Security Architectures Issues
Zero-Trust Trust Domains

Roots-of-trust models: EUID ARF 1.4 and KERI

KERI

https://keri.one
https://github.com/WebOfTrust

Samuel M. Smith Ph.D.
sam@keri.one

Resources

Documentation:
https://keri.one/keri-resources/

KERI/ACDC Community: (meetings, open source code Apache2, specification drafts)
https://github.com/WebOfTrust
https://github.com/WebOfTrust/keri

ToIP: (meetings, specifications OWF License)
https://trustoverip.org/
https://wiki.trustoverip.org/display/HOME/ACDC+(Authentic+Chained+Data+Container)+Task+Force
https://wiki.trustoverip.org/display/HOME/Trust+Spanning+Protocol+Task+Force

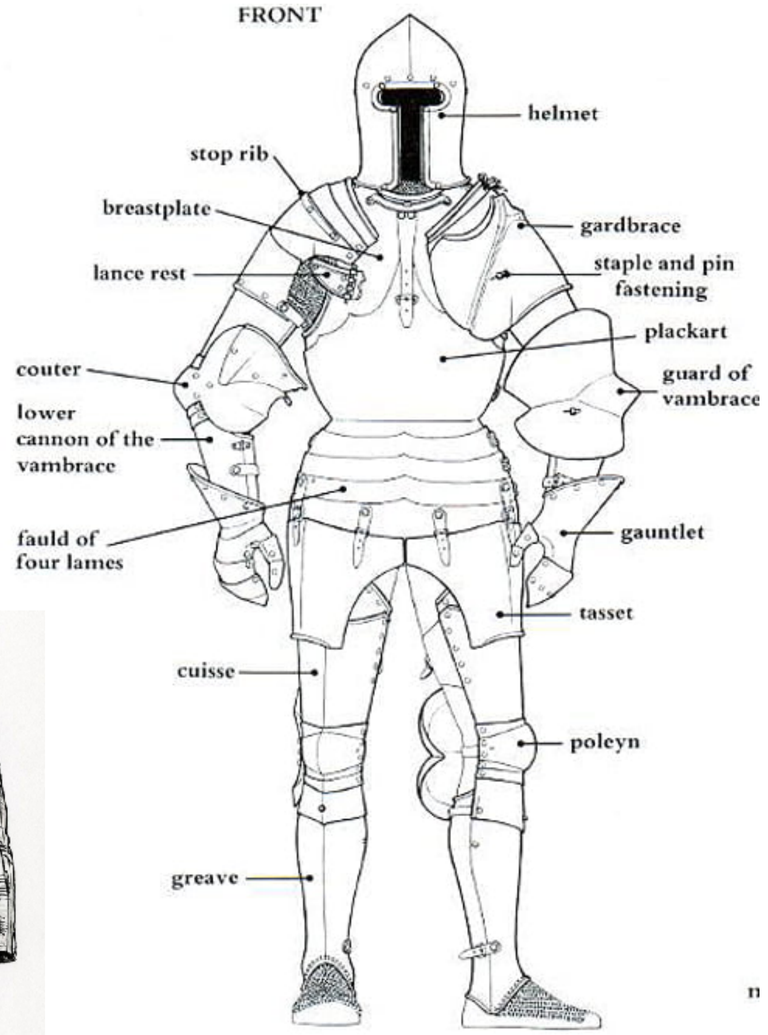
GLEIF:
https://www.gleif.org/en/lei-solutions/gleifs-digital-strategy-for-the-lei/introducing-the-verifiable-lei-vlei

healthKERI:
https://healthkeri.com/

KERI GLEIF TRUST Over IP FOUNDATION ISO DirectTrust® healthKERI

Armor

Preventing *wounds* especially *fatal* *wounds*



Each component protects a vital area from injury.

Remove even one component and the adversary will target that area to the exclusion of all else.

Hard Problems & Solutions

Moving Data Across Trust Domains.

No Shared Secrets

- No passwords
- No shared encryption keys
- No bearer tokens
- No shared private keys

Key Management (rotation)

True Zero-Trust = Sign Everything

Global Portability At-Scale

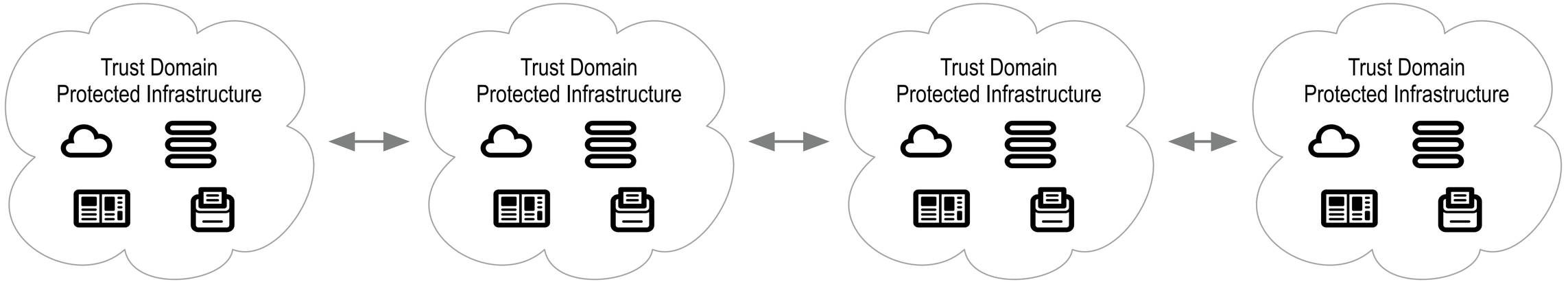
Trust Spanning Protocol (TSP)(SPAC)

Authentic Chained Data Container (ACDC)

Key Event Receipt Infrastructure (KERI)

Composable Event Streaming Representation (CESR)

GLEIF vLEI



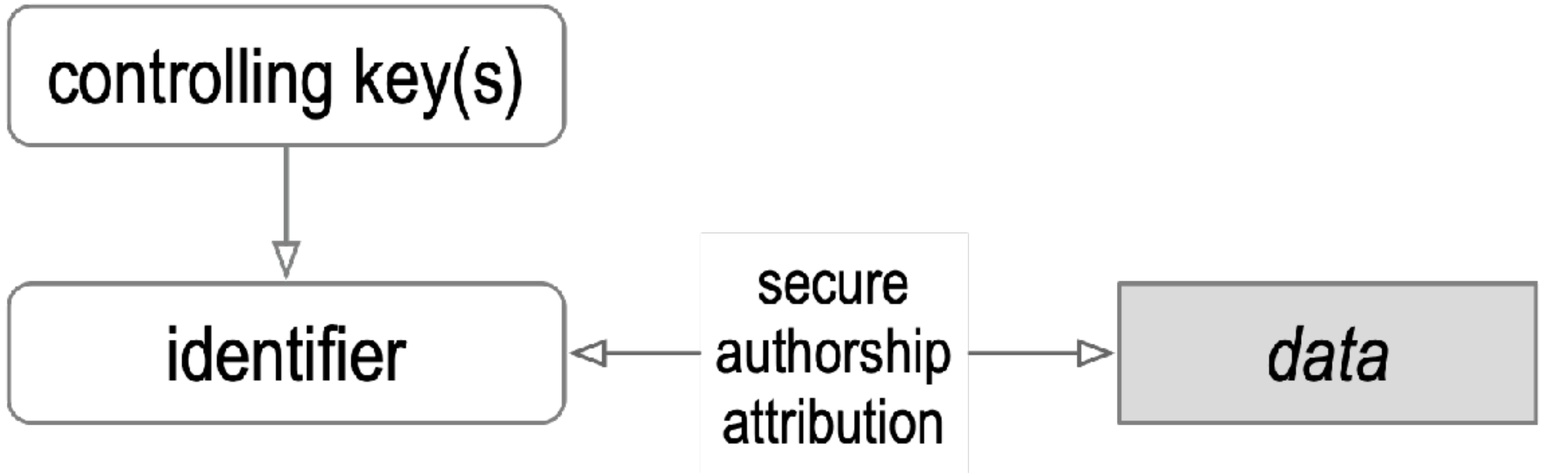
Universal Secure Attribution Problem

Establish authorship of data, documents, credentials, entitlements, …

= Verifiable secure attribution of any communication to its source

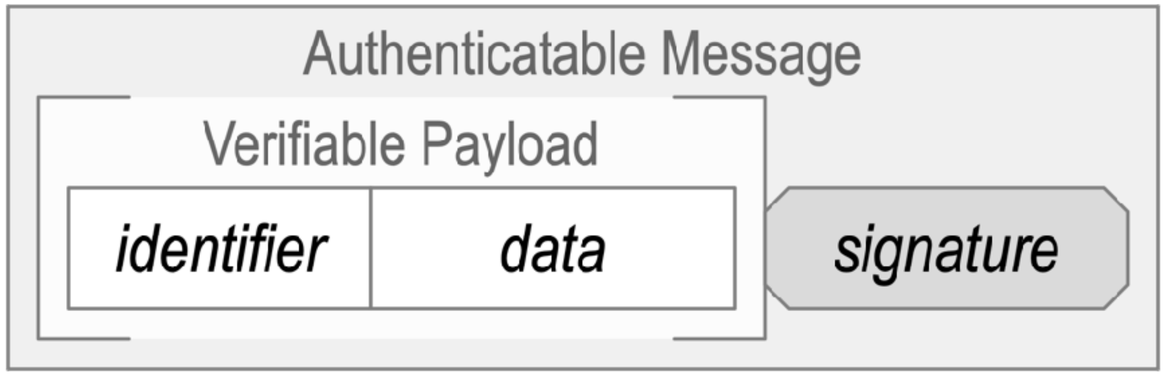
= Authentic data provenance by anyone to anyone from anyone

Solve data provenance to solve security



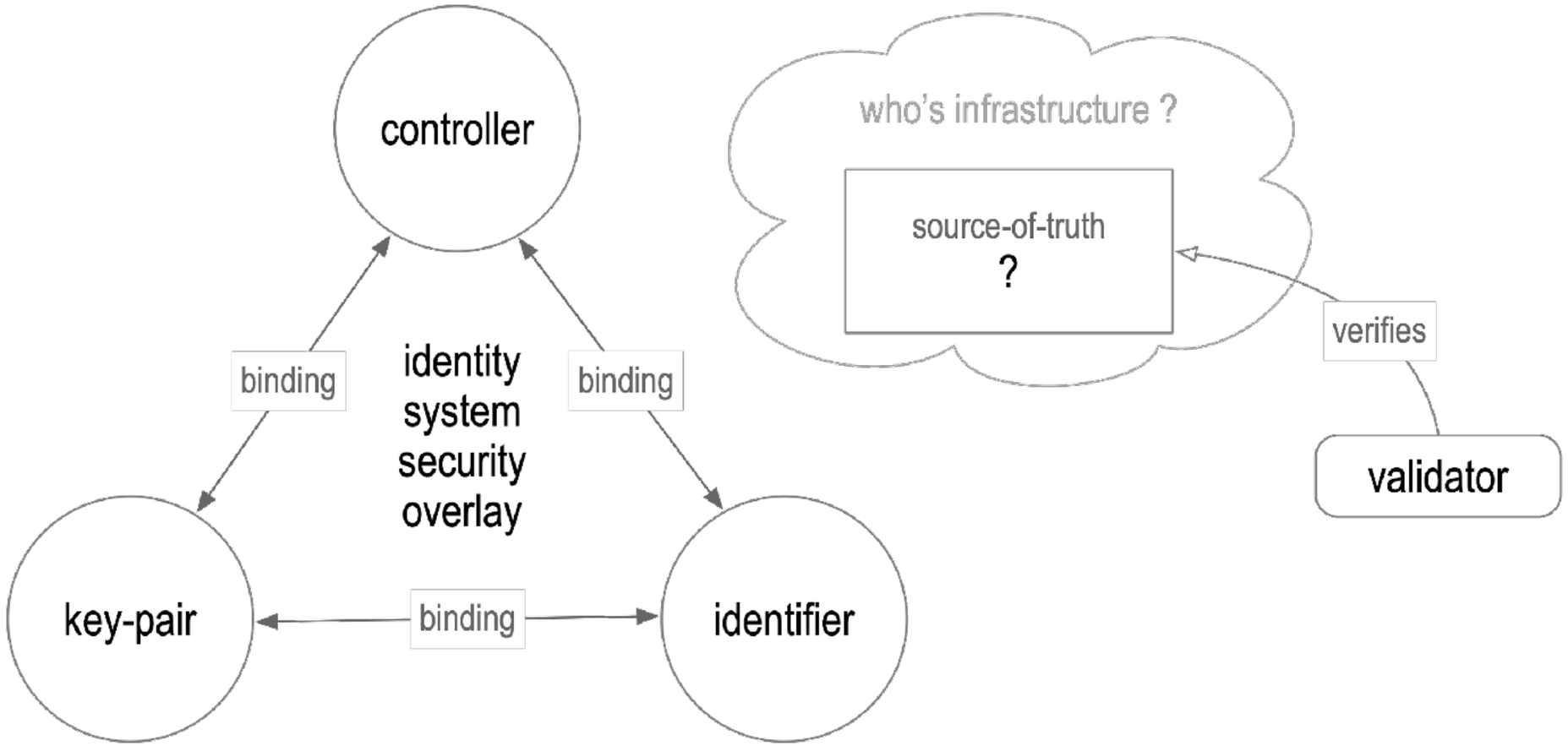
Identity (-ifier) System Security Overlay

Establish authenticity of IP packet's message payload.



