Protocols

Trust Spanning Layer

IP to Trust Routing & Discovery Protocols

Trust Layer Supporting = IP Layer Supported

IP Spanning Layer

Support Protocols

Down

Across

Hop

Receiver

Trust Spanned Applications

Trust Spanned Application Support Protocols

Trust Spanning Layer

Platform 1

Platform 2

Platform Agnostic

Trust Layer Supporting = IP Layer Supported

HTTP FTP SMTP RTP

TCP UDP

IP Spanning Layer

Support Protocols

Up

Across

# Last Hop Shortcut Trusted Layer Router

Sender
Hop
Hop
Last Hop
Receiver
Router
Trust Layer Router
Routing & Discovery
Router
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Down
Up
Trust Layer Routing & Discovery Protocols
Trust Spanning Layer
IP to Trust Routing & Discovery Protocols
Trust Layer Supporting = IP Layer Supported
IP Spanning Layer
Support Protocols
Down
Up
Across
Trust Layer Routing & Discovery Protocols
Trust Spanning Layer
IP to Trust Routing & Discovery Protocols
Trust Layer Supporting = IP Layer Supported
IP Spanning Layer
Support Protocols
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Across
Up
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Across
Across

# Trusted Application Layer Router Intermediation

Sender

Trust Spanned Applications

Trust Spanned Application Support Protocols

Trust Spanning Layer

Platform 1 | Platform 2 | Platform Agnostic

Trust Layer Supporting = IP Layer Supported

HTTP FTP SMTP RTP

TCP UDP

IP Spanning Layer

Support Protocols

Down

Hop

Up

Router

Trust Spanned Application Layer Routing and Discovery

Trust Spanned Application Support Protocols

Trust Spanning Layer

Platform 1 | Platform 2 | Platform Agnostic

Trust Layer Supporting = IP Layer Supported

IP Spanning Layer

Support Protocols

Down

Hop

Up

Router

Trust Spanned Application Layer Routing and Discovery

Trust Spanned Application Support Protocols

Trust Spanning Layer

Platform 1 | Platform 2 | Platform Agnostic

Trust Layer Supporting = IP Layer Supported

IP Spanning Layer

Support Protocols

Down

Hop

Receiver

Trust Spanned Applications

Trust Spanned Application Support Protocols

Trust Spanning Layer

Platform 1 | Platform 2 | Platform Agnostic

Trust Layer Supporting = IP Layer Supported

HTTP FTP SMTP RTP

TCP UDP

IP Spanning Layer

Support Protocols

Up

Across

Across

Across

# Last Hop Shortcut Trusted Application Spanning Layer

Sender
Hop
Router
Hop
Router
Last
Hop
Receiver
Trust Spanned Applications
Trust Spanned Application
Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform
Agnostic
Trust Layer Supporting
=
IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support
Protocols
Down
Trust Spanned
Application Layer
Routing and Discovery
Trust Spanned Application
Support Protocols
Trust Spanning Layer
Up
Down
Platform 1
Platform 2
Platform
Agnostic
Trust Layer Supporting
=
IP Layer Supported
IP Spanning Layer
Support
Protocols
Trust Spanned
Application Layer
Routing and Discovery
Trust Spanned Application
Support Protocols
Trust Spanning Layer
Up
Down
Platform 1
Platform 2
Platform
Agnostic
Trust Layer Supporting
=
IP Layer Supported
IP Spanning Layer
Support
Protocols
Trust Spanned Applications
Trust Spanned Application
Support Protocols
Trust Spanning Layer
Up
Across
Platform 1
Platform 2
Platform
Agnostic
Trust Layer Supporting
=
IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support
Protocols
Across
Across

# Last Hop Shortcut Trusted Application Layer Router

Sender
Hop
Router
Hop
Router
Last Hop
Receiver
Trust Spanned Applications
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Down
Up
Trust Spanned Application Layer Routing and Discovery
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
IP Spanning Layer
Support Protocols
Down
Up
Trust Spanned Application Layer Routing and Discovery
Down
Across
Across
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
IP Spanning Layer
Support Protocols
Trust Spanned Applications
Up
Trust Spanned Application Support Protocols
Trust Spanning Layer
Platform 1
Platform 2
Platform Agnostic
Trust Layer Supporting = IP Layer Supported
HTTP FTP SMTP RTP
TCP UDP
IP Spanning Layer
Support Protocols
Across
Across

# Trust Spanning Layer Solution

Minimally sufficient means (satisfies hourglass theorem’s weakness property) for trust spanning layer is an identity (identifier) system security overlay.

Also need routing and discovery protocols that are compatible with the trust spanning layer.

Routing and protocols are not part of the trust spanning layer but augment it. Routing and discovery protocols are essentially adjacent to the trust spanning layer protocol.

# IP Analogy

A MAC Address is to an IP Address as an IP address is to a TSP identifier

A cellular SEI is to an IP address as an IP address is to a TSP identifier

Mapping to/from TSP Identifier and IP Address is like ARP/RARP which map to/from IP address and MAC address

ARP is routing and discovery below IP

Router that forwards source TSP packets to dest TSP is below TSL and is not trustable.

Router that verifies source TSP signatures on packets is below TSL and is not trustable but provides a quality of service.

Router that has TSP identifier and endorses source TSP packets for which it has verified signatures one is above TSL and is a trustable router.

TCP came before IP. TCP is a connected services protocol originally design to support remote terminals (telnet, rsh, rcp) that is inappropriate for other connectionless applications (like rpc/udp). Routing is difficult with TCP. So IP was invented to support all types of protocols that all share one  interface, the IP layer to enable routing. And with IP came UDP/IP and TCP/IP.

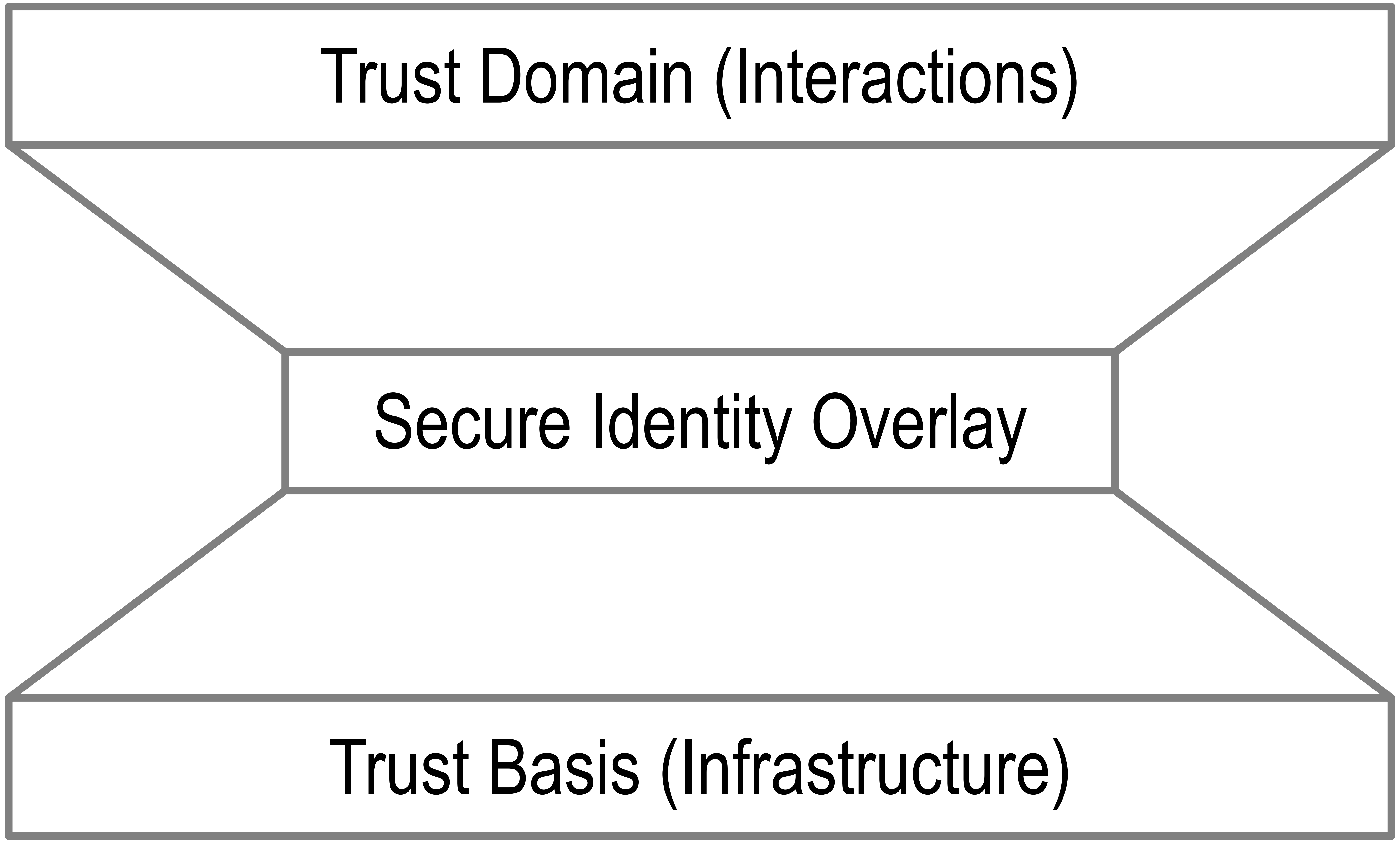
But IP is not secure which forces the need for a trust spanning layer that is secure.

Trustable Connections (signed traffic) between two endpoints are analogous to TCP above IP i.e. above TSP. Untrustable connections (not signed traffic ) are below TSP.

TCP/IP != Connectioned TSP

Signed statements that are not part of a dedicated connection are like UDP datagrams, they are trustable but are topology free for routing purposes.

