SPAC (Secure Privacy, Authenticity, and Confidentiality)

Privacy Given Strongest Authenticity and Confidentiality

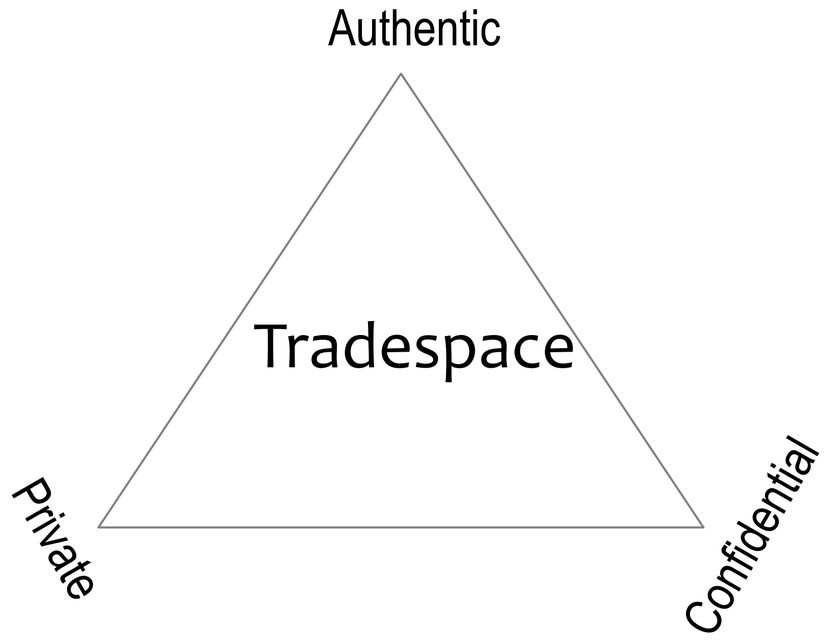
Samuel M. Smith Ph.D.
sam@keri.one
https://keri.one
2023/02/08

Whitepaper

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/SPAC_Message.md

PAC Theorem

One can have any two of the three (privacy, authenticity, confidentiality) at the highest level but not all three.



Trade-offs required!

ToIP Design Goals

Establishing trust between parties requires that each party develop confidence in the following properties of their relationship:

Authenticity

Is the receiver of a communication able to verify that it originated from the sender and has not been tampered with?

Confidentiality

Are the contents of a communication protected so that only authorized parties 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 our design goal with the ToIP stack is to achieve these three properties in the order listed.

The ToIP design goals provide an order of importance. The design goals indicate that one should start with high authenticity, then high confidentiality, and then as high as possible privacy, given there is no trade-off with respect to the other two.

https://github.com/trustoverip/TechArch/blob/main/spec.md#61-design-goals

Practical Elaborations

Authentic and Authenticity

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

Authenticity is primarily about control over the key state needed to prove who said what in the conversation via digital signatures. In other words, it is about secure attribution.

Confidential and Confidentiality

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

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

Privacy is primarily about control over the disclosure of *who* participated in the conversation (non-content meta-data = identifiers). More specifically, privacy is about managing exploitably correlatable identifiers, not merely identifiers of *who* but also any other type of identifier in non-content metadata, including conversation (*interaction*) identifiers.

Privacy and Confidentiality

There are also two fundamentally different but complementary definitions of privacy that are relevant to protocol design.

- The first (ToIP) definition is about the recipient respecting the data privacy concerns of the sender with respect to the content disclosed to the receiver. The protection comes from the recipient protecting the data privacy rights of the sender once the recipient has received that data. This first type of privacy protection may be achieved independently of how private the communications channel was that resulted in the receiver obtaining the disclosure of the content.
- The second definition is about the correlatability of the publically viewable metadata included in the communication. This is called non-content meta-data by the surveillance community. This definition distinguishes between confidential data (meta-data or otherwise) that is not publicly viewable, where public means any third party not a party to the conversation. Typically confidentiality protection comes from encryption (i.e., cipher text vs. plain text). But an out-of-band (OOB) exchange of information is confidential (not viewable) by a third-party surveillor of an in-band (IB) channel, thus making the OOB exchange confidential without encryption.