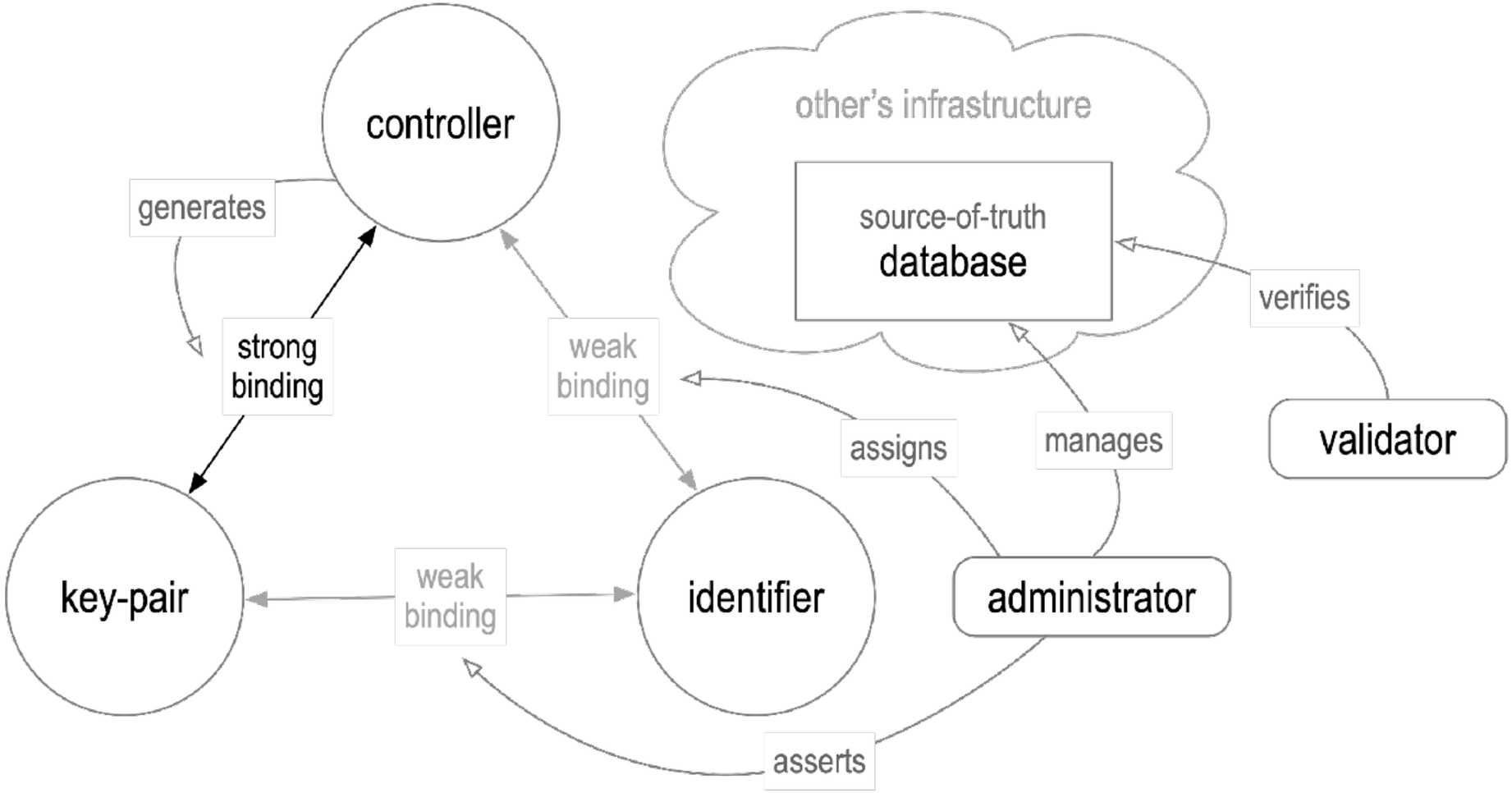
The overlay's security is contingent on the mapping's security.

Trust Basis of a Trust Domain



Administrative Trust Basis

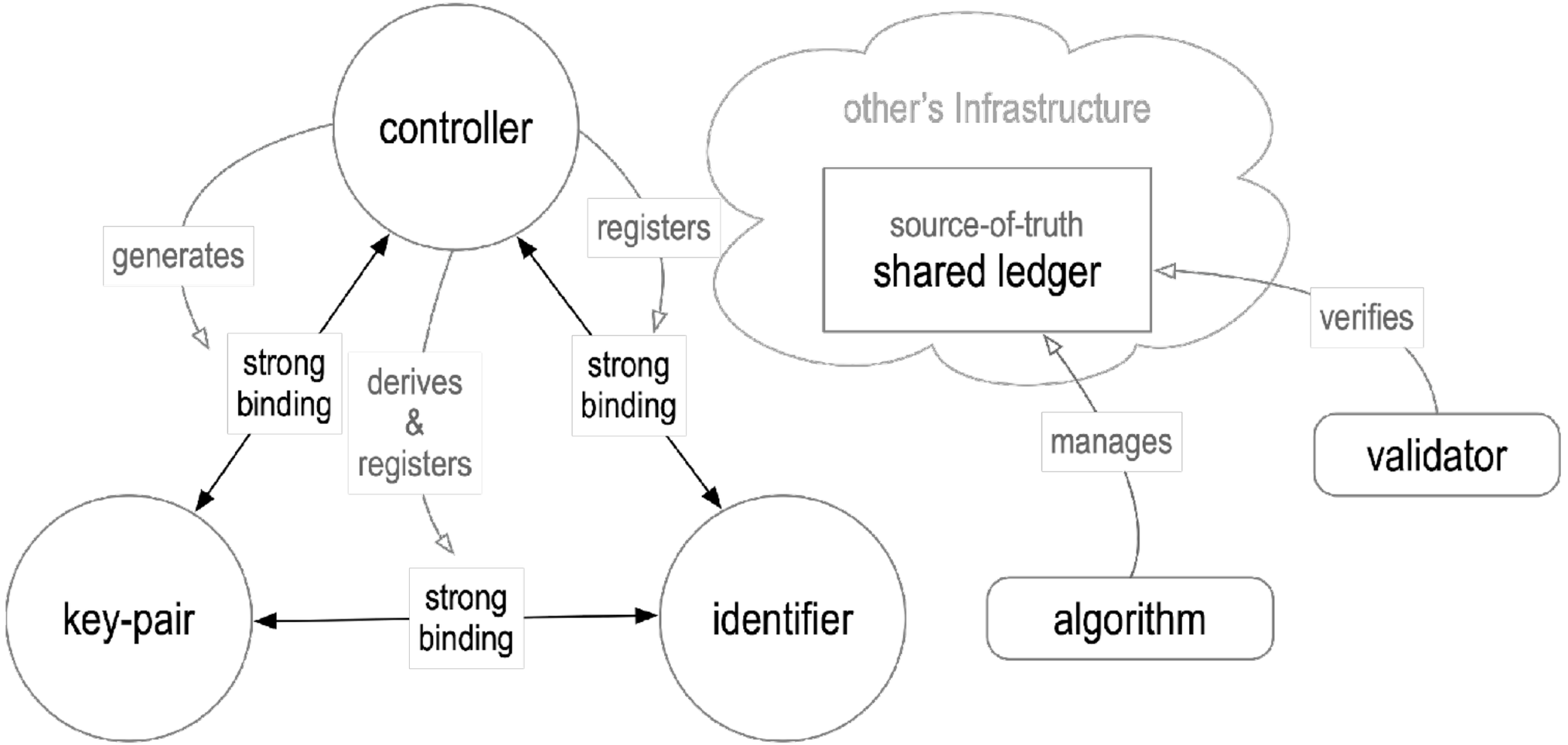
DNS/CA, OIDC IP



root-of-trust in non-verifiable operational infrastructure with opaque governance