# Identity (-ifier) System Security Overlay

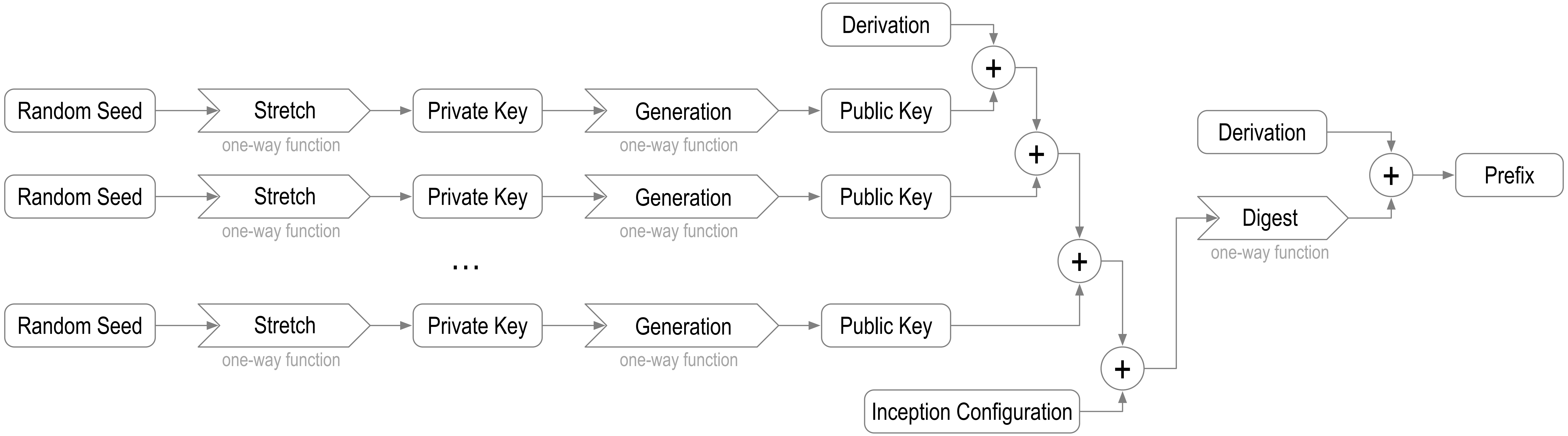


Trust Domain (Interactions)

Secure Identity Overlay

Trust Basis (Infrastructure)

# Cryptographic Root-of-Trust & Persistent Control:
# Self-Certifying Identifier + Key Event Log



Prefix

| Derivation | Inception Digest |
|---|---|

Inception Statement

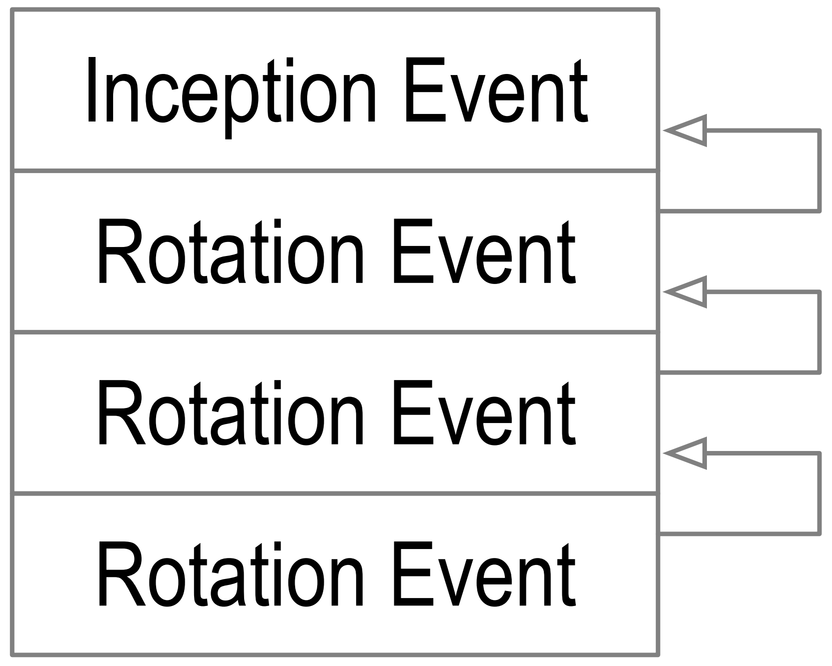
Inception Data

| Derivation | Public Keys | Configuration |
|---|---|---|

Signatures

## Key Event Log

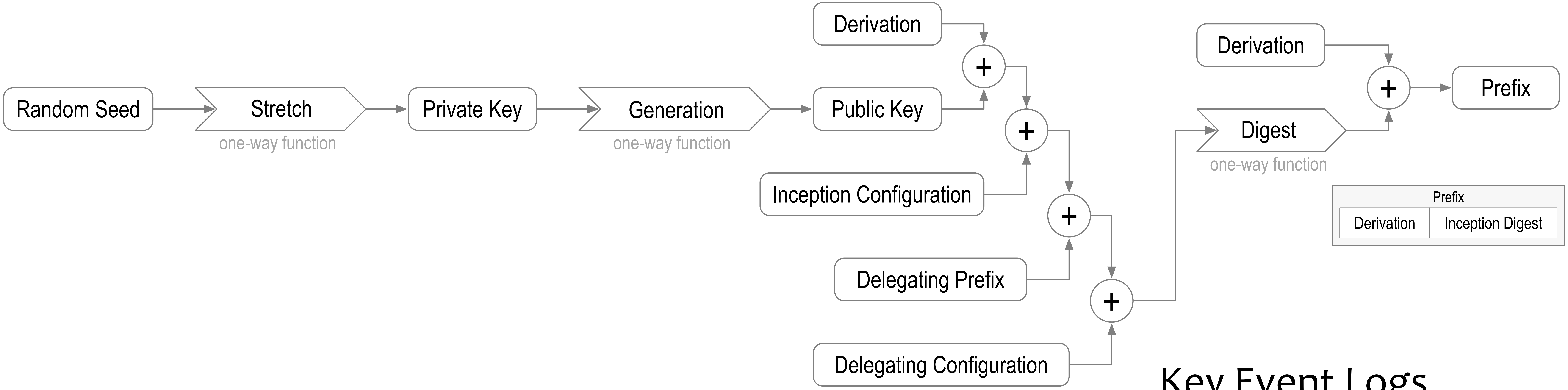
Establishment
Subsequence

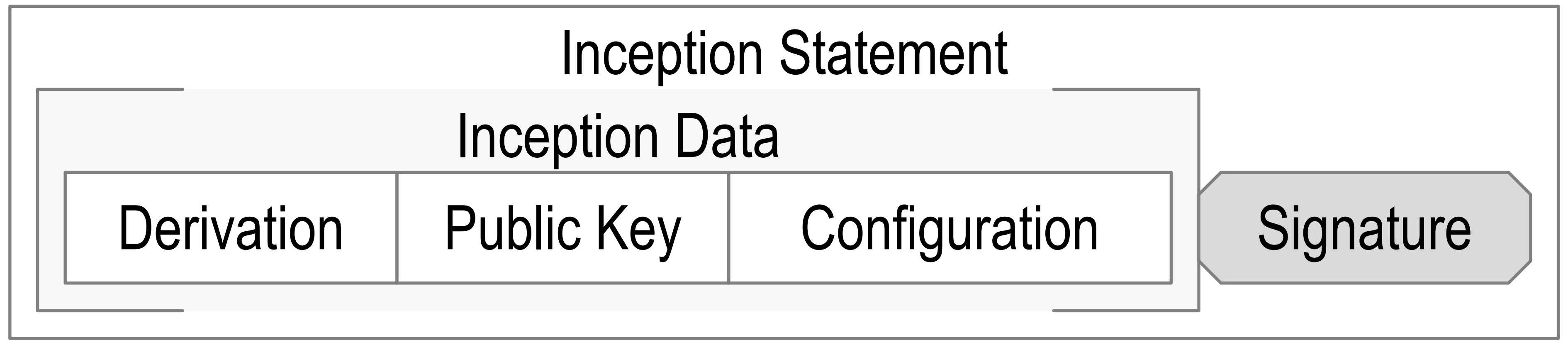


```
EXq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148
```

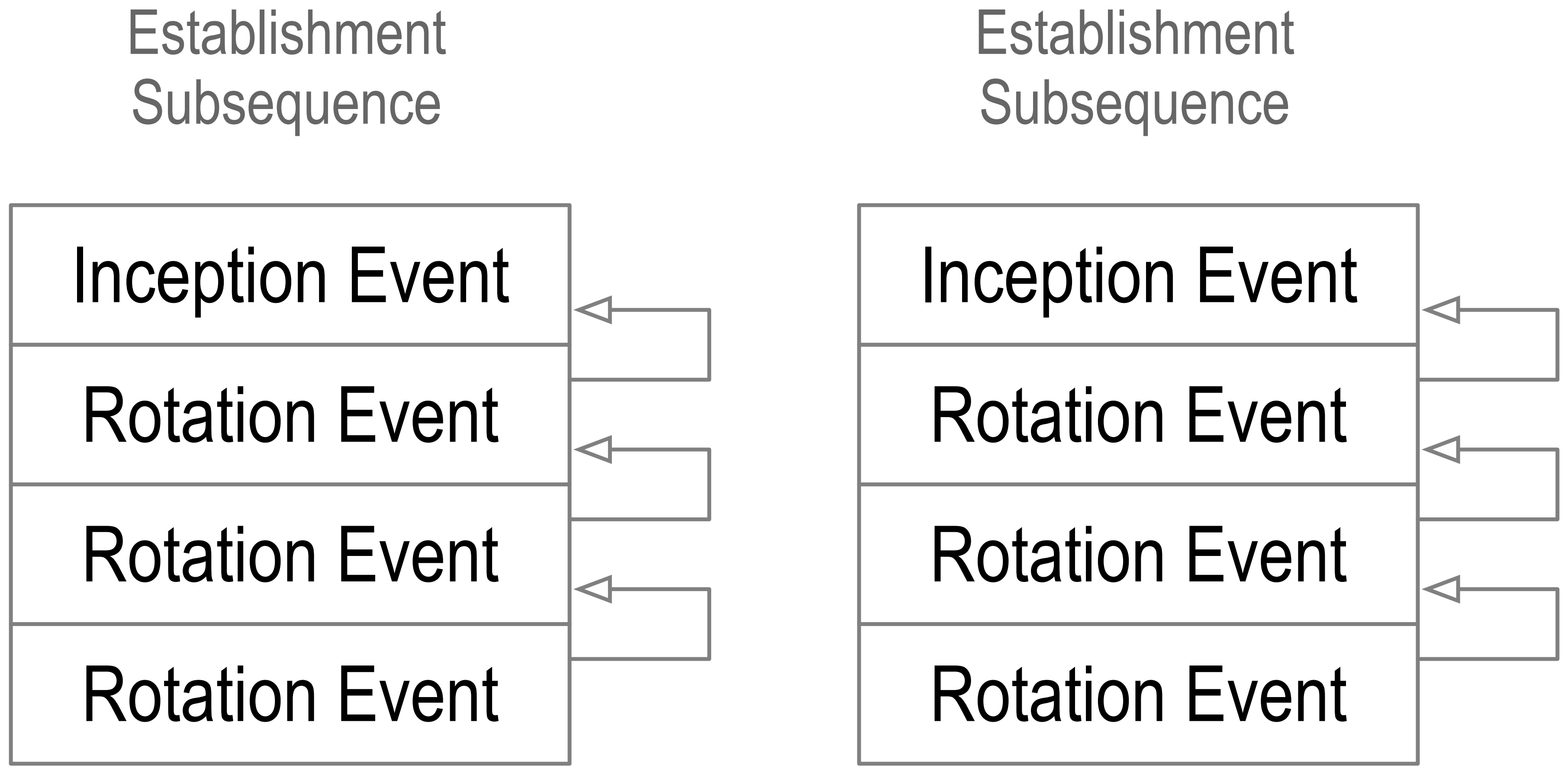
```
did:un:EXq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148/path/to/resource?name=secure#really
```