Privacy and Confidentiality Legalese

The terms confidential and private also have distinct legal definitions.

The legal definition of confidentiality, however, is consistent with the surveillance definition above as well as the definition used by the ToIP properties above. Confidentiality law is about contractual agreements governing the disclosure of data by one party to another.

The legal definition of privacy, on the other hand, is not so consistent. The regulatory surveillance law definition is different from the privacy rights regulatory law definition. The former is about drawing the line between non-content metadata and content data in order to be classified as searchable or not (see https://www.lawfareblog.com/relative-vs-absolute-approaches-contentmetadata-line and https://www.pogo.org/analysis/2019/06/the-history-and-future-of-mass-metadata-surveillance/) and the latter is mainly about protecting personal data rights by data holders, controllers, and processors such as GDPR (see https://gdpr-info.eu).

Of interest is the fact that the former (confidentiality law) can be used better to protect the latter (personal data rights). For example, the well-known concept called [chain-link confidentiality](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2045818), circa 2012, details how one can protect the data privacy concerns of the sender via contractual confidentiality law. ACDCs, for example, use the well-known [Riccardian Contract](https://iang.org/papers/ricardian_contract.html) formalism, circa 1996, to support chain-link-confidential through what is called contractually protected disclosure.

Historical Basis

Historically, an address of any kind (including the original postal address) exists for two main purposes.

1. So that stuff can get routed (delivered) to that address as a destination via public roads.

2. So that legitimate parties can tax or bill efficiently other parties at any or all addresses.

In the electronic communication world, the same applies. Addresses need to be public so that communications can be routed to destinations over multiple disparately owned, controlled, and regulated communications infrastructures (largely public-regulated monopolies). Likewise, the main mechanism for billing for communication services rendered is address based. So addresses have to be public, not confidential, not only so that stuff can get routed to those addresses but also so they can be the destination for bills to pay for the communication itself. So pragmatically, the legal surveillance world respects that these two purposes (routing and billing) require identifiers (addresses) that must need be public and eaves-droppable.

This is merely a logical, practical extension of property rights w.r.t. unreasonable search. The right to be protected from search only extends to what is not viewable behind closed doors. Anything viewable from a public street is fair game with no protection. Likewise to some extent an automobile or delivery van. What's locked in the trunk or the back of the van (unless it's making noise hearable by a bystander) may be protected. Where it goes on a public street is not protected, and its license plate is a publically viewable address proxy.

As a result, both practically and legally, all routing and billing mechanisms assume to a large degree, that addresses are not private. Anyone trying to build a house and then telling the post office that it can't assign them a public address or expecting that anyone will build infrastructure to support getting stuff delivered but not using public streets will likely fail. Imagine the cost of supplanting public streets with a totally private conveyance system (air space, sea space, and space space are public already).

If one wanted to build a totally private alternative to the public road network, then maybe automobiles and vans on private roads wouldn't need license plates (oh wait these are called toll roads, and they need a billing address) so not.

Survelliance Legalese

Physical Search

A physical 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.

Privacy Surveillance Definition

We can have strong authenticity and confidentiality (of anything but addresses) but weaker privacy w.r.t addresses (identifiers). Beyond a certain point, better privacy over addresses becomes impractical because the world needs addresses for routing and billing on public infrastructure and will always need them.

Privacy, as non-correlatable identifiers (metadata) will always be difficult.

Whereas, privacy as control over one's data rights is easier because it's not about routing or billing.

For the purposes of protocol design the definitions of confidentiality and privacy that are most suitable are the surveillance definitions because these are primary to the trade-space of PAC.

Privacy Protection as Protection from Exploitation

We define privacy protection as protection against the ***exploitation of correlatable identifiers*** (metadata).

The goal becomes limiting the exploitation via exploitably correlatable identifiers (metadata).

If we can protect against ***exploitation*** derived from the correlation of identifiers, we may achieve effective privacy.

Protection against exploitation via correlation provides a more gentle trade space than mere protection against correlation.

The former is practical and amenable to cost-benefit analysis, but the latter is largely impractical.