Algorithmic Trust Basis

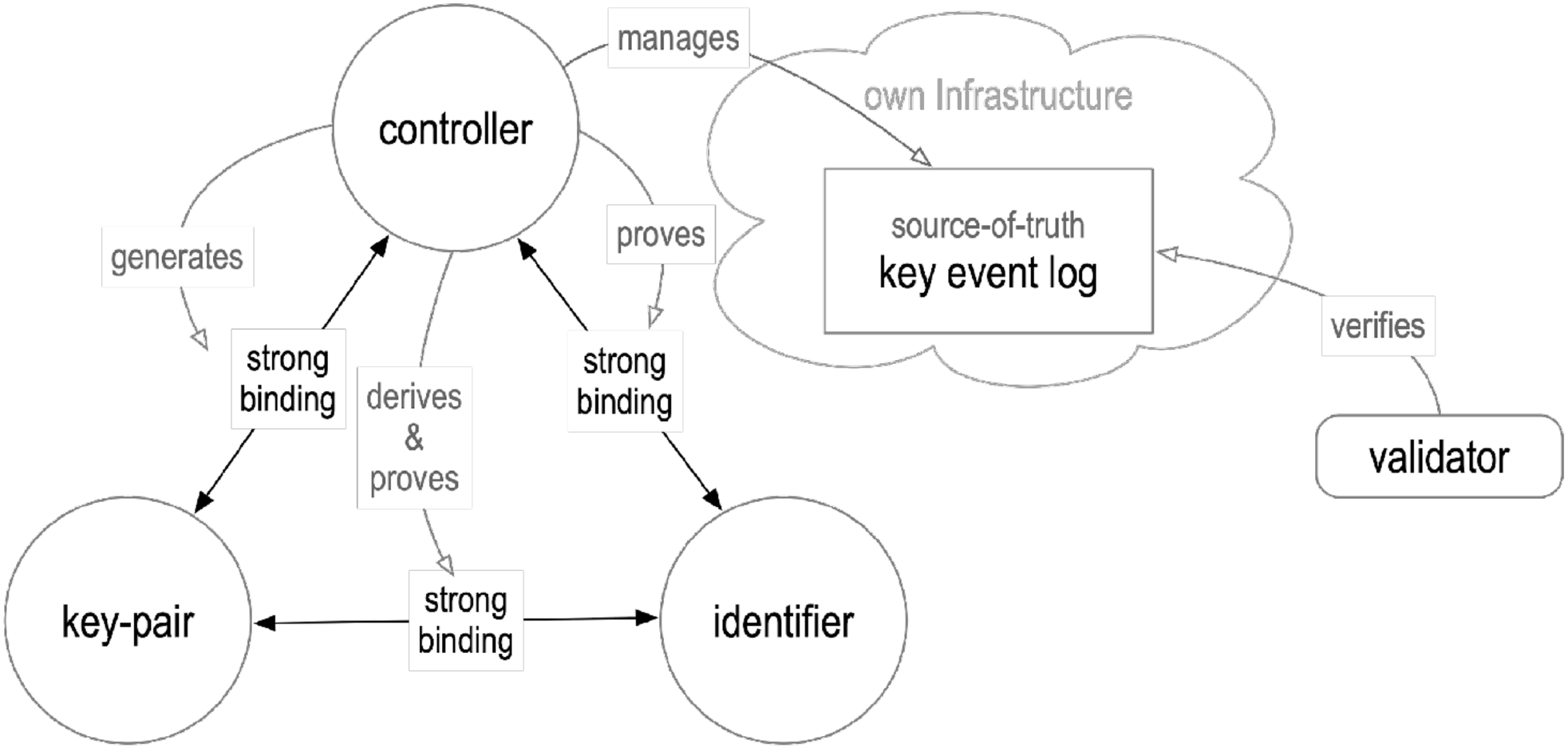
Shared distributed ledgers



root-of-trust in verifiable operational infrastructure with shared governance

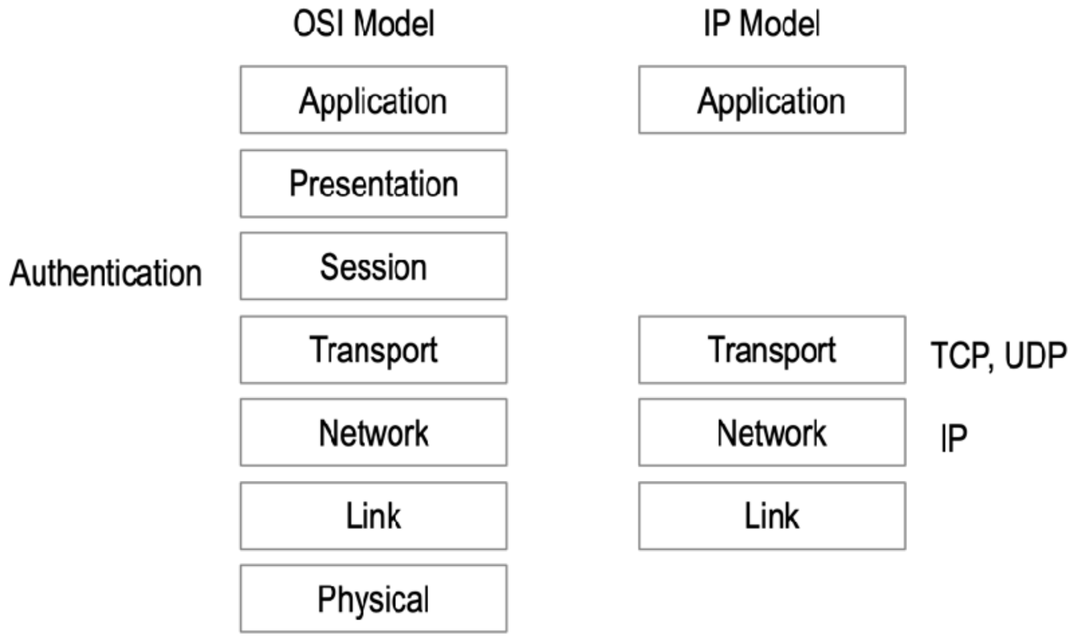
Autonomic Trust Basis

Cryptographic proofs via verifiable data structures



root-of-trust in verifiable cryptographic proofs of infrastructure with no shared governance

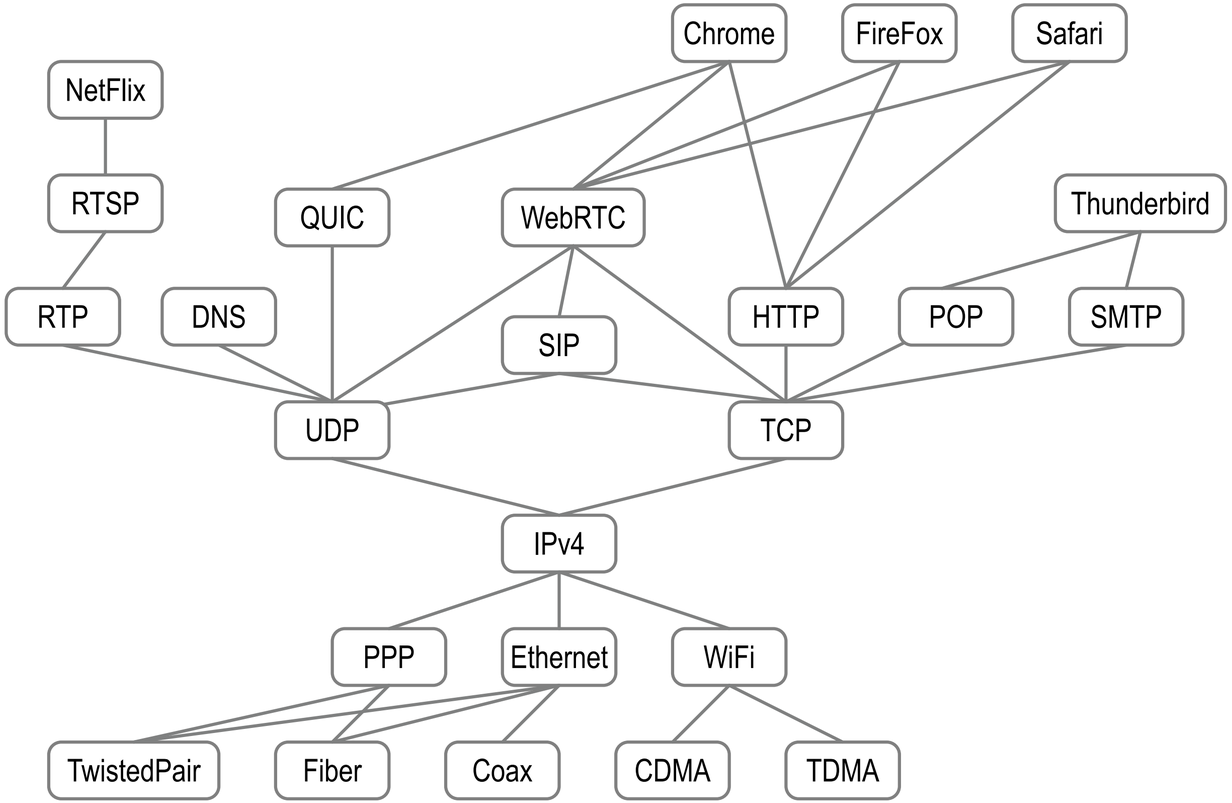
The Internet Protocol (IP) is *bro-ken*
because it has no *security (trust)* layer.

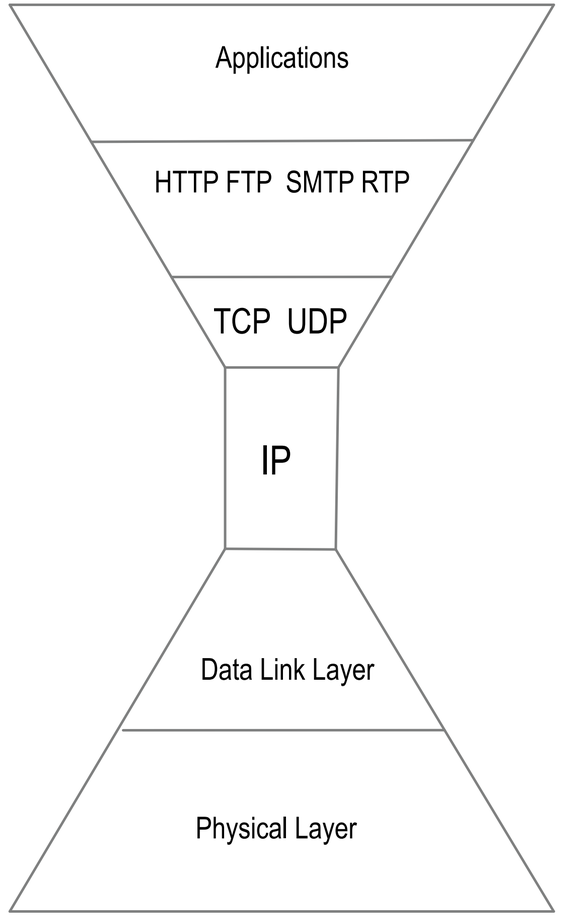


Instead …

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

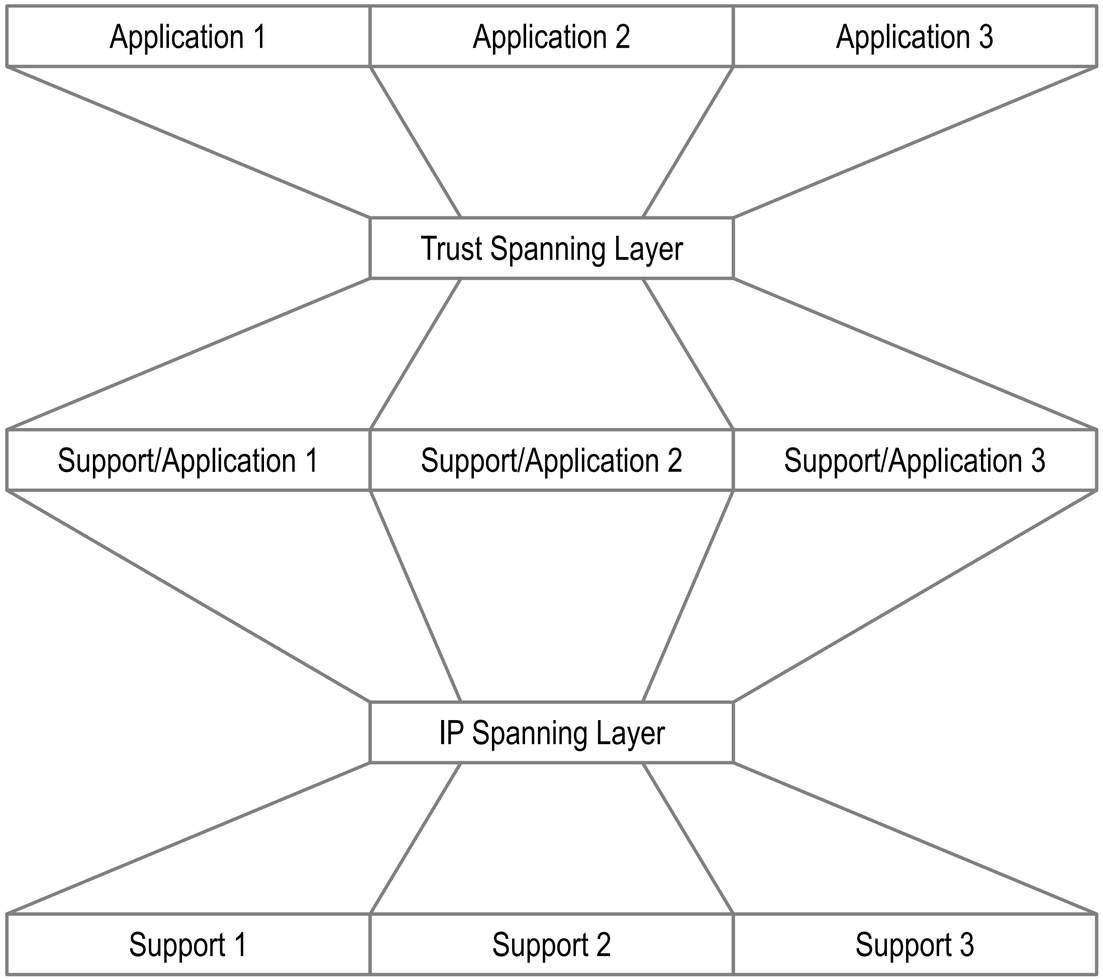
Spanning Layer

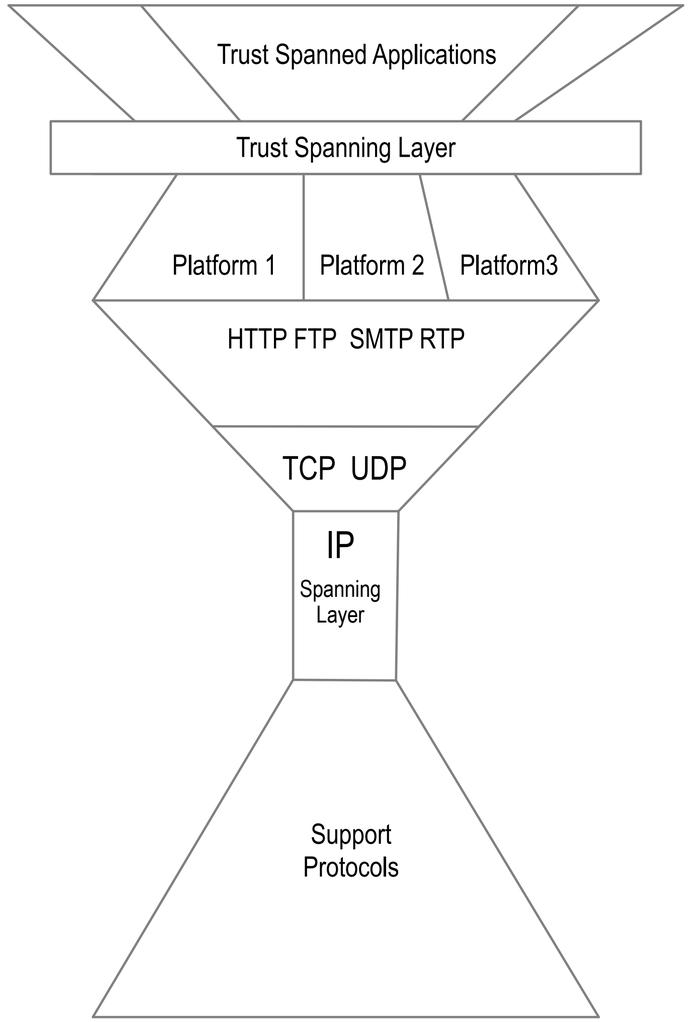




https://web.archive.org/web/20050415042854/http://www.csd.uch.gr/~hy490-05/lectures/Clark_interoperation.htm