# Delegated Identifiers





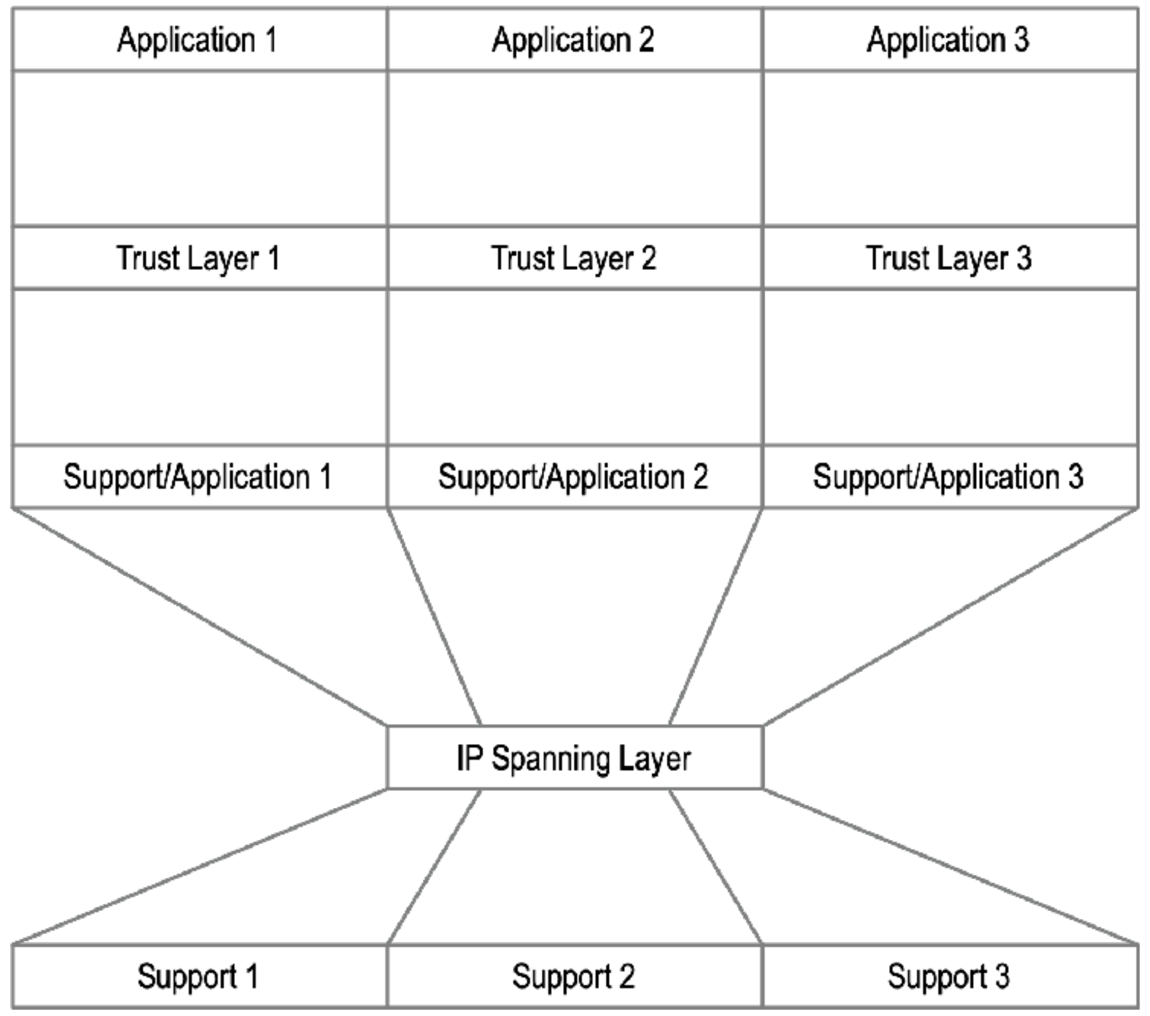
## Key Event Logs



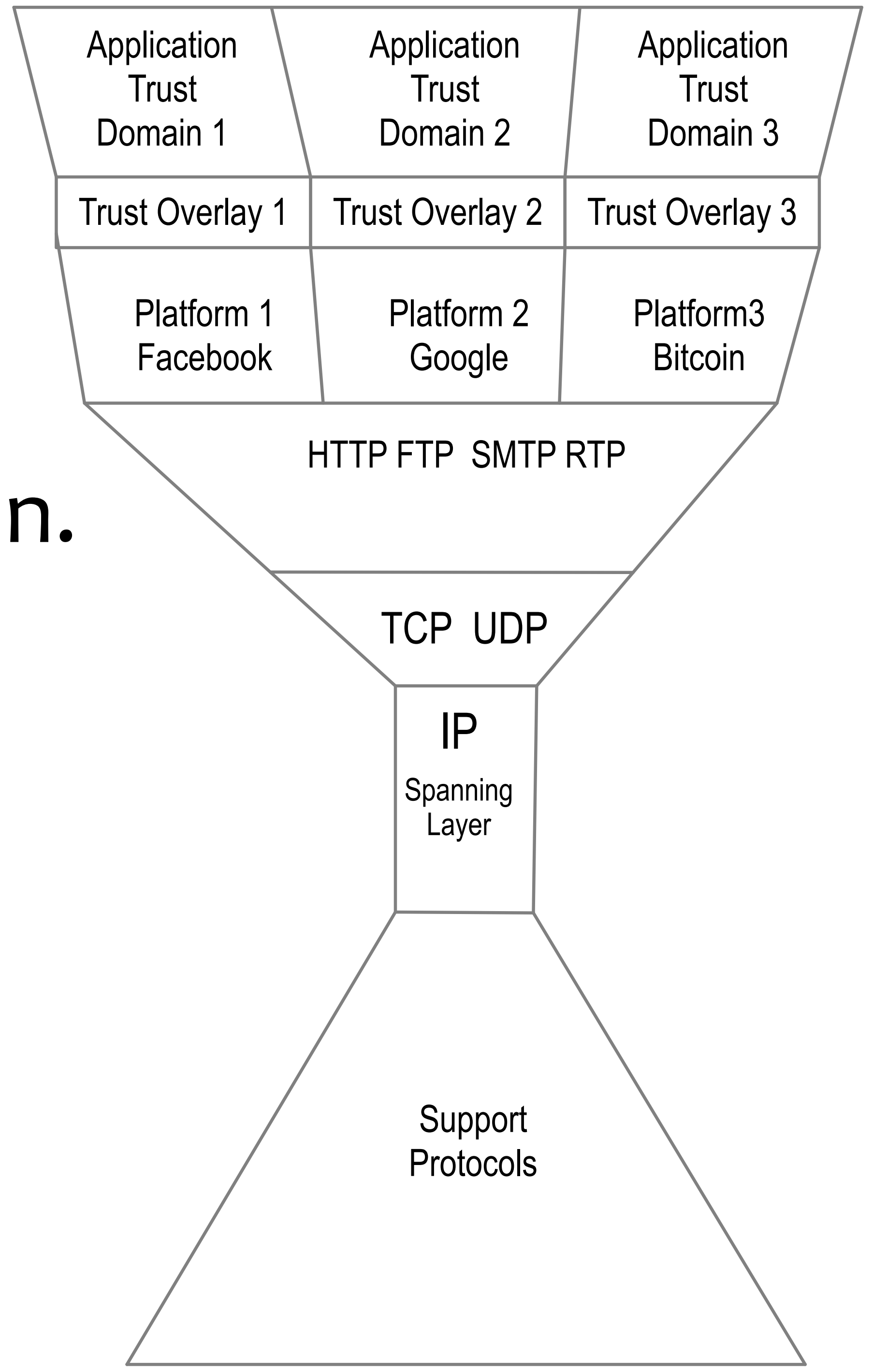
EXq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148

did:keri:EXq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148/path/to/resource?name=sec#yes

# Interoperability for Segmented Trust



All interoperability mechanisms are not created equal.
Interoperability based on a spanning layer is unique.
Interoperability based on a trust-spanning layer is also unique.
A trust-spanning layer uses a common protocol to make one trust domain.
This is different from interoperability across multiple trust domains.



Examples of interoperability across multiple trust domains:

DNS/CA (trust spanning layer protocol but effective trust domain segmentation across multiple CA administrators)

URL based on DNS/CA (common name-spaced resources found across segmented trust domains)

“Login with” to a website supporting multiple OpenID providers (trust domain segmentation across multiple Identity Providers)

SSO with OpenID aggregator (Hide trust domain segmentation behind centralized Identity Provider)

Each host is its own OpenID Identity Provider

Client-side rendering of multiple language-specific backends (hide code segmentation with browser rendering)

Client-side routing of multiple language-specific backend renderers (hide code segmentation with browser routing of code

Universal DID method Resolver (trust domain method code segmentation by resolver rendering of did:docs)

DIDs as Namespace (name spacing segmented trust domains)

Multi-DID Method Wallet as Aggregator (Hide trust domain segmentation behind wallet UX, client side routing & rendering)

# Flaws of DNS/CA as Trust Spanning Layer

Insecure Key Rotation

Binding between the controlling keys and the controlled identifier is asserted by one or more CAs.

Security strength or weakness derived not cryptography but from the operational processes of CAs.

DNS provides rented identifiers under centralized control. DNS protocols are insecure due to certain structural security limitations. Domain validation weakness problem:  DNS is always vulnerable to attacks that allow an adversary to observe the domain validation probes that CAs send. These can include attacks against the DNS, TCP, or BGP protocols (which lack the cryptographic protections of TLS/SSL), or the compromise of routers. Such attacks are possible either on the network near a CA, or near the victim domain itself.