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

Three Party Exploitation Model

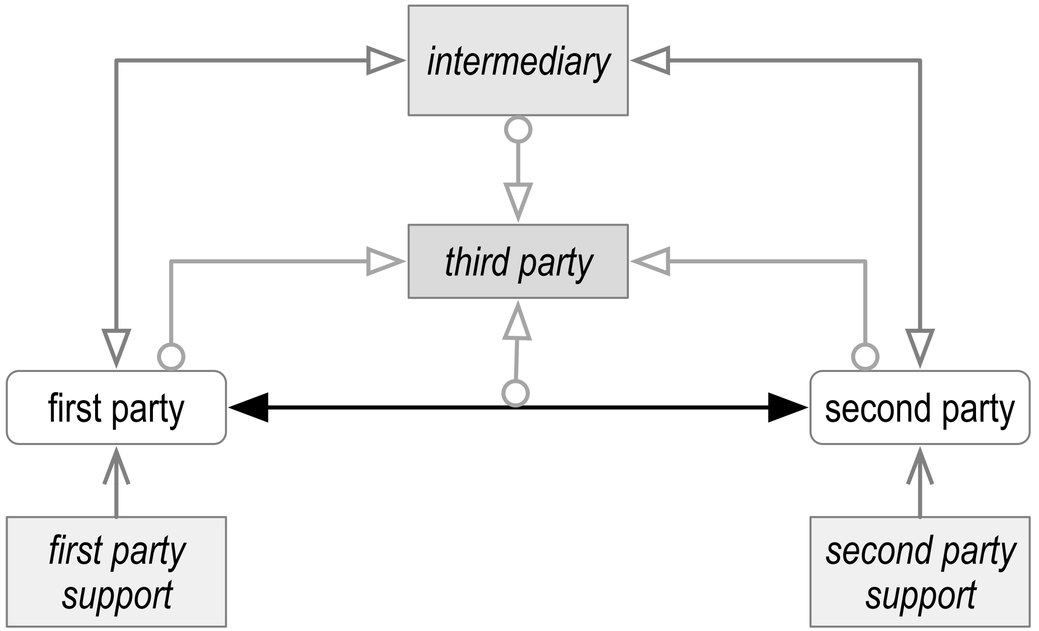
The 1st and 2nd parties represent the parties to a conversation or exchange.

The 1st party discloses data to a 2nd party.

The 1st and 2nd parties have knowledge of each other via their identifiers. The 1st party as discloser has knowledge of the disclosed data (content).

As a result of the disclosure the 2nd party as disclosee also has knowledge of the disclosed data (content).

The identifiers are non-content meta-data, and the disclosed data is content data. A 3rd party is any party that is not either a 1st or 2nd party. An intermediary may be considered a 1st, 2nd, or 3rd party depending on how and by whom it is trusted or not.



Privacy with respect to the 3rd party is protected if the 3rd party has no knowledge of the identifiers used by the 1st and 2nd parties for the conversation (disclosure). Confidentiality with respect to the 3rd party is protected if the 3rd party has no knowledge of the disclosed data (the content data disclosed). A 3rd party may break privacy by directly observing messages that contain the identifiers of the 1st and 2nd parties. A 3rd party may break confidentiality directly by observing the content of messages between the 1st and 2nd parties. In addition, the 3rd party may break both privacy and confidentiality by collusion with the 1st or 2nd party or via collusion with an intermediary. This is diagrammed below.