Solution: Waist and Neck





Organizational Identity

Zero-trust architecture

Autonomic (cryptographic) decentralized root-of-trust (per organization)

Protocol not Platform

Delegable Authority

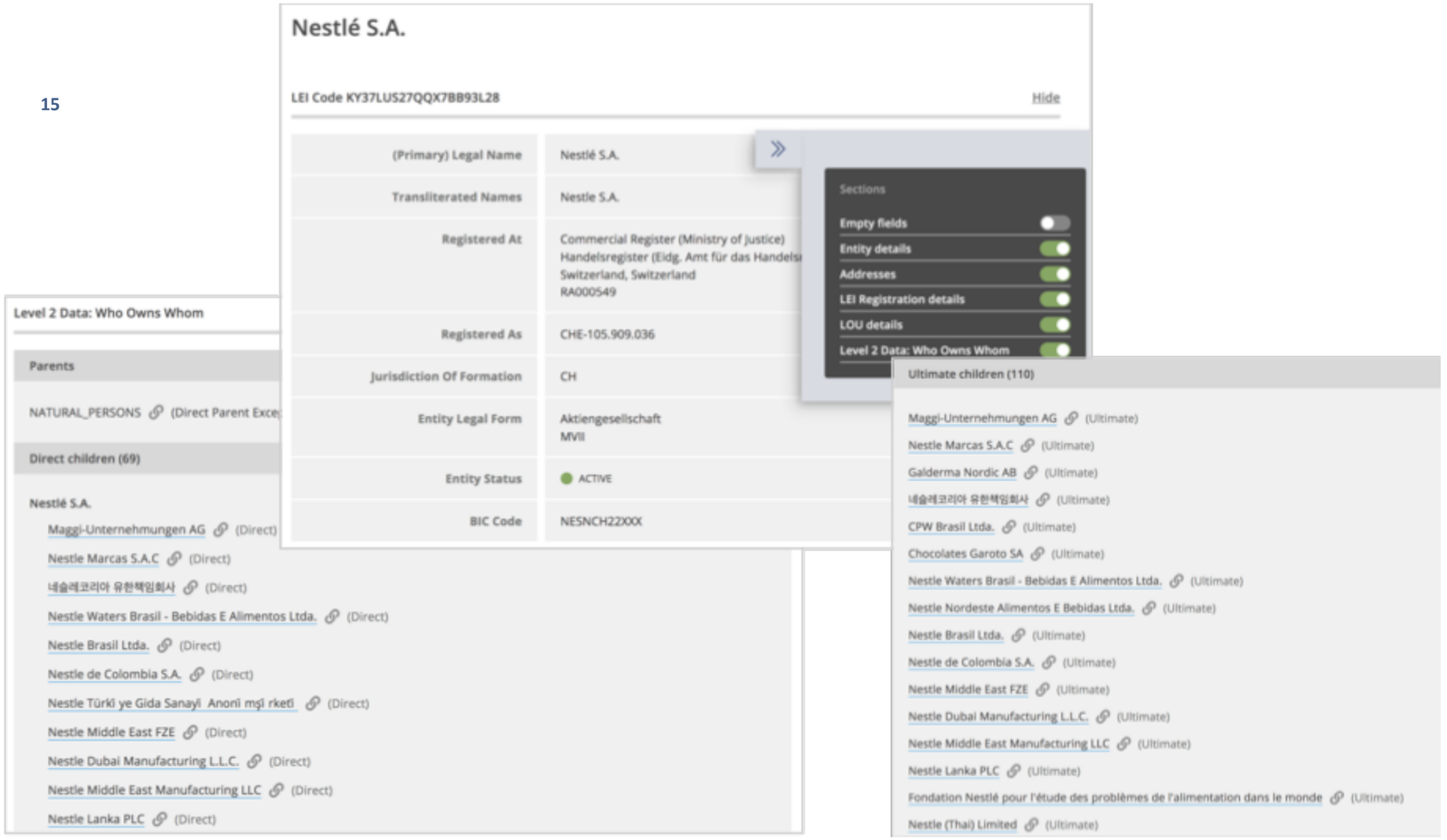
Multi-sig DPKI

Authentic Chained Data Containers

The Legal Entity Identifier – the LEI

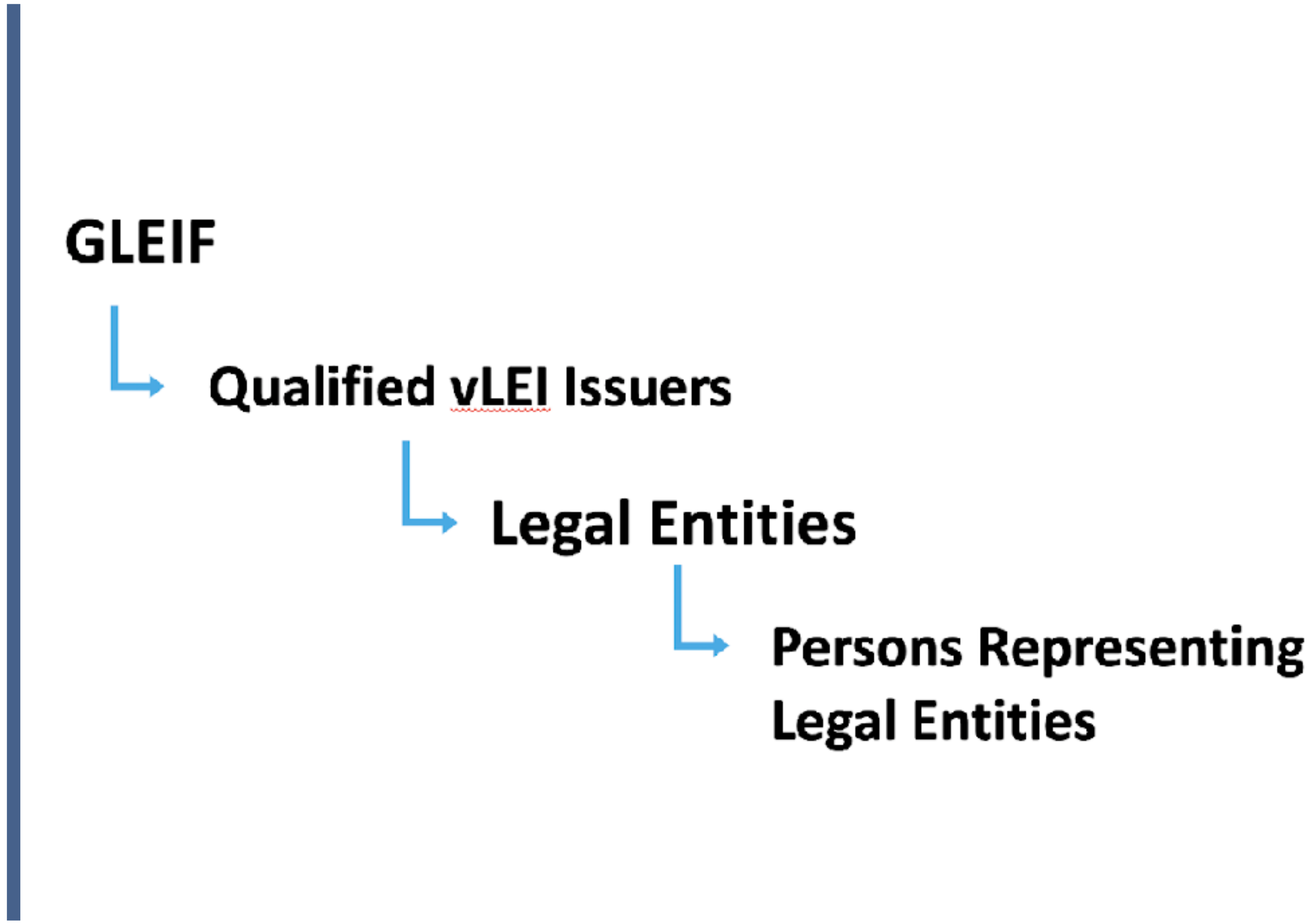
- The LEI is a life-long code **owned** by the respective legal entity.
- It points to the associated reference data.
- The LEI is an ISO standard ISO 17442

15



The LEI as a Verifiable Credential – the vLEI Trust Chain

- Every verifiable LEI (vLEI) is created by an **issuer**
- The issuer **cryptographically** signs the credential with its private key
- An issuer is the organization or entity that asserts information about a **subject** to which a credential is issued
- The vLEI Issuer is an organization **qualified** by GLEIF as part of a trusted network of partners
- GLEIF issues vLEIs to Qualified vLEI Issuers as attestation of trust.
- GLEIF is the Root of Trust



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.

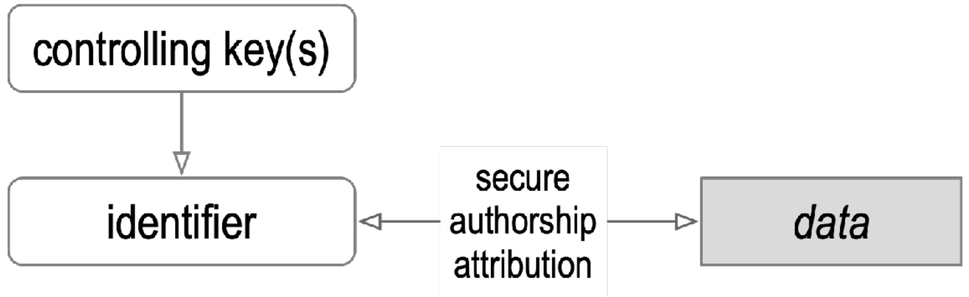
Then: Weak Crypto

Now: Strong crypto: ECC signing & ECC asymmetric encryption.

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

Now: decentralized zero-trust architecture for identity (KERI).

Flaw of PKI (DNS/CA)



Conventional PKI uses signed assertions (x509 certs) made by trusted entities to bind key state (public, private) key pairs to identifiers.
Use of private keys for either signing or decryption exposes them to side-channel attack.

Over-time, exposure makes private keys weak.

Thus, from time-to-time one must therefore revoke and replace, i.e. rotate the controlling private keys for a given identifier

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

This breaks the chain-of-trust-of-control over the identifier

What is KERI? (Key Event Receipt Infrastructure)
Decentralized Key Management Infrastructure (DKMI)
Decentralized Public Key Infrastructure (DPKI)

KERI fixes the security flaw (authenticity) in PKI (Public Key Infrastructure):

That flaw is key rotation.

In conventional PKI there is no cryptographic binding between one set of keys and the next.

KERI solves the key rotation problem for control over an identifier via pre-rotation which binds the next key-state to the prior key-state.

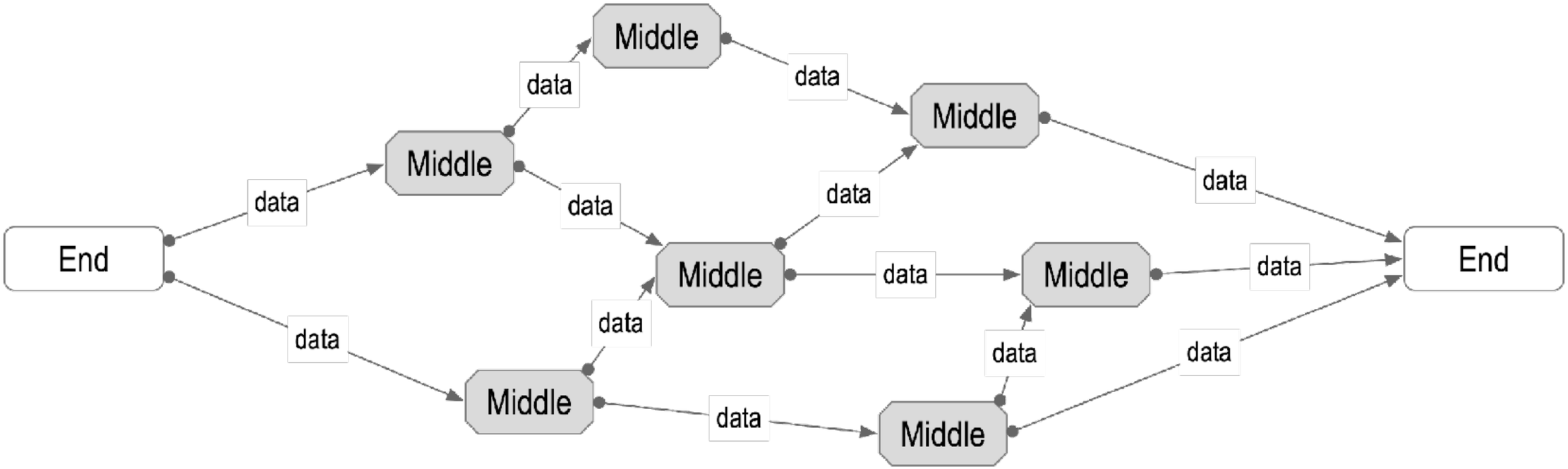
With KERI, key state is cryptographically verifiably bound to a class of self-certifying identifiers that use portable verifiable data structures called *key event logs* (KELs) to provide duplicity evident proof of the controlling key state.