It is difficult to assure the correctness of the match between data and entity when the data are presented to the CA (perhaps over an electronic network), and when the credentials of the person/company/program asking for a certificate are likewise presented.

Aggregation problem: Identity claims (authenticate with an identifier), attribute claims (submit a bag of vetted attributes), and policy claims are combined in a single container. This raises privacy, policy mapping, and maintenance issues.

Delegation problem: CAs cannot technically restrict subordinate CAs from issuing certificates outside a limited namespaces or attribute set; this feature of X.509 is not in use. Therefore, a large number of CAs exist on the Internet, and classifying them and their policies is an insurmountable task. Delegation of authority within an organization cannot be handled at all, as in common business practice.

Federation problem: Certificate chains that are the result of subordinate CAs, bridge CAs, and cross-signing make validation complex and expensive in terms of processing time. Path validation semantics may be ambiguous. The hierarchy with a third-party trusted party is the only model. This is inconvenient when a bilateral trust relationship is already in place.

DNS/CA is badly broken.

Attempts to secure it without changing its fundamental design is like putting a bandage on a compound fracture.

https://en.wikipedia.org/wiki/X.509

https://en.wikipedia.org/wiki/Certificate_authority

# Flaws of original PGP Web-of-Trust as Trust Spanning Layer

No in-band Key Rotation mechanism

Limited supporting protocols (non minimally sufficient support)

Limited supported protocols (all essential applications not supported)

# Using the Hourglass Theorem in Protocol Stack Design

Taking a descriptive view of the hourglass allows us to use it as an analytical or predictive tool to understand the impact of a community's adopting a particular interface as a standard, be it de jure or de facto. Making the distinction between the use of the hourglass as a descriptive tool or as a means of justifying a standard also explains how different hourglasses can be examined and compared within the discussion of the same layered system. Every prospective spanning layer has an associated pre- and post-image, regardless of whether it is considered for any kind of standardization.

Pre-image of layer S is the set of services that support S

Post-image of layer S is the set of services that S supports.

(https://cacm.acm.org/magazines/2019/7/237714-on-the-hourglass-model/abstract)

# Trade-offs

The balance between more applications and more supports is achieved by first choosing the set of necessary applications N and then seeking a spanning layer sufficient for N that is as weak as possible. This scenario makes the choice of necessary applications N the most directly consequential element in the process of defining a spanning layer that meets the goals of the hourglass model.

Note the implication that the tradeoff between the weakness of the spanning layer and its sufficiency for a particular set of applications N is unavoidable. This suggests the design of a spanning layer may have a tendency to fail if it attempts to both achieve a high degree of weakness and also be sufficient to support a large set of necessary applications.

For example: the set of necessary applications may be only verifiably authentic attestations for the application stack and a minimally sufficient routing and discovery mechanism for the routing stack.

Over time broader adoption induced by the broader support likewise induced by the simpler (weaker) spanning layer drives implementers to create more applications and protocol layers that are supported by that spanning layer. The goal is adoptability via most supporting services for only the necessary or essential supported services.

(https://cacm.acm.org/magazines/2019/7/237714-on-the-hourglass-model/abstract)

# Methodology

The hourglass model can be understood as describing the general shape of the subspace that we navigate in designing layered systems. If one goal is maximizing possible supports, then the Hourglass Theorem tells us that

the slope of the subspace of feasible solutions when considering this goal as a function of the logical weakness of the spanning layer is non-negative. We have no metrics for logical strength or for the size of the space of possible solutions, only for the notions of one service description being weaker than another and one set of service descriptions being included in another.

Resist the temptation to thicken the "spanning layer" in order to support everyone's preferred application as is. Instead intent everyone to port their application to use the thin "spanning layer" because of broader support below the spanning layer.

This thickening to support existing applications increases interoperability but is not spanning or hourglass interoperability and ultimately results in less adoption not more. Cases in point, DID method proliferation interoperability resulting in thick wallets; Shared ledger proliferation interoperability resulting in thick cross ledger transfer protocols.

From M x N to M+N with the hourglass (M applications and N supports)

(https://cacm.acm.org/magazines/2019/7/237714-on-the-hourglass-model/abstract)

(https://www.oilshell.org/blog/2022/02/diagrams.html)

# Hourglass vs. End-to-end

The countervailing theory for IP dominance is called the end-to-end theory. End-to-end theory (OSI ISO model) is also about layering in order to encapsulate functionality. Moving a function higher or lower in the protocol stack may or may not weaken or strengthen a lower layer. They are not strongly correlated, end-to-end analysis is therefore less exacting than hourglass analysis for predicting maximum adoption and scalability.

(https://cacm.acm.org/magazines/2019/7/237714-on-the-hourglass-model/abstract)

# Weakness Adoption Advantage Examples

- IP does not enforce the property of symmetric routable endpoints. This would make it stronger (thicker)
  - This weakness enabled DHCP and NAT below to leverage TCP and UDP ports above without changing IP in the middle. Most internet access does not require symmetry. TCP ephemeral ports allows asymmetry so DHCP and NAT were invented to work around too few IPv4 addresses.
- Mobile IP leverages multiple routed IP addresses to maintain a persistent connection for mobile IP devices. Cellular networks converted to IP due to the broadening of support given by mobile IP.
- IPv6 has enough addresses so no need for NAT and DHCP.
  - IPv6 routing includes Mobile IP.
  - IPv6 could do away with MAC addresses and make the MAC address an IPv6 service address inside the IPv6 block allocated to a device.
  - But IPv6 could have been thinner. Only add address space not add features hence slower IPv6 adoption. Cellular saved IPv6.
- HTTP is weaker because it does not support synchronous resource state (full duplex) or peer-to-peer. This simplified HTTP and led to rapid adoption.
  - HTTP cache management is a weaker optional solution to synchronizing resource state.
  - Later web sockets, HSL, WebRTC were layered on top of HTTP for full duplex peer-to-peer streaming and synchronized resource state.

# Internet is a binary protocol, Trust should be a ? protocol

With CESR the Trust spanning layer protocol can eat its cake and have it too.

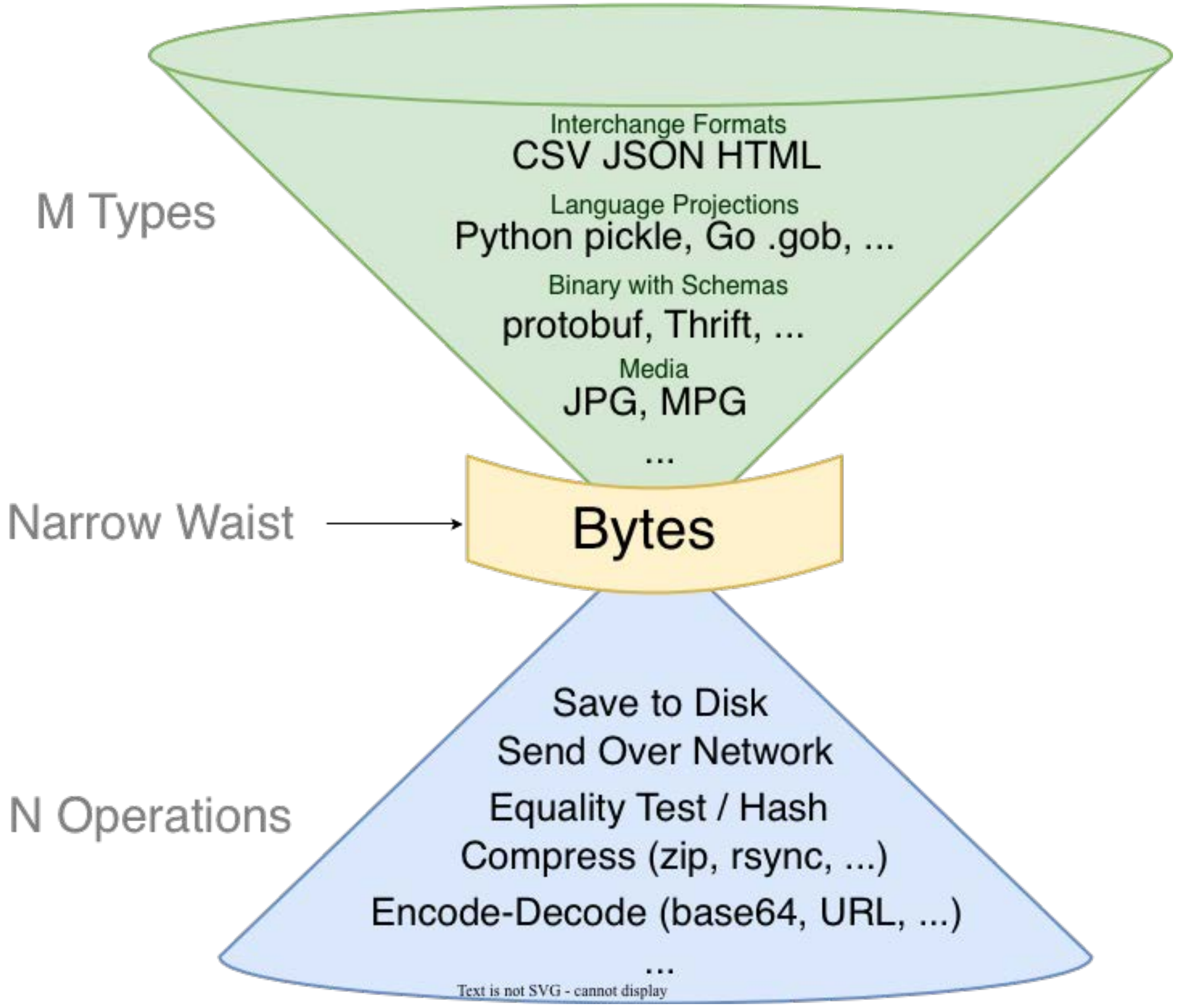
Avoid the text vs. binary protocol wars.

It can be text-native for adoptability but use binary for terseness when beneficial.

# CESR Unifies Bytes and Text Hourglasses (Unix)

https://www.oilshell.org/blog/2022/02/diagrams.html

https://www.oilshell.org/cross-ref.html?tag=osh-language#osh-language



M Types

Interchange Formats
CSV JSON HTML
Language Projections
Python pickle, Go .gob, ...
Binary with Schemas
protobuf, Thrift, ...
Media
JPG, MPG
...