Privacy in Three Party Exploitation Model

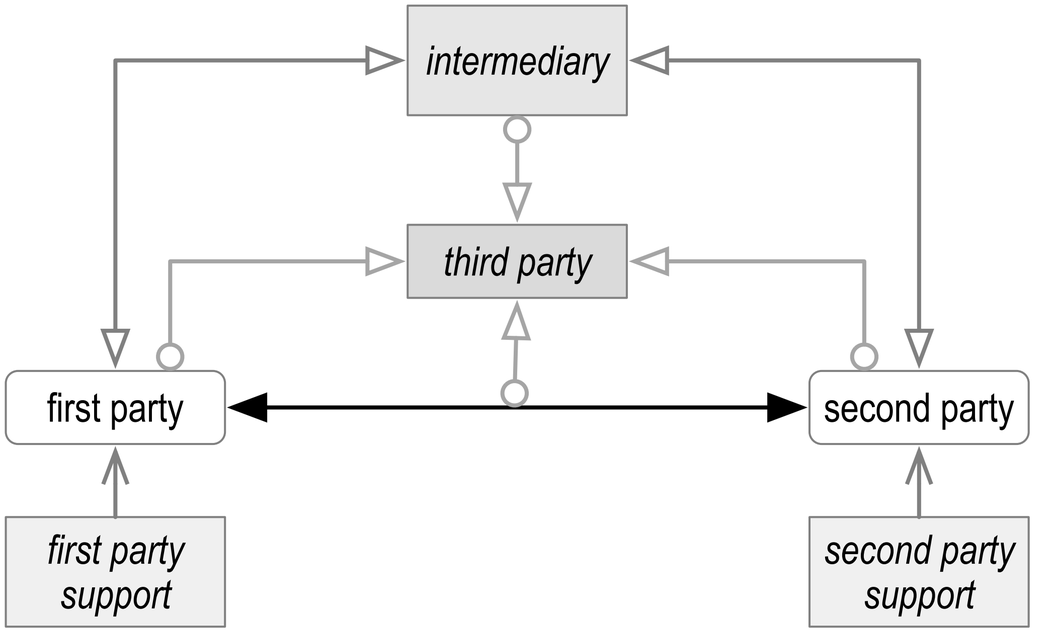
Privacy with respect to the 3rd party is protected if the 3rd party has no knowledge of the identifiers used by the 1st and 2nd parties for the conversation (disclosure).

Confidentiality with respect to the 3rd party is protected if the 3rd party has no knowledge of the disclosed data (the content data disclosed).

A 3rd party may directly break privacy by directly observing messages that contain the identifiers of the 1st and 2nd parties.

A 3rd party may directly break confidentiality by directly observing the content of messages between the 1st and 2nd parties.

The 3rd party may indirectly break both privacy and confidentiality by collusion with the 1st or 2nd party or via collusion with an intermediary.



AIDs as Unbounded Term Identifiers

Long-term public keys and short-term public keys.

Many public key cryptographic algorithms employ the concept of ***long-term*** public keys and ***short-term*** public keys. Often long-term keys have less exposure than short-term keys so they can maintain their cryptographic strength longer.

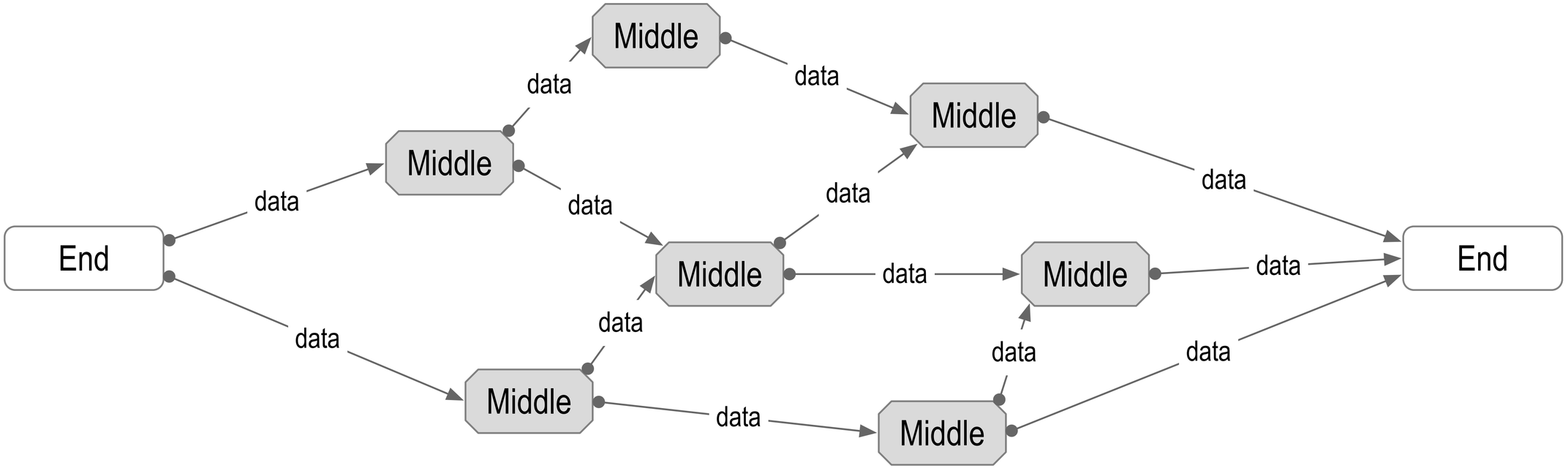
Typically this is to better manage the friction of key exchange where higher friction authentication of long-term keys is used to enable or bootstrap cryptographic operations using short-term or ephemeral keys.