With KERI every statement associated with a KERI identifier may be non-repudiably and securely attributed to the controller of the identifier via a signature made with keys given by cryptographically verifiable key state.

KERI solves the *secure attribution* problem with zero trust.

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

It's much easier to protect one's private keys than to protect everyone else's internet infrastructure

Zero-Trust Architectures & Computing

Never trust, always verify.

Perimeter-less security model.

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

No such thing as true zero-trust.

All architectures lie on the zero-trust spectrum given by:

the ratio of trusted surface to verifiable surface.

End goal = all data has end-to-end verifiable authenticity (provenance)

Resources:

NIST: Developing a Framework to Improve Critical Infrastructure Cybersecurity 04/08/2013 Zero Trust Model for Information Security, Forrester Research. http://csrc.nist.gov/cyberframework/rfi_comments/040813_forrester_research.pdf

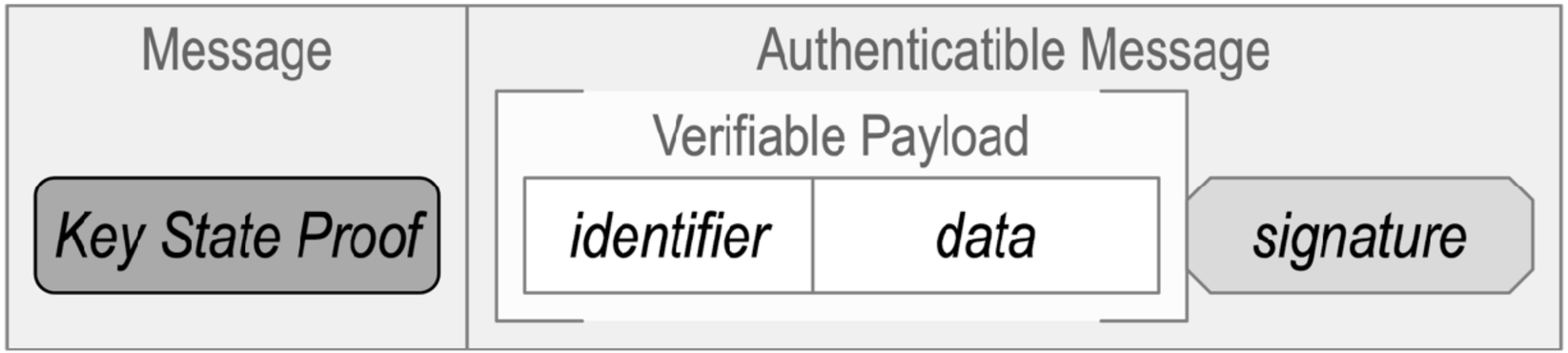
https://www.nist.gov/cyberframework Zero Trust Networks 2017 Gilman & Barth https://www.amazon.com/Zero-Trust-Networks-Building-Untrusted/dp/1491962194/ref=sr_1_1?

s=books&ie=UTF8&qid=1499871379&sr=1-1&keywords=zero+trust+networks

Identity (-ifier) System Security Overlay

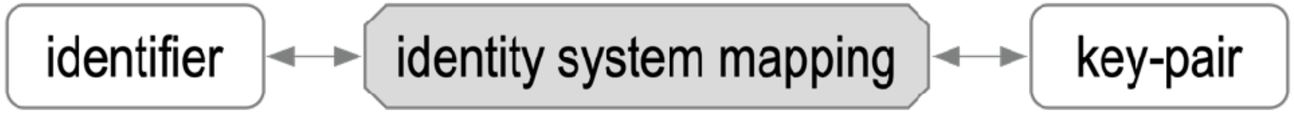
persistent mapping via verifiable data structure of key state changes

Establish authenticity of IP packet's message payload.

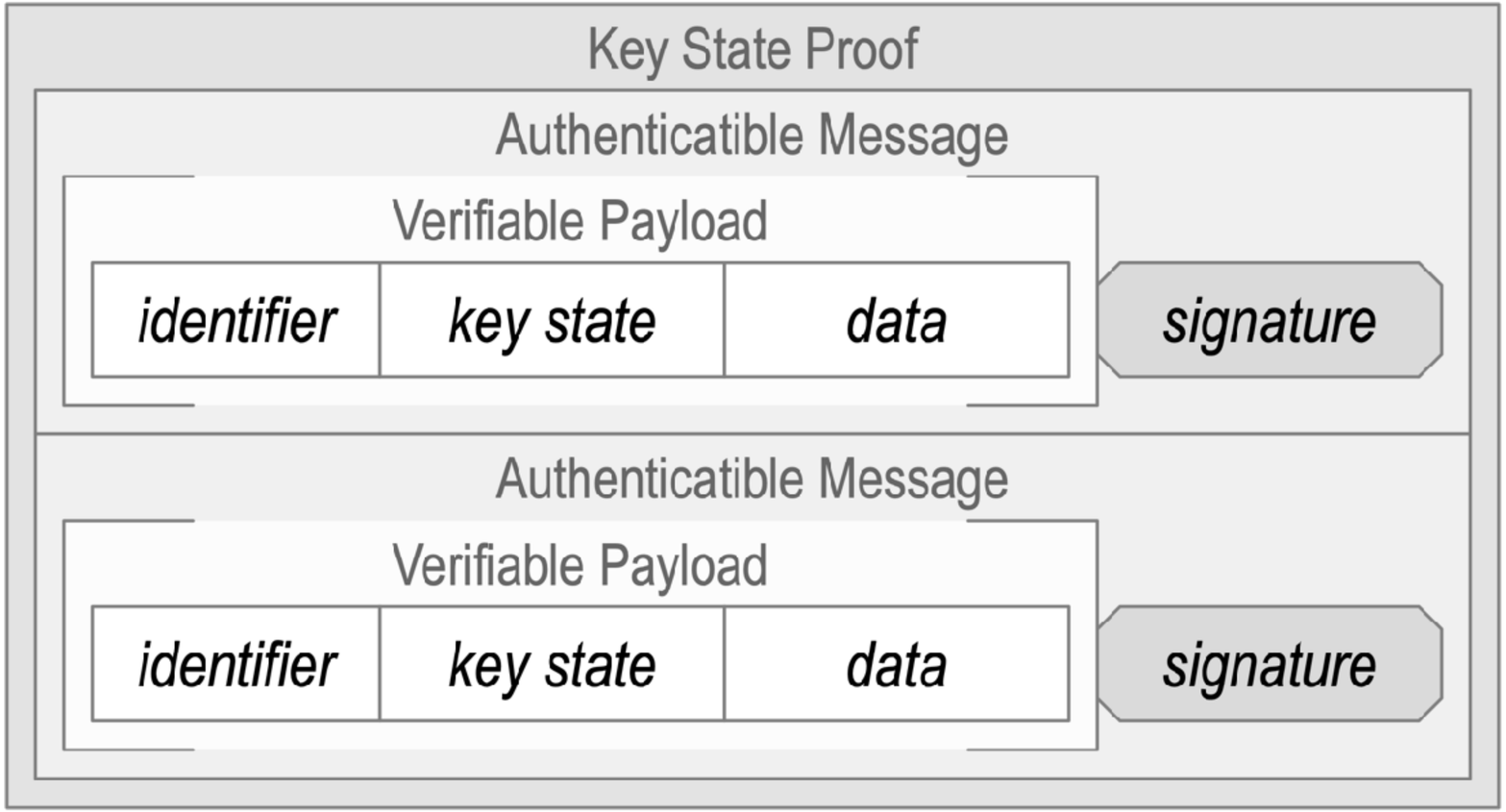


The overlay's security is contingent on the mapping's security.

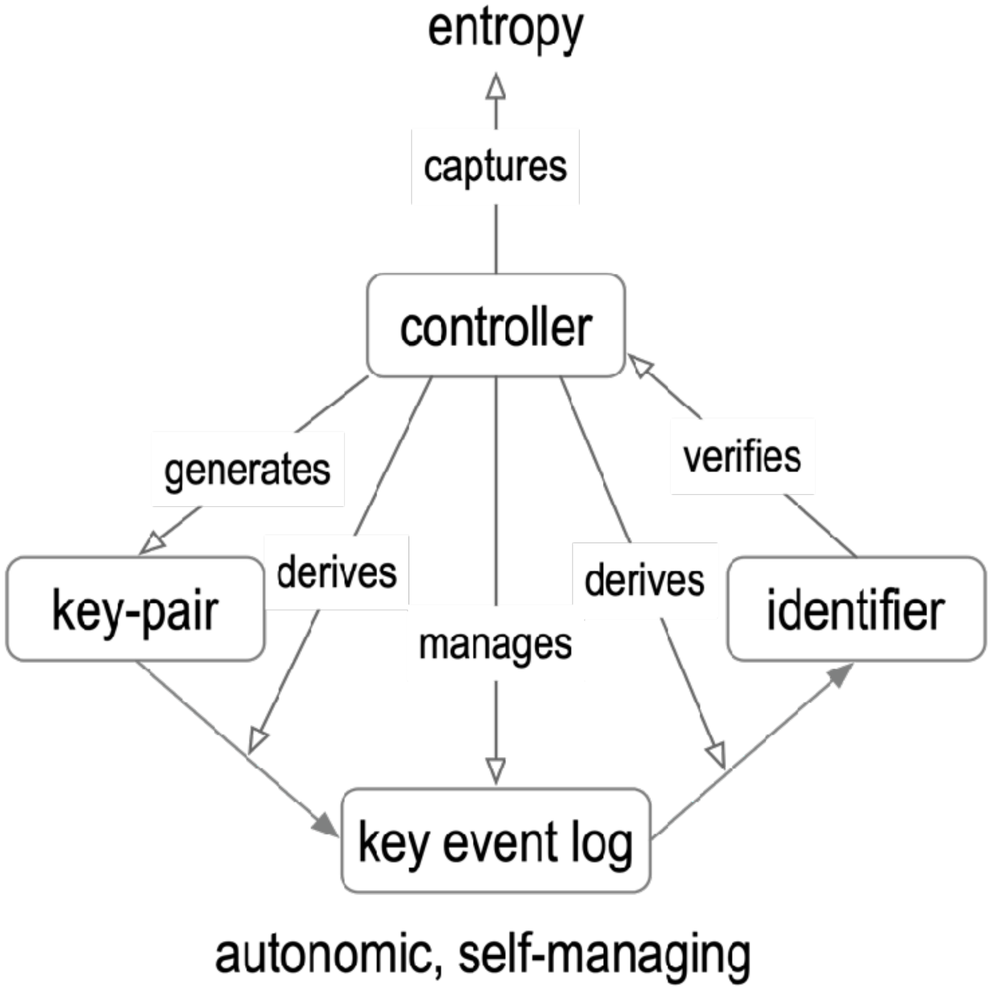
Key State Proof is Recursive Application of Overlay



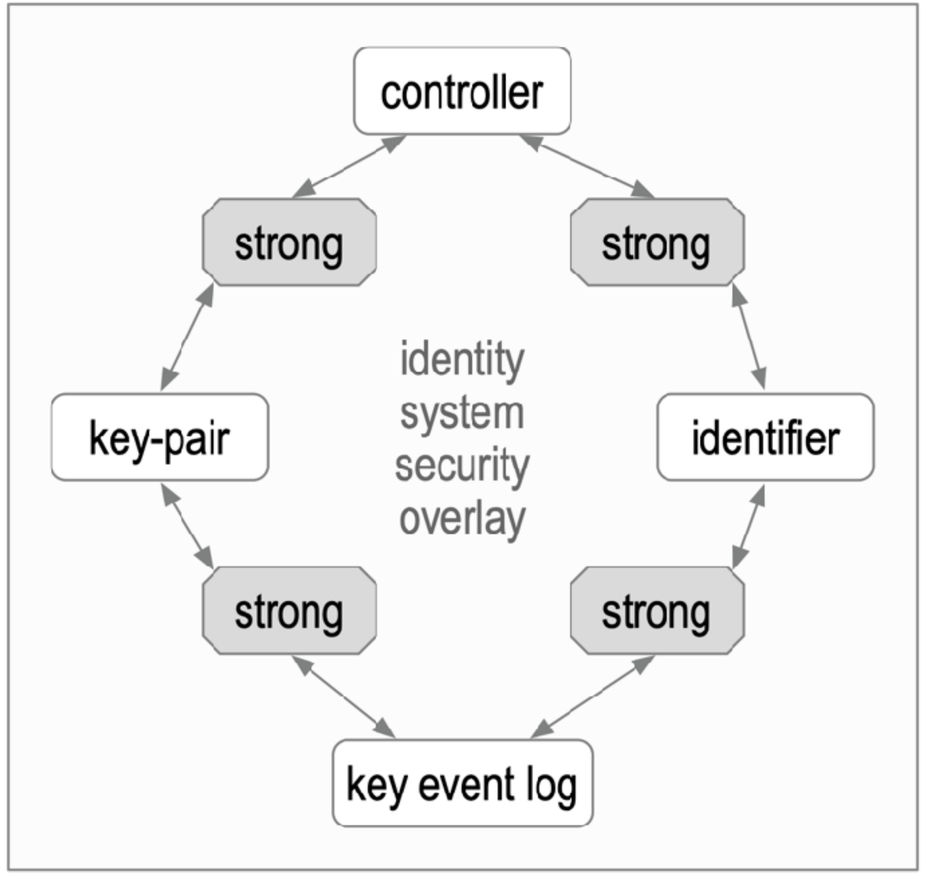
Persistent mapping via verifiable data structure of key state changes