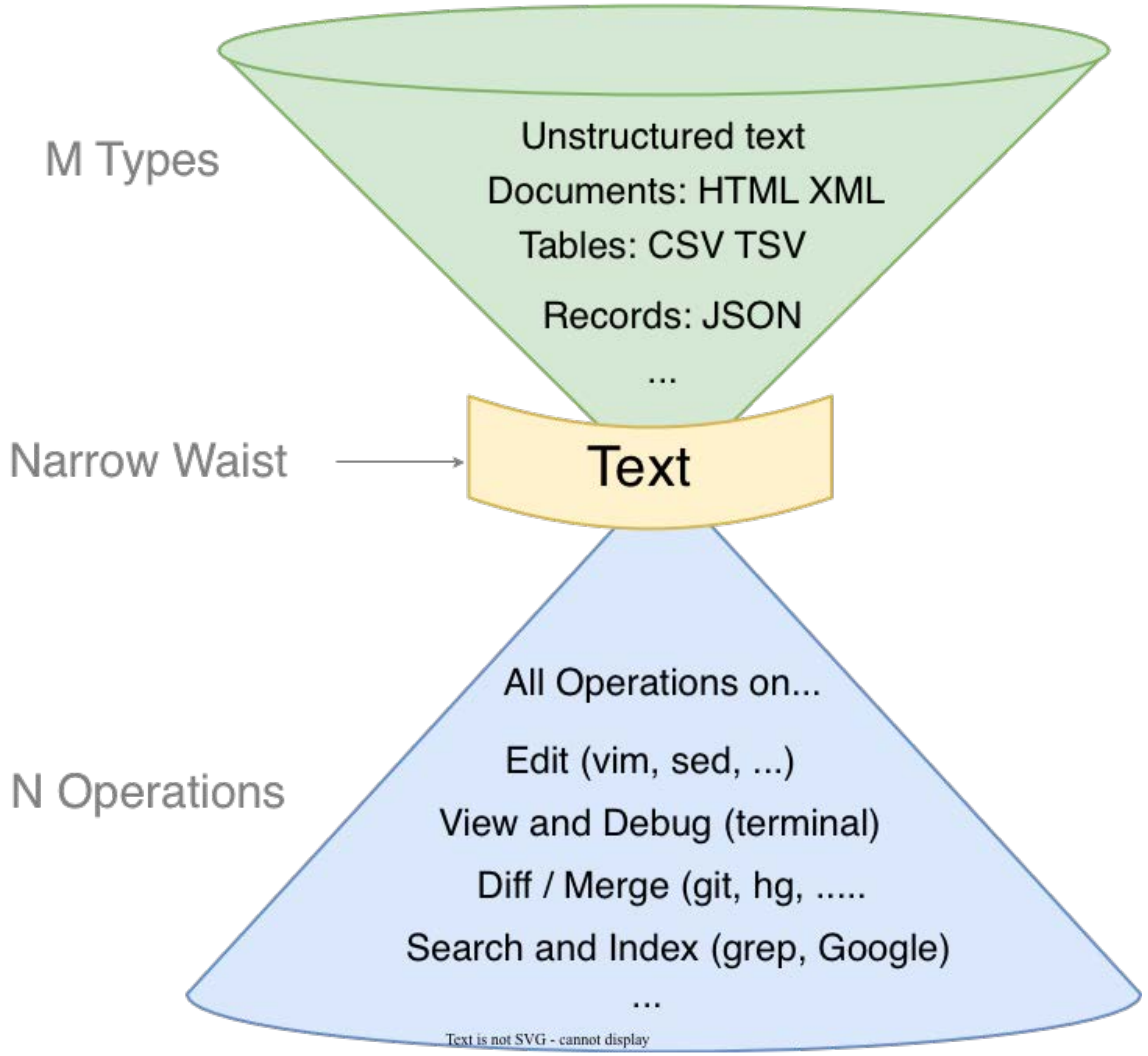
Narrow Waist

Bytes

N Operations

Save to Disk
Send Over Network
Equality Test / Hash
Compress (zip, rsync, ...)
Encode-Decode (base64, URL, ...)
...

Text is not SVG - cannot display



M Types

Unstructured text
Documents: HTML XML
Tables: CSV TSV
Records: JSON
...

Narrow Waist

Text

N Operations

All Operations on...
Edit (vim, sed, ...)
View and Debug (terminal)
Diff / Merge (git, hg, .....
Search and Index (grep, Google)
...

Text is not SVG - cannot display

# Two Types of Trust

Attributional Trust (“who” said it)

Reputational Trust (“what” was said)

RAP: Reputable Authenticatable Pseudonymity

Cryptographic Pseudonyms (Cryptonyms) enable both security and privacy.

Privacy to the degree unlinked with natural person i.e. a pseudonym.

Securely attributable = authenticatable and accountable

Reputable = Accountable

Identity Graph to manage mutual links between different identifiers and to derive reputation by reference.

# Who Blesses Whom (or what)

- AID Controller blesses controlling keys and content addresses (attributional trust via cryptographic secure attribution to AID)
- Trust Anchor blesses other AIDs and content addresses (reputational trust via reference lends credibility and veracity to the what )
- Trust Registry blesses Trust Anchors and other content addresses. (layered reputational trust)

# Human *Trust-Basis*

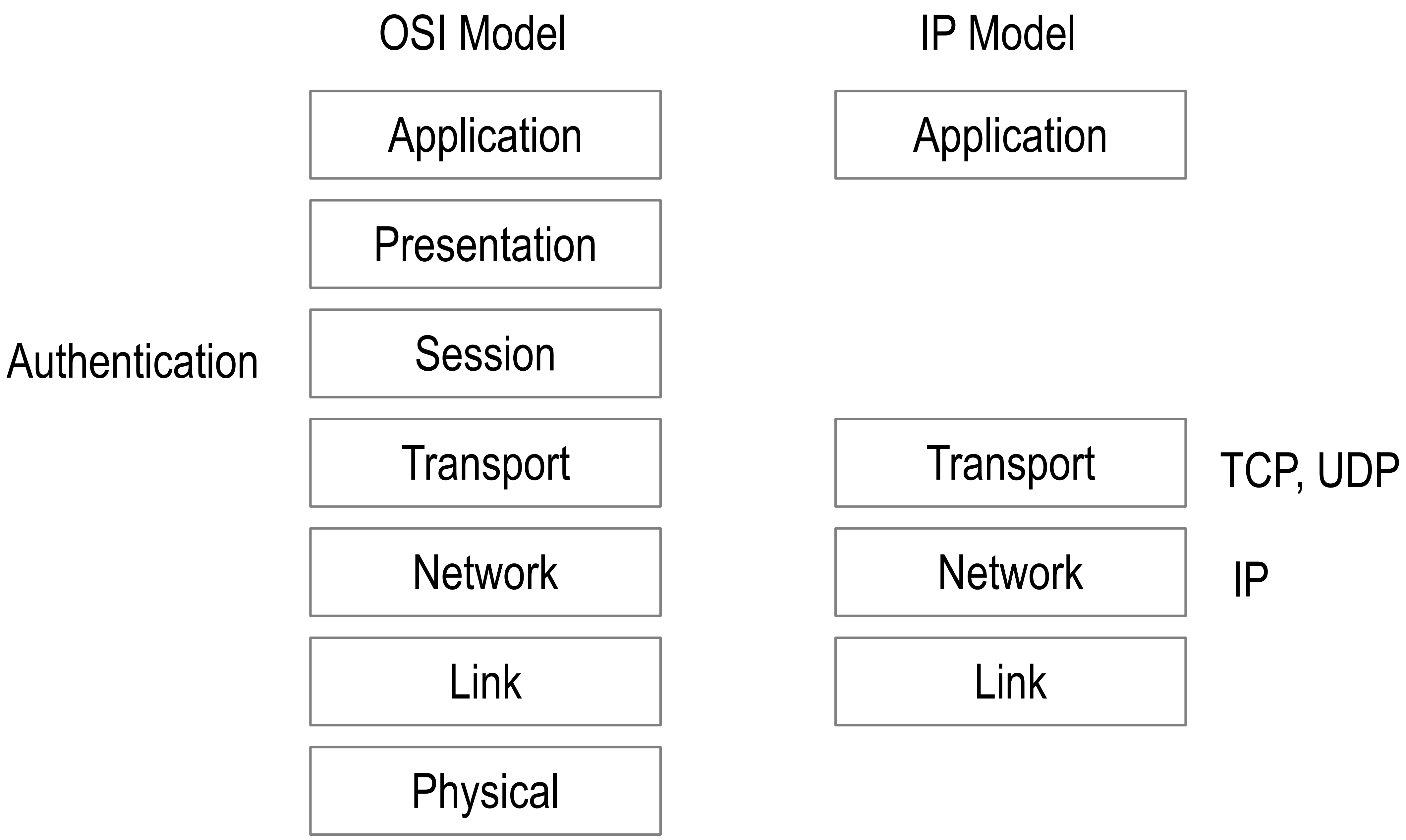
*"in person"*

*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*

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



|  | OSI Model | IP Model |  |
|---|---|---|---|
|  | Application | Application |  |
|  | Presentation |  |  |
| Authentication | Session |  |  |
|  | Transport | Transport | TCP, UDP |
|  | Network | Network | IP |
|  | Link | Link |  |
|  | Physical |  |  |