Autonomic Identifiers (AIDs): (type of self-certifying identifier) Issuance and Binding



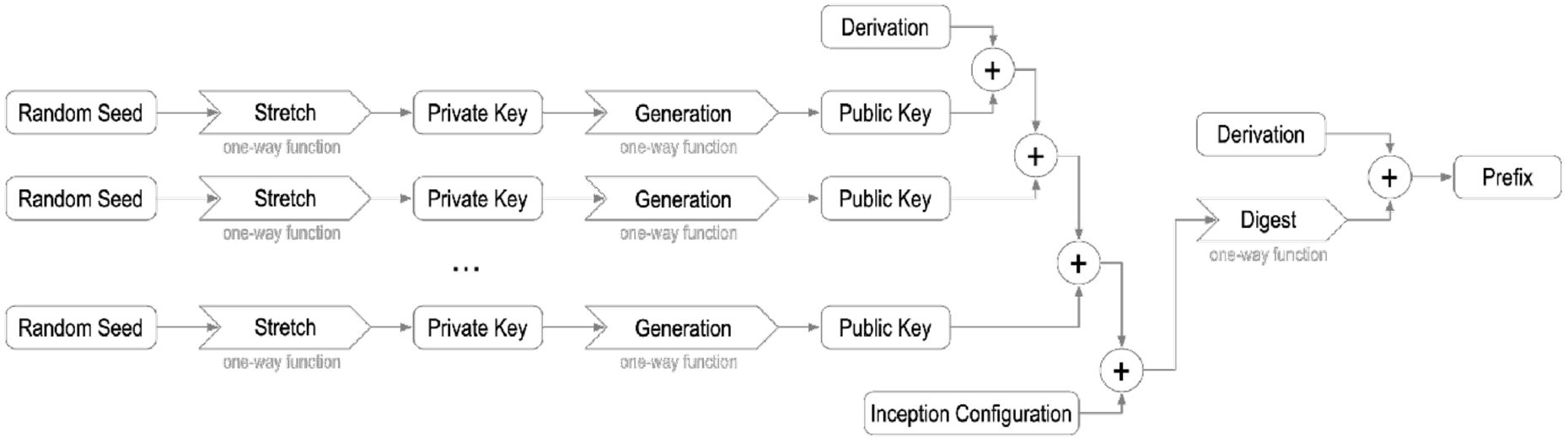
autonomic, self-managing

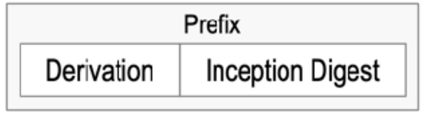


Autonomic Identifier Issuance Tetrad

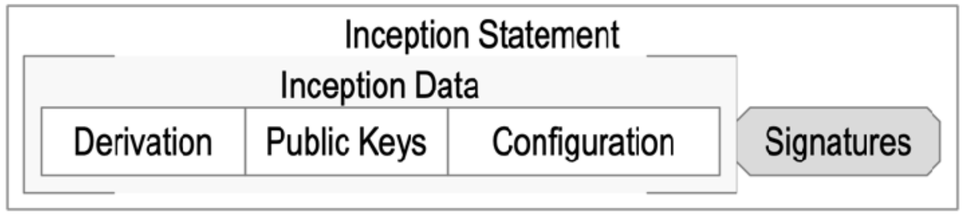
cryptographic root-of-trust with verifiable persistent control

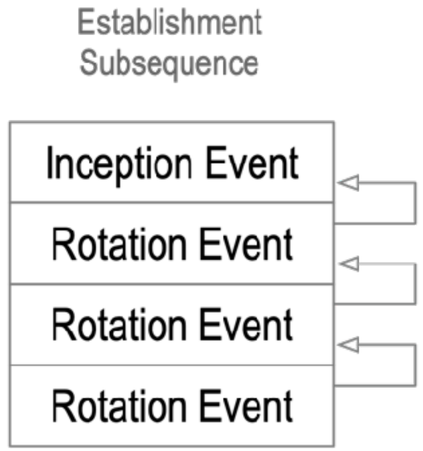
Cryptographic Root-of-Trust:
Self-Certifying Identifier (SCID) + Key Event Log = Autonomic Identifier (AID)





Key Event Log



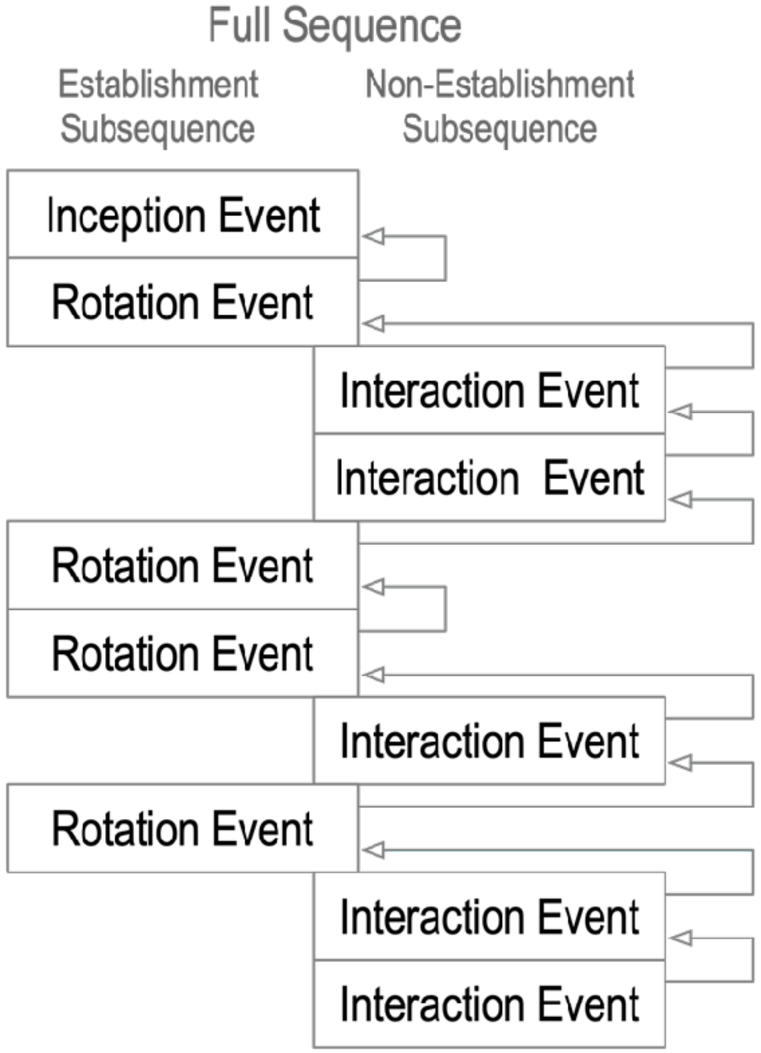


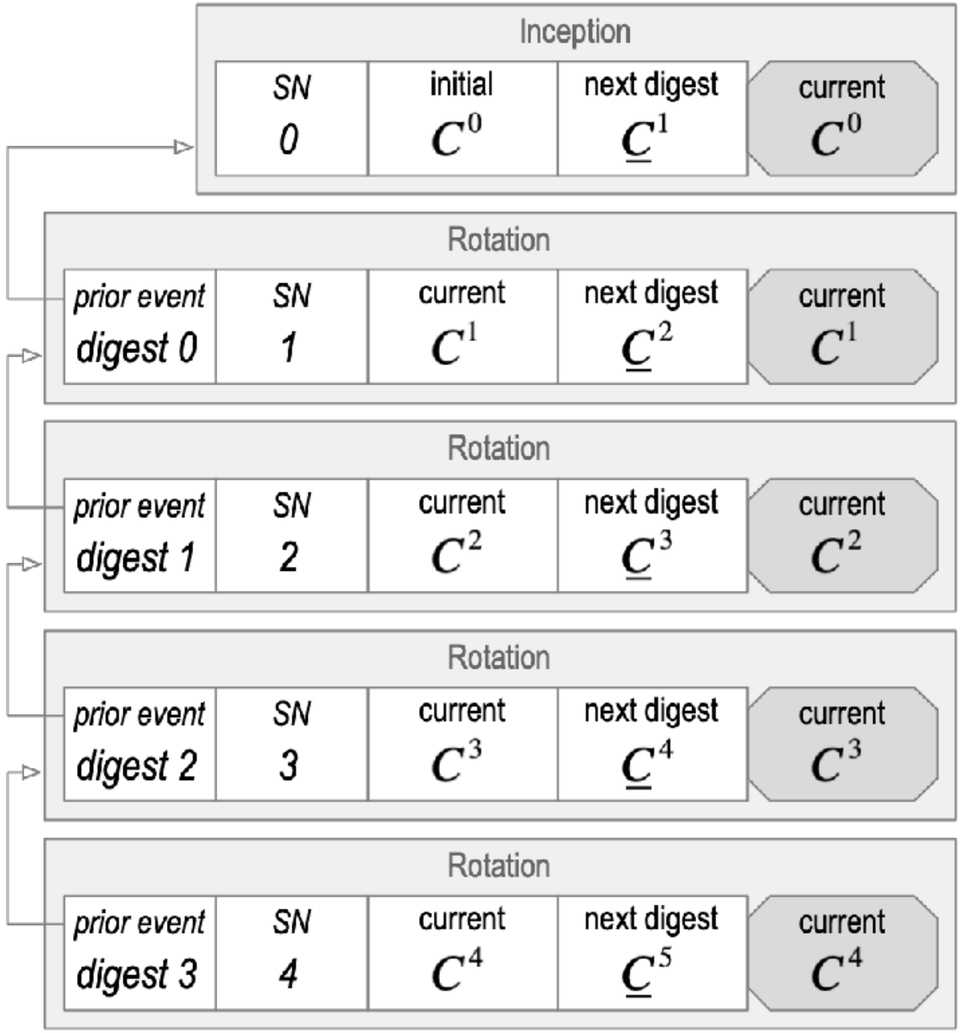
`EXq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148`

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

Solution: Key Pre-Rotation

duplicity evident
verifiable data
structure

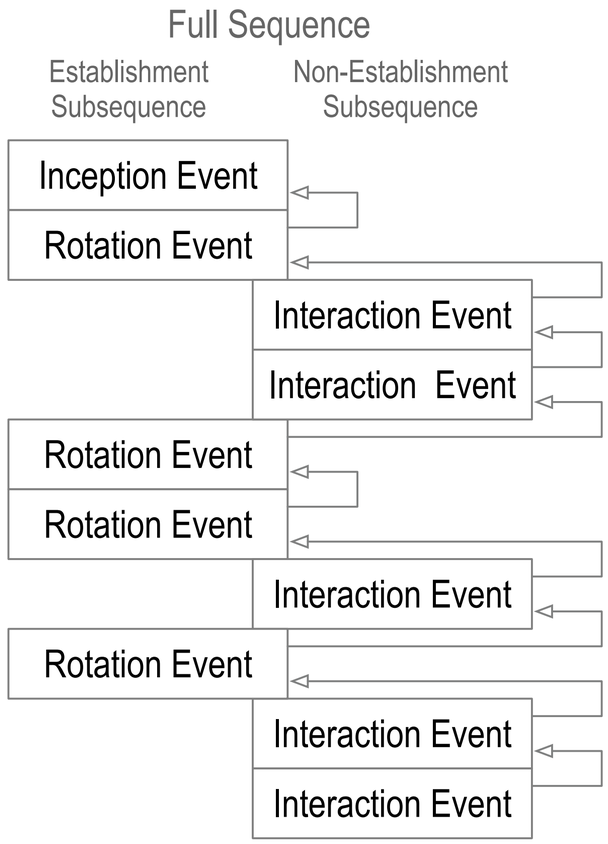




Digest of *next* key(s) makes pre-rotation post-quantum secure

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

Inconsistency and Duplicity



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

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

Internal vs. External Inconsistency

Internally inconsistent log = not verifiable.