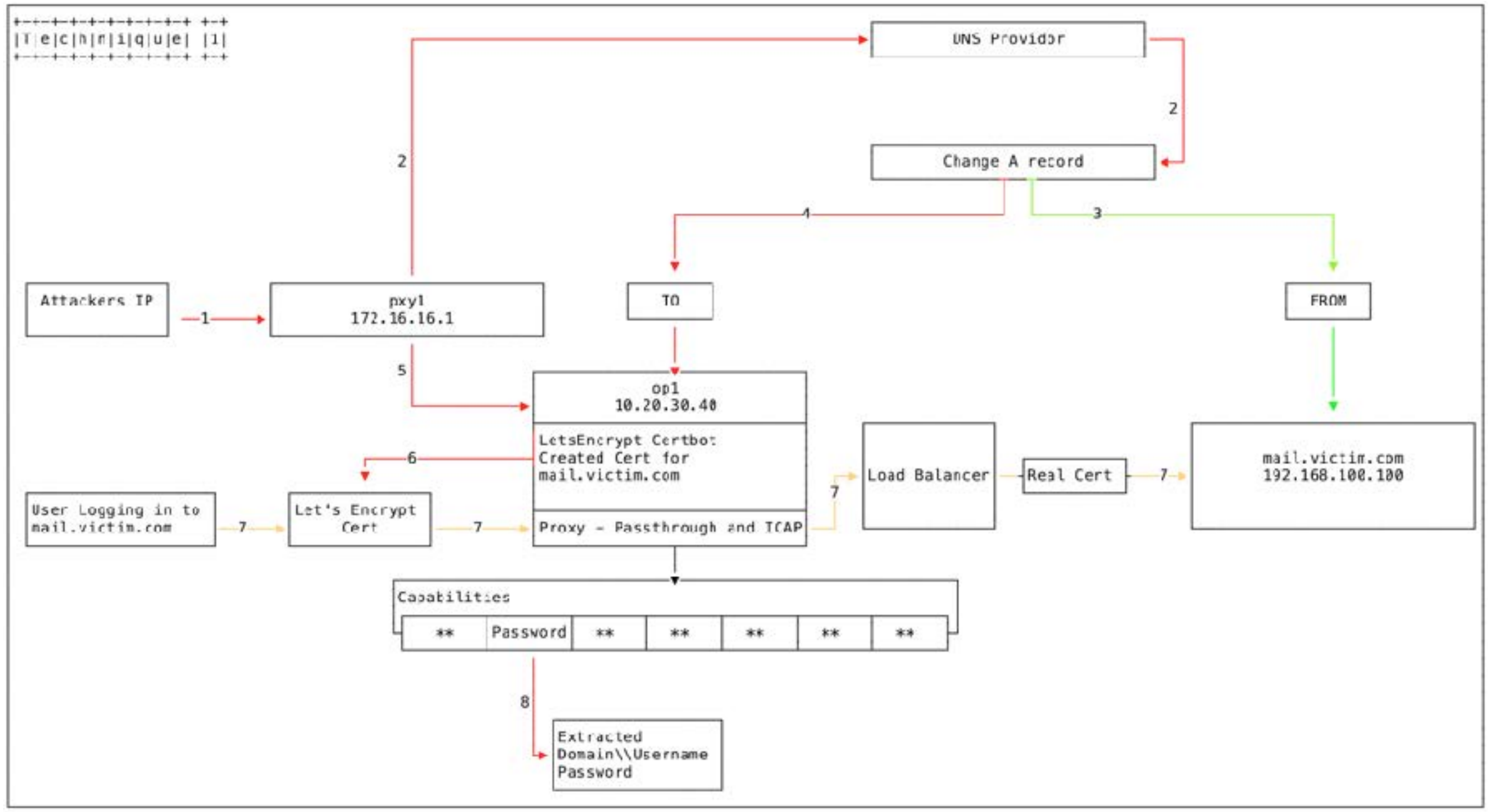
Instead …

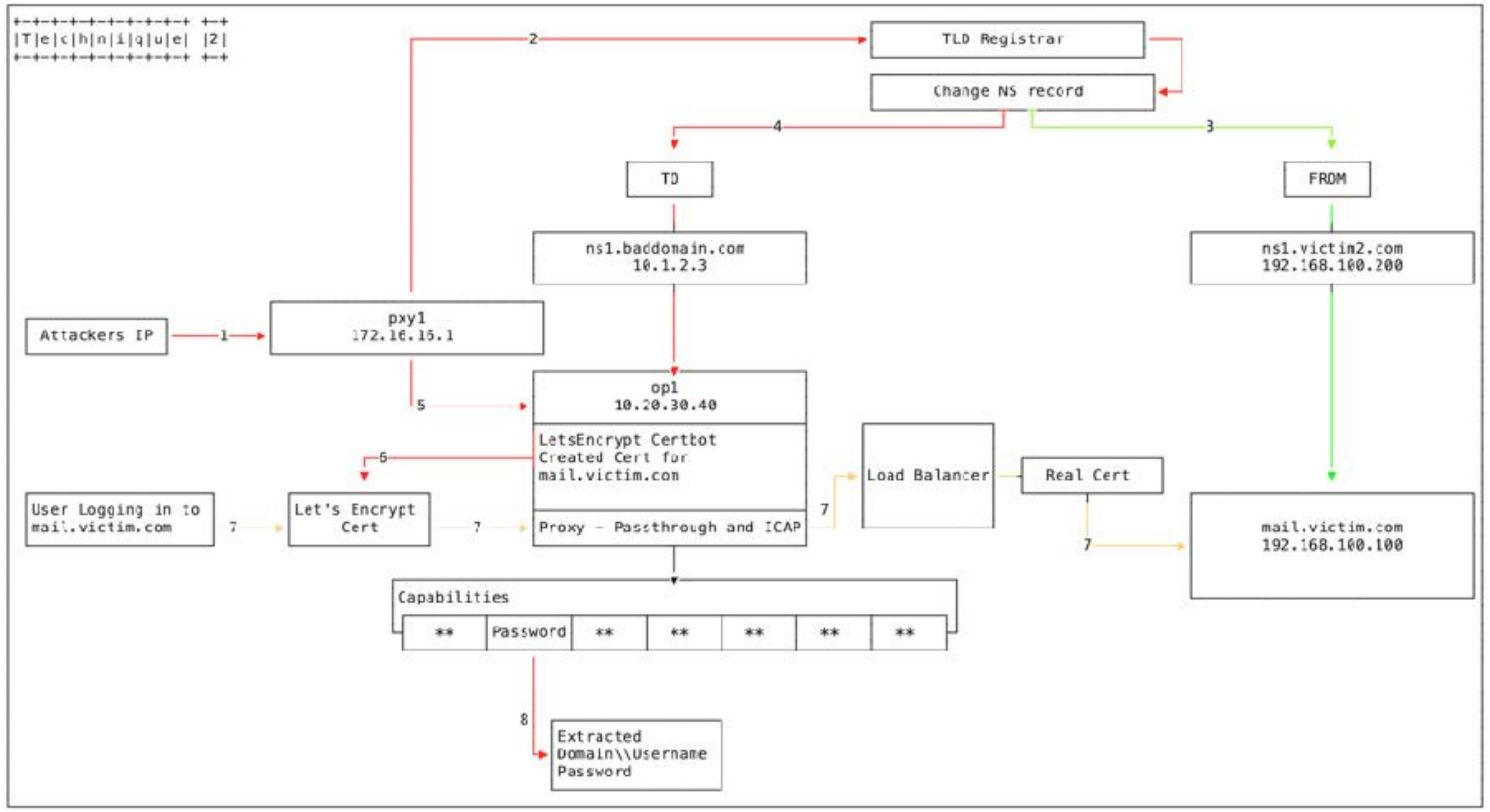
We use *bolt-on* identity system security overlays. (DNS-CA … )

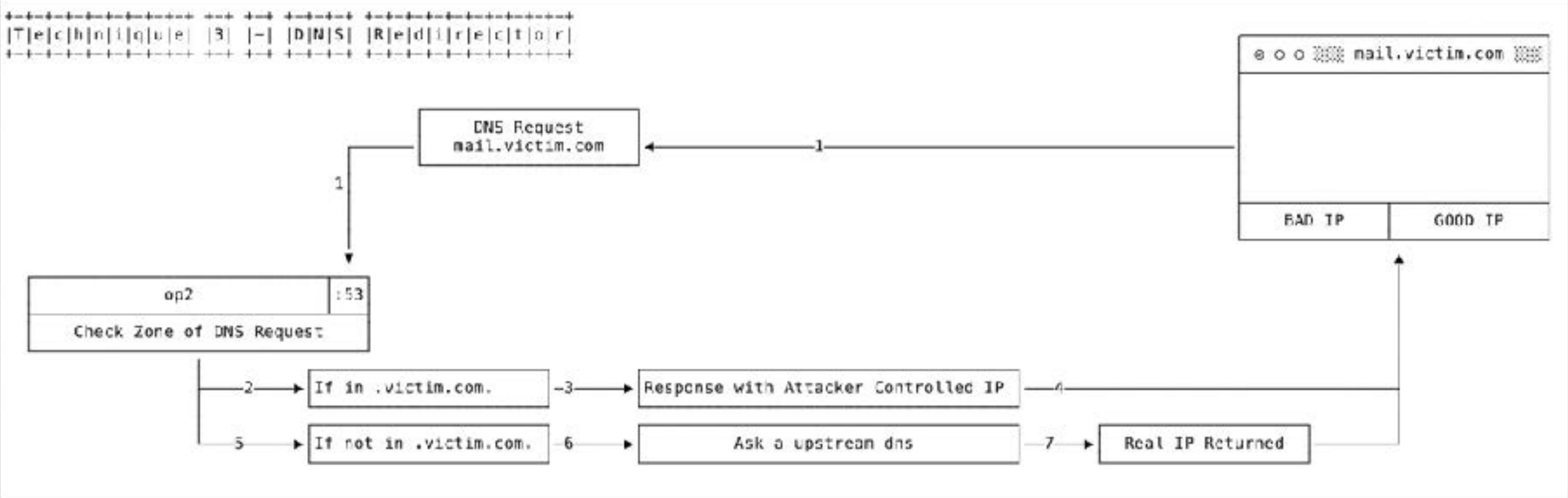
# DNS Hijacking

DNS hijacking uses clever tricks that enable 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/





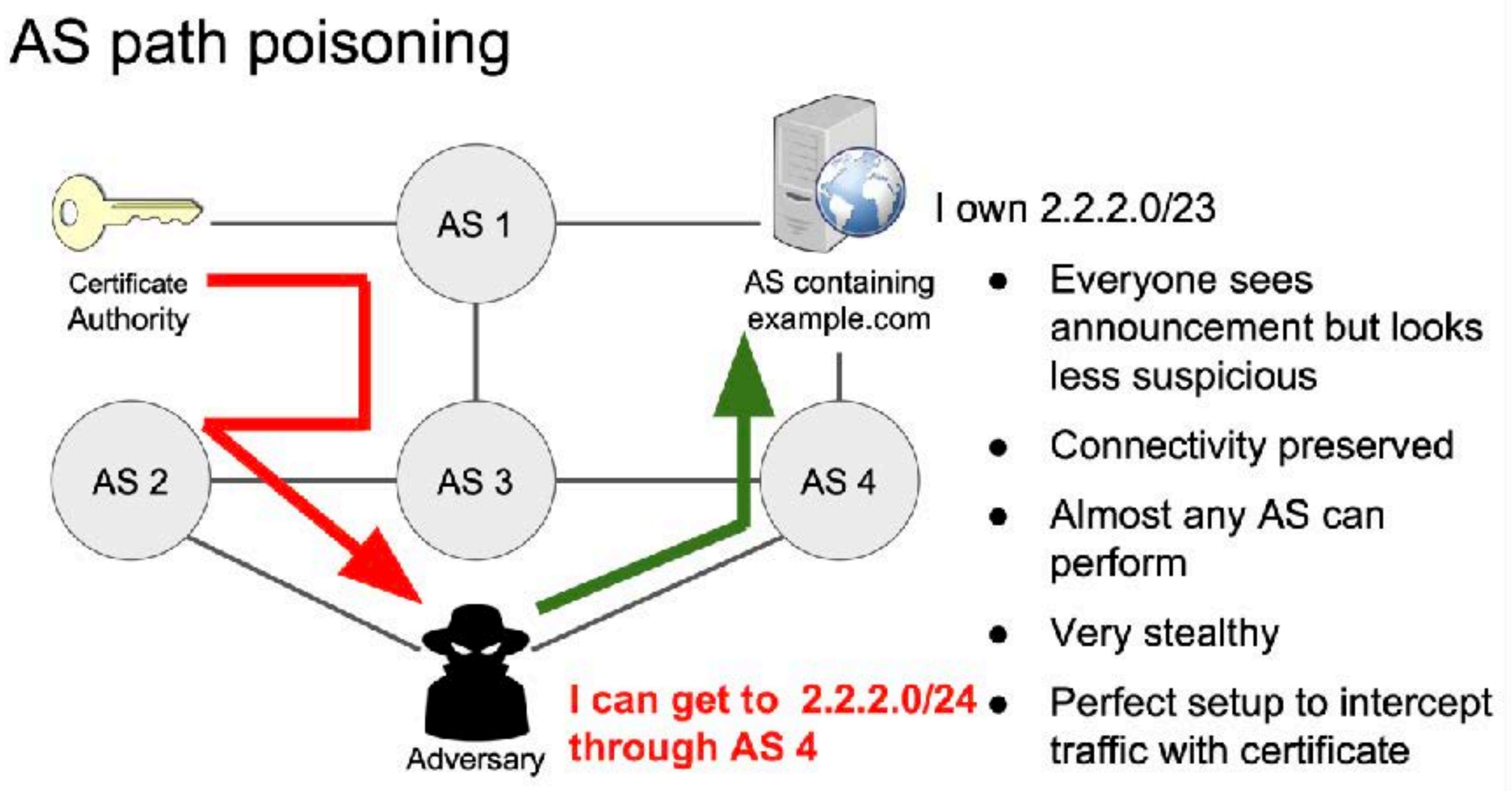


# BGP Hijacking: AS Path Poisoning

Spoofing domain verification process from CA enables attackers to obtain valid TLS certificate for hijacked domains.

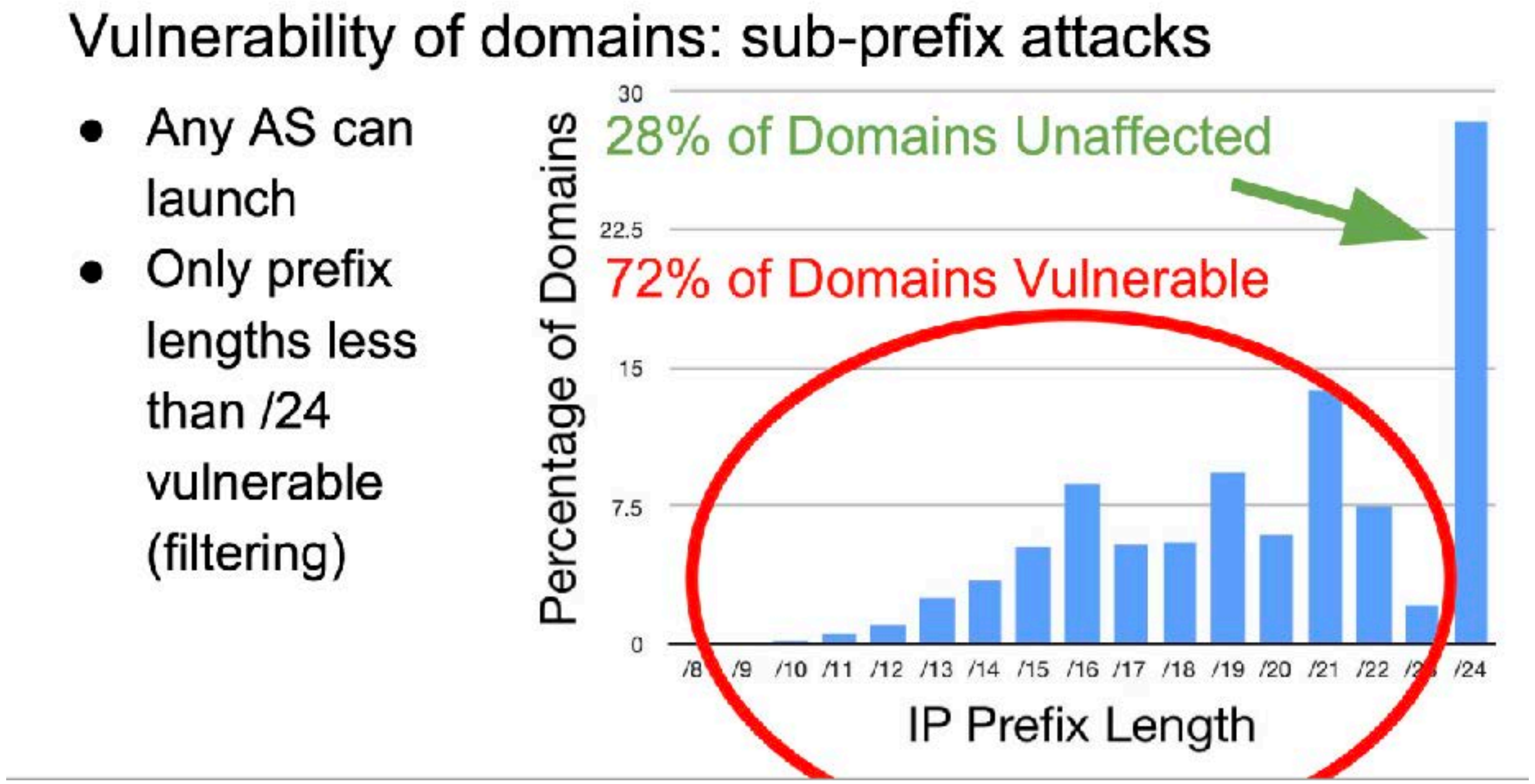
Birge-Lee, H., Sun, Y., Edmundson, A., Rexford, J. and Mittal, P., “Bamboozling certificate authorities with {BGP},” vol. 27th {USENIX} Security Symposium, no. {USENIX} Security 18, pp. 833-849, 2018 https://www.usenix.org/conference/usenixsecurity18/presentation/birge-lee

Gavrichenkov, A., “Breaking HTTPS with BGP Hijacking,” BlackHat, 2015 https://www.blackhat.com/docs/us-15/materials/us-15-Gavrichenkov-Breaking-HTTPS-With-BGP-Hijacking-wp.pdf



## AS path poisoning

- Everyone sees announcement but looks less suspicious
- Connectivity preserved
- Almost any AS can perform
- Very stealthy
- Perfect setup to intercept traffic with certificate



## Vulnerability of domains: sub-prefix attacks

- Any AS can launch
- Only prefix lengths less than /24 vulnerable (filtering)

# Zero-Trust Architecture

Never Trust, Always Verify

Perimeter-less Security Model

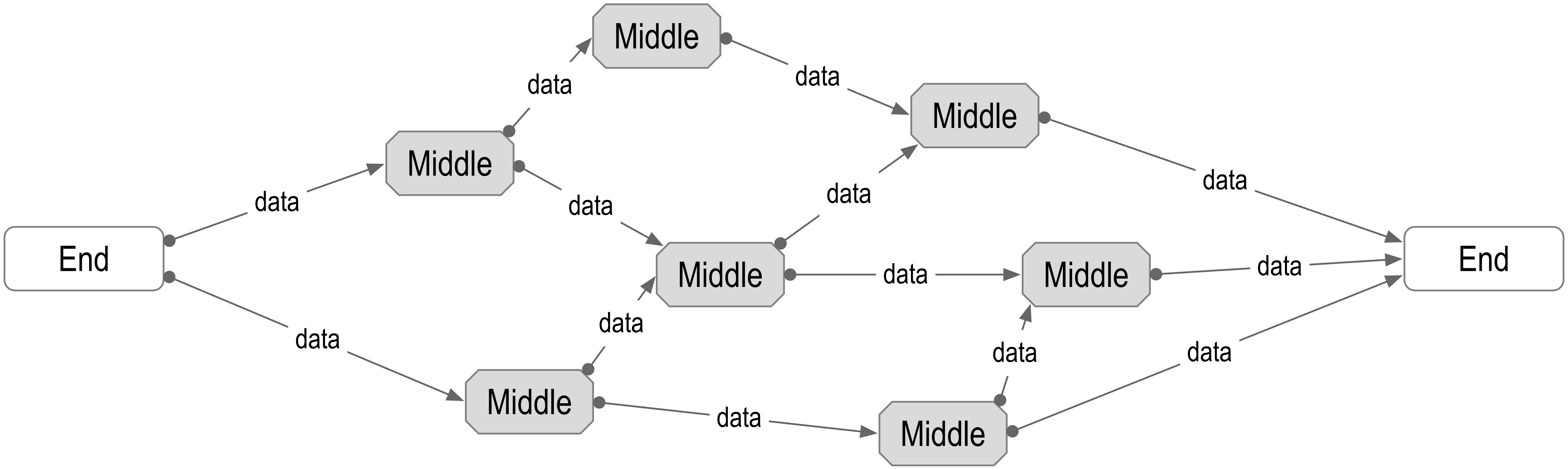
Data is signed and/or encrypted both in motion and at rest.

Zero-Trust Spectrum: ratio of trusted surface to verifiable surface

End goal = all data has end-to-end verifiable authenticity

# End Verifiability

## *End-to-End* Verifiability



If the edges are secure, the security of the middle doesn’t matter.

*Ambient Verifiability*: any-data, any-where, any-time by any-body

*Zero-Trust-Computing*

*Its much easier to protect one’s private keys than to protect all internet infrastructure*

# PKI Then and Now

Who uses a password manager?

Who uses an authenticator app?

Who uses password-less login?

Then: Managing private keys impossible for users, federated identity.

Now: Mobile Devices with MFA & secure boot, password-less login.