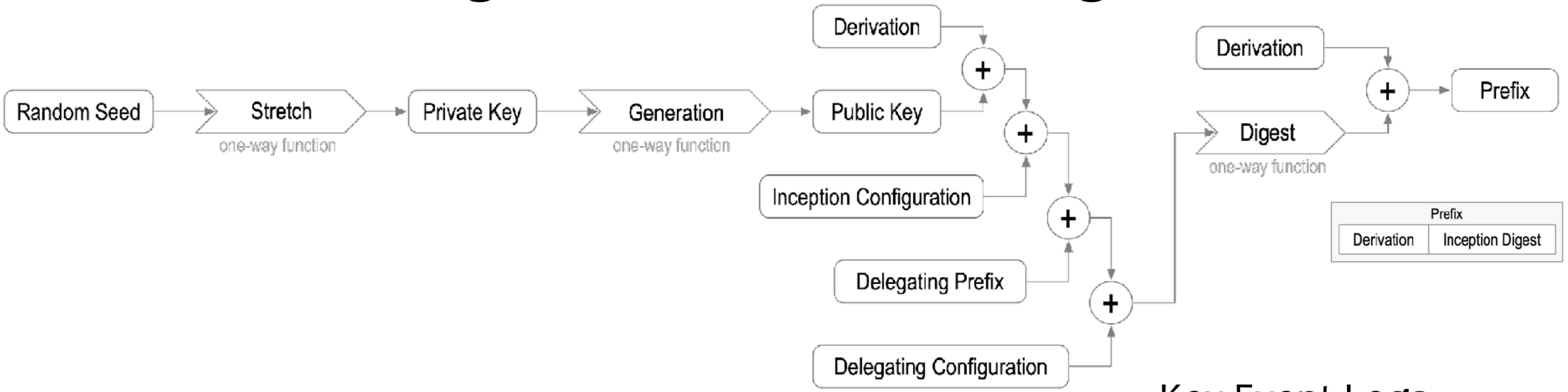
Log verification from self-certifying root-of-trust protects against internal inconsistency.

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

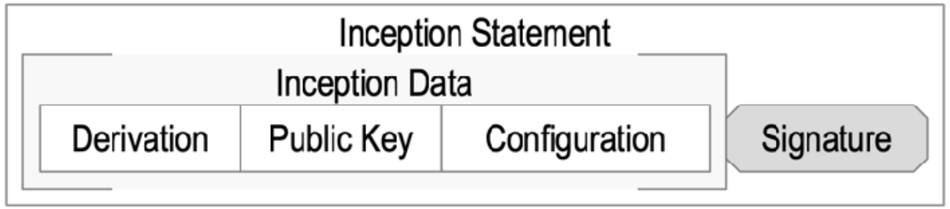
Duplicity detection protects against external inconsistency.

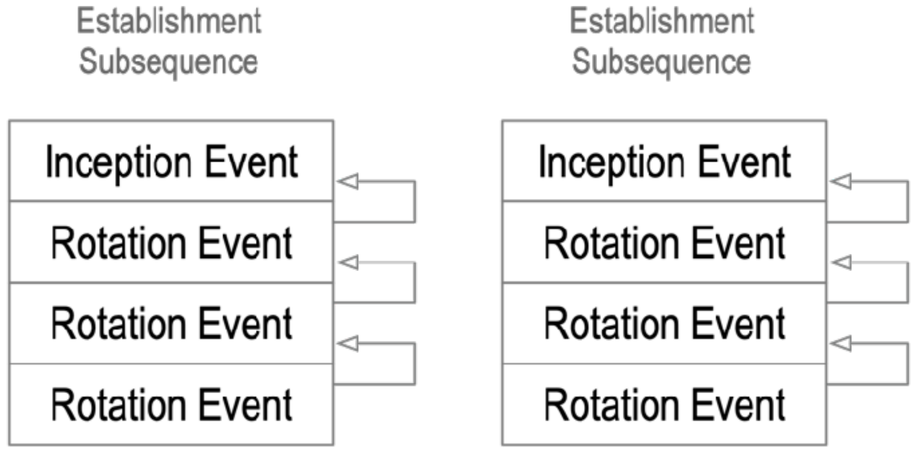
KERI provides duplicity evident DKMI

Delegated Self-Addressing SCID



Key Event Logs





`EXq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148`

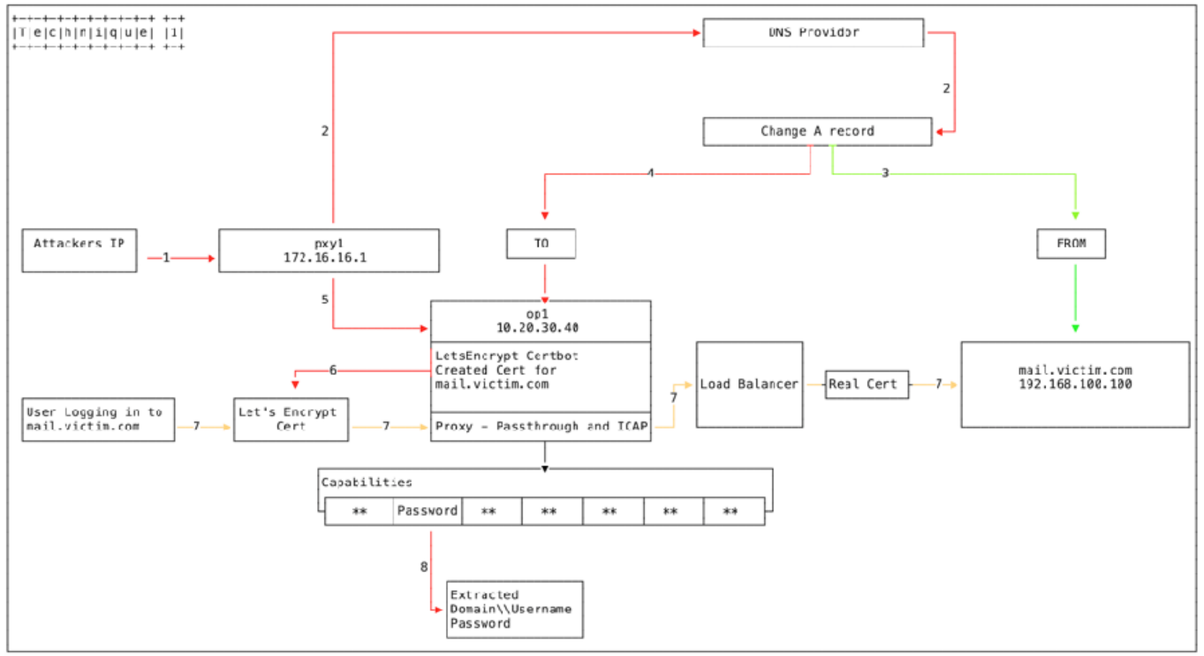
`did:keri:EXq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148/path/to/resource?name=sec#yes`

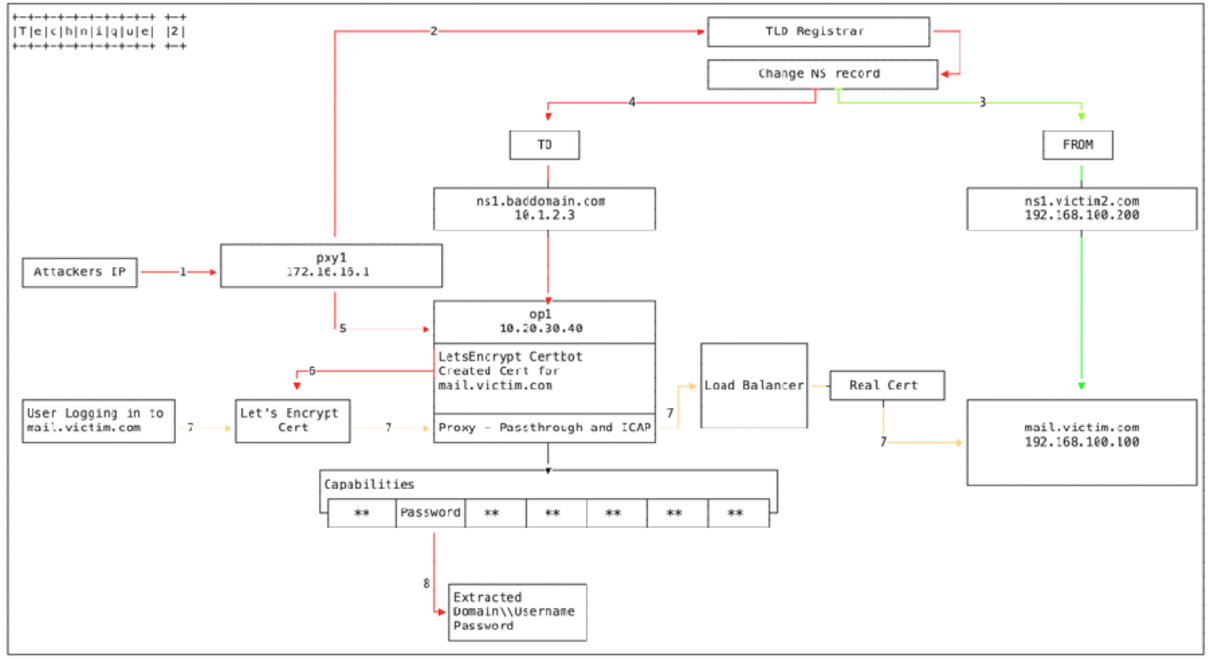
Backup Slides

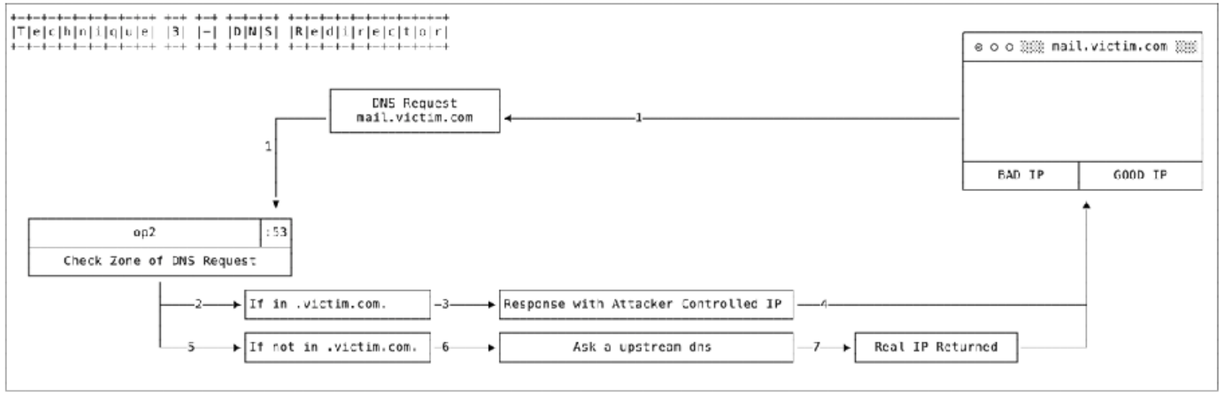
DNS Hijacking

A DNS hijacking is occurring at an unprecedented scale. Clever tricks allows attackers to obtain valid TLS certificate for hijacked domains.