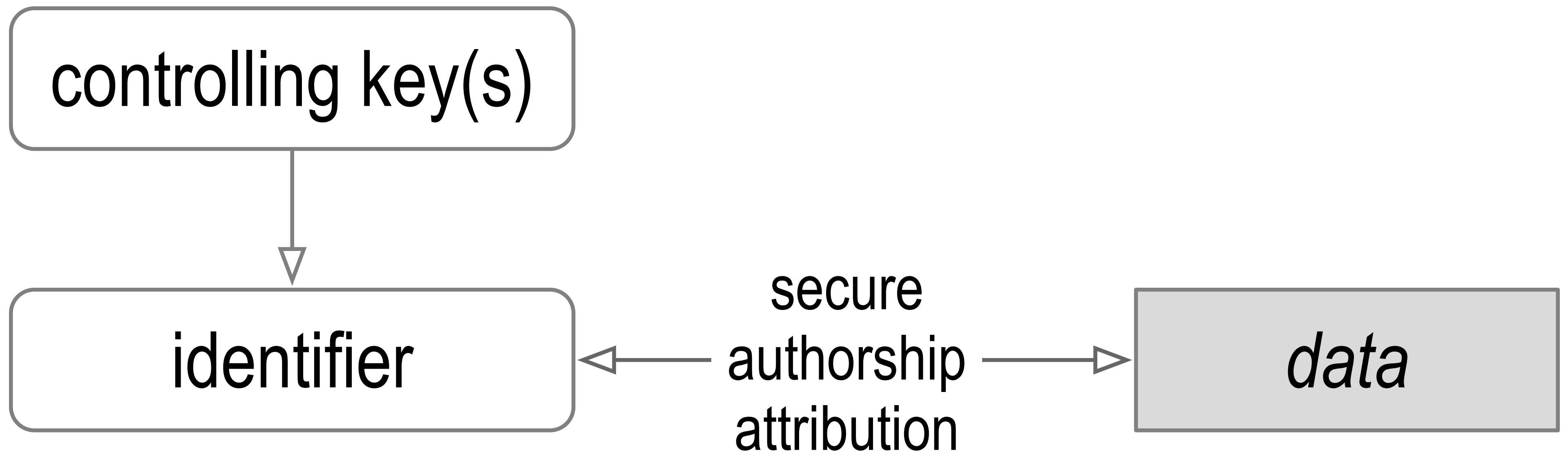
Then: Weak Crypto

Now: Strong crypto: ECC signing & asymmetric encryption.

Then: Perimeter security, no persistence of control over identifiers.

Now: authentic web and zero-trust architecture for identity.

# Flaw of PKI (DNS/CA)



Use of private keys exposes them to side-channel attack.

Over-time, exposure makes private keys weak.

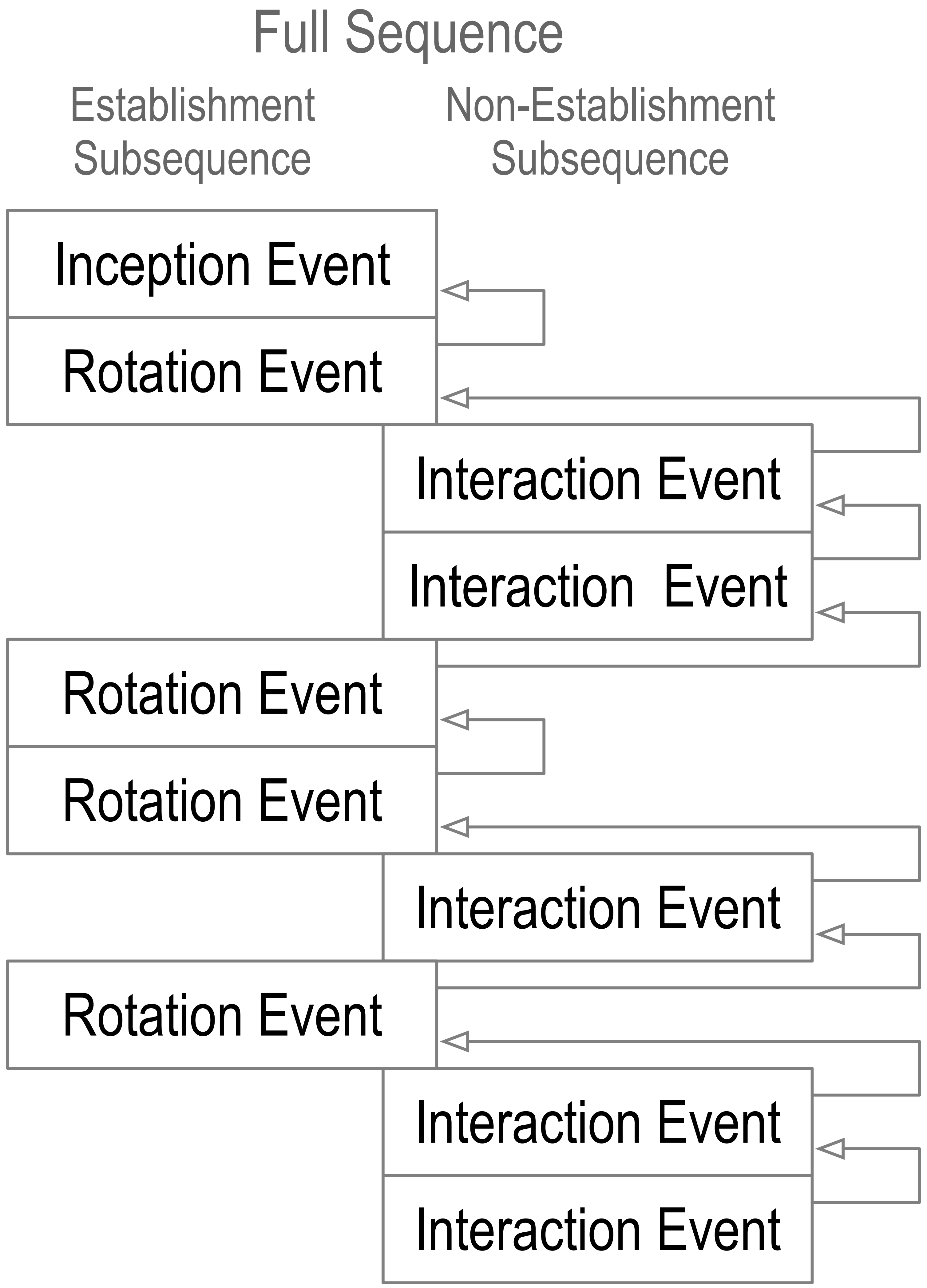
Thus, from time-to-time one must revoke and replace the controlling private keys for a given identifier

Hence key rotation

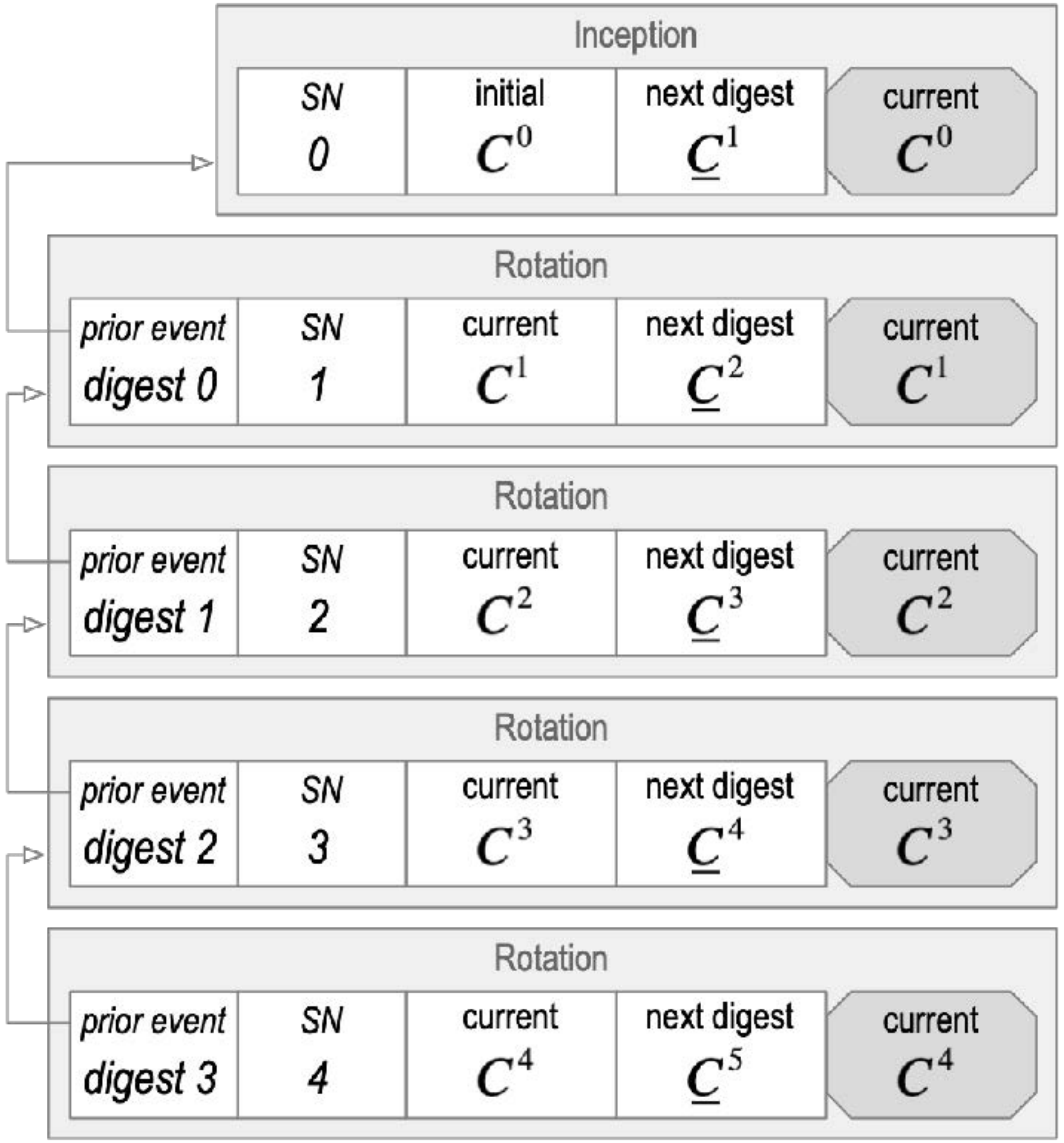
Existing PKI must re-establish the root-of-trust with each rotation thereby making it vulnerable to attack

Breaks the chain-of-trust-of-control over the identifier

# Solution: Key Pre-Rotation



*duplicity evident*
*verifiable data structure*



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

D.J. Bernstein: https://cr.yp.to/hash/collisioncost-20090517.pdf