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



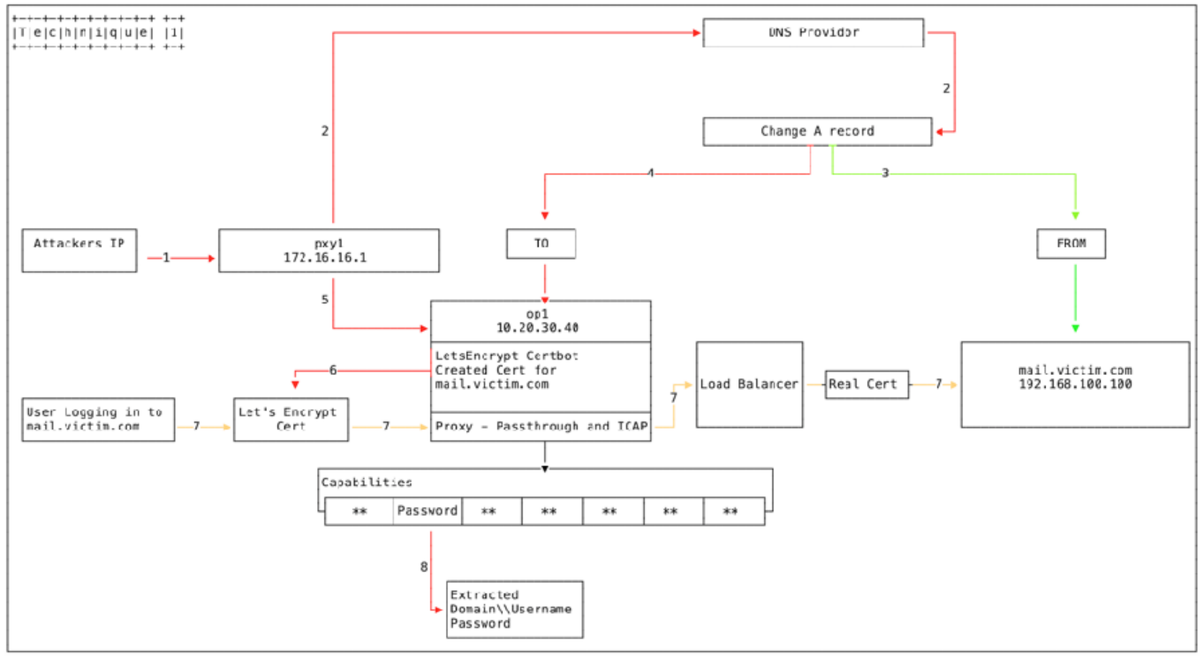


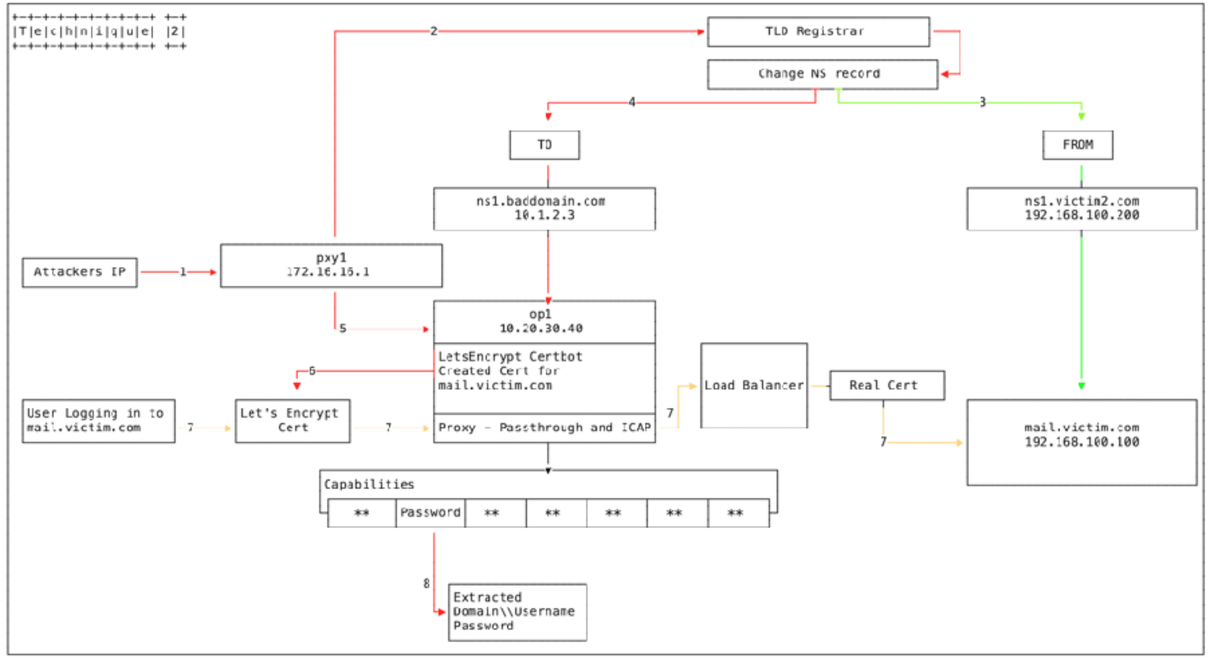


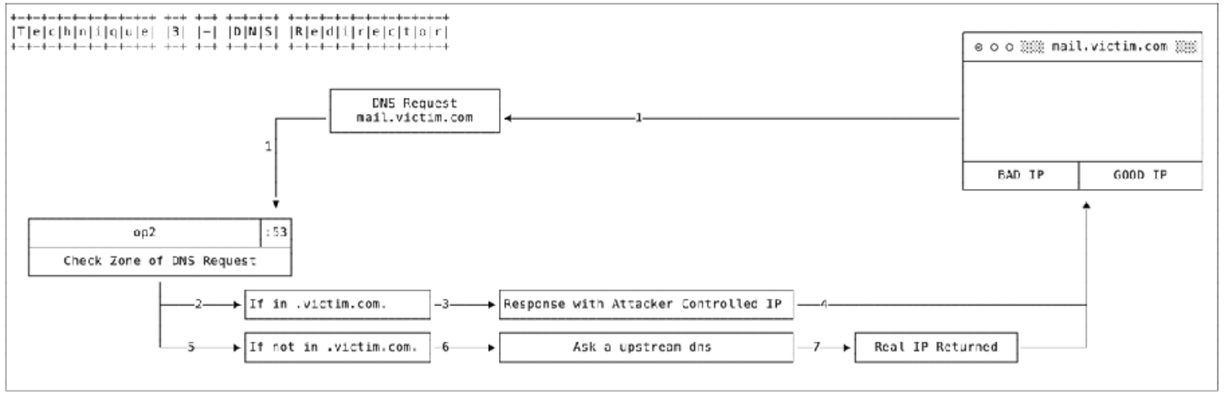
DNS Hijacking

A DNS hijacking is occurring at an unprecedented scale. Clever tricks allows attackers to obtain valid TLS certificate for hijacked domains.

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







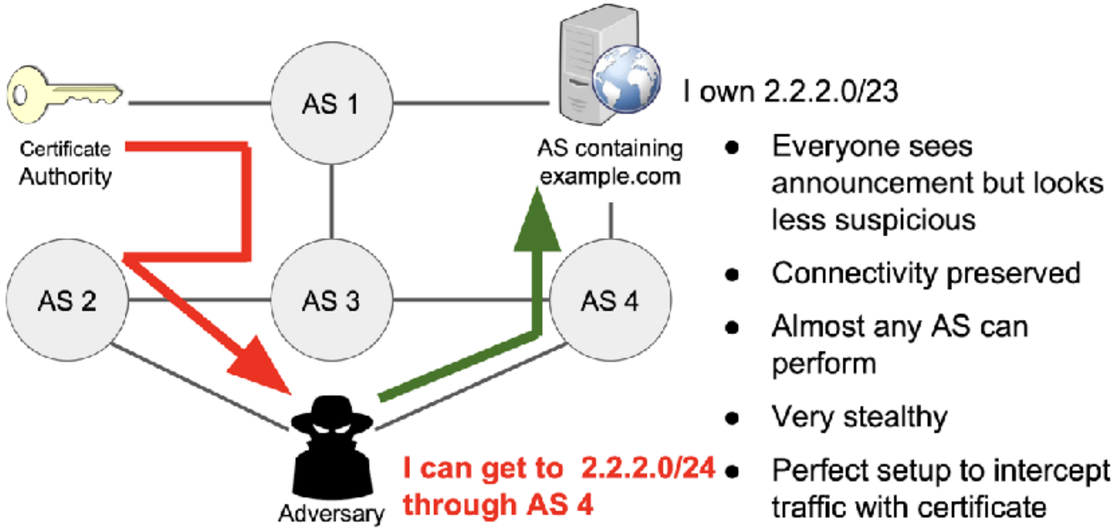
BGP Hijacking: AS Path Poisoning

Spoof domain verification process from CA. Allows 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

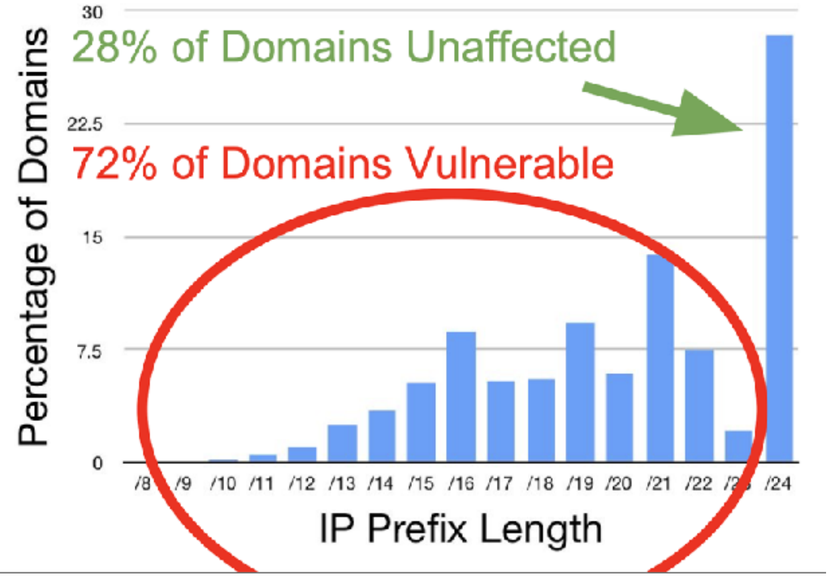


I own 2.2.2.0/23

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

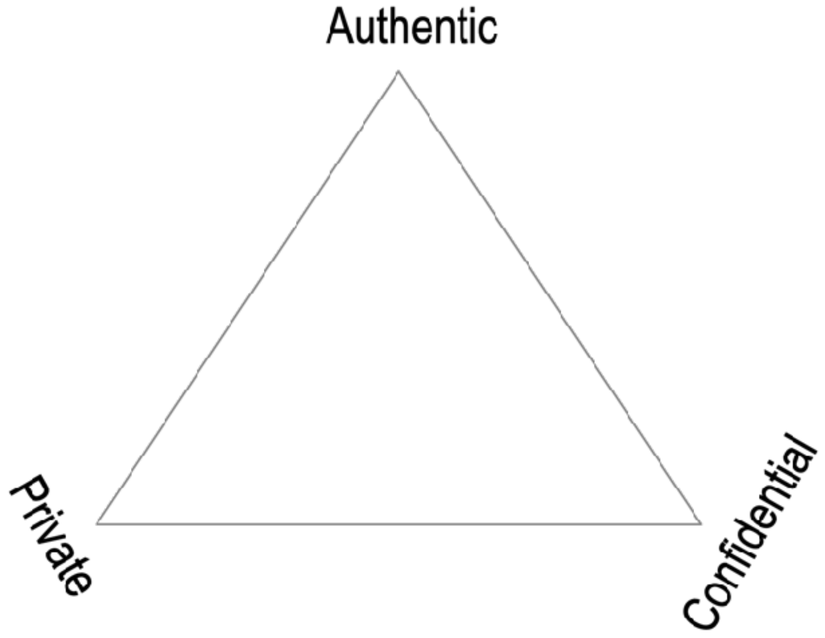
Vulnerability of domains: sub-prefix attacks

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



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!

Proving Authenticity

Non-repudiable Proof:

a statement's author cannot successfully dispute its authorship
Asymmetric key-pair digital signature

Repudiable Proof:

a statement's author can successfully dispute its authorship
DH shared symmetric key-pair encryption (auth crypt)
Shared secret makes every verifier a potential forger

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)

Trust Domain

A *primary* root-of-trust is *irreplaceable*.

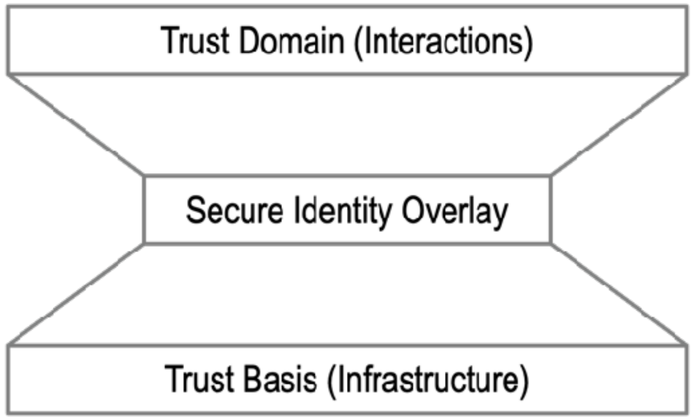
A *secondary* root-of-trust is *replaceable*.

A *trust basis* binds controllers, identifiers, and key-pairs.

A *trust domain* is the ecosystem of interactions (functions) that rely on a trust basis.

The hard problem is cross-domain value transfer.
The solution is transitive trust.

A *secure identity overlay* maps the *trust basis* to the *trust domain.*



Control over Trust Bases and Domains

Want decentralized control over trust domains or at least the trust bases.

Shared control over a trust domain is less decentralized than non-shared identifier specific control over a trust domain.

A shared primary (non-replaceable) root-of-trust aka shared ledger is the trust basis for one trust domain.

It has shared governance which is more decentralized but it’s not solving the hard problem of moving value between trust domains under different control.