One simplification that results from this approach is that many protocols can be simplified to remove repeated setups using ephemeral (short-term) keys whose purpose is to reduce the exposure of the long-term keys.

Dual End Verifiability and End-Only Viewability

End-to-End Verifiability *of* Authenticity

Only-at-End Viewability via Confidentiality



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

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

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

Zero-Trust-Computing

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

Strong End-Only-Viewable Confidentiality

Use the 3-party model where:

1st party is the sender,

2nd party is the intended receiver

3rd party is any unintended receiver.

A system with strong end-only viewable confidentiality may have the following properties:

- 3rd party non-viewability
- 1st party non-viewability (this protects 1st party from liability wrt to 2nd party and protects 2nd party from exploit by 1st party) (see appendix for a more detailed description of this property)
- 2nd party partition-ability by 1st party
- Detectable leakage (2nd party to 3rd party)
- Detectable collusion of 2nd party against 1st party (2nd party with 2nd party, or 2nd party with 3rd party)
- Detectable collusion of 1st party against 2nd party (1st party with another 1st party relative to same 2nd party or 1st party with 3rd party)

Best Authenticity (Secure Attribution) SUF-CMA

The standard for strong public key signing algorithms is SUF-CMA (Strong UnForgeability under Chosen Message Attack).

When a digital signature has SUF-CMA with at least 128 bits of cryptographic strength then it is computationally infeasible to forge a signature or to discover the private key by choosing messages.

The Libsodium implementation of the NaCL Ed25519 signature scheme has SUF-CMA (See Provable Security of Ed25519 https://eprint.iacr.org/2020/823.pdf).

In comparison, the popular ECDSA signature scheme only has the weaker EUF-CMA (Existential UnForgeability under Chosen Message Attack). Any scheme with SUF-SMA also has EUF-CMA This relative weakness of ECDSA has already been exploited in bitcoin. Consequently, some additional care must be taken when using signature schemes, like ECDSA, that are not SUF-CMA.

Strong Authenticity Baseline

Security properties of Ed25519 in libsodium:

scheme	EUF-CMA	SUF-CMA	S-UEO	M-S-UEO	MBS
Ed25519-LibS	X	X	X	X	X

S-UEO (Strong Universal Exclusive Ownership) and M-S-UEO (Malicious Strong Universal Exclusive Ownership) properties provide additional resiliency against key substitution attacks. Whereas the MBS (Message Bound Security) property ensures a given signature verifies a unique message even for malicious keys.

In addition, Ed25519 avoids private key memory leakage attacks, and its key clamping mechanism makes the re-use of a Ed25519 key pair for its bi-rationally equivalent X25519 encryption key pair resistant to key leakage attacks via the X25519 key pair.

The baseline for strong authenticity is to use the libsodium Ed25519 signature schem

Strong Confidentiality IND-CCA2

The standard for strong public key encryption (PKE) algorithm is IND-CCA2 (Indistinguishability under Adaptive Chosen Ciphertext Attack).

A scheme that has IND-CCA2 also has the weaker IND-CCA (Indistinguishability under Chosen Ciphertext Attack) which also has the weaker still IND-CPA (Indistinguishability under Chosen Plaintext Attack).

What IND-CCA2 means is that an adversary can't submit ciphertexts in such a way as to discover the private key used to decipher those texts even when the adversary has access to an oracle that tells the adversary whether or not the ciphertext was a valid ciphertext. In addition, an adversary can't construct a valid ciphertext unless the adversary is aware of the associated plaintext.

https://blog.cryptographyengineering.com/2018/04/21/wonk-post-chosen-ciphertext-security-in-public-key-encryption-part-1/

https://blog.cryptographyengineering.com/2018/07/20/wonk-post-chosen-ciphertext-security-in-public-key-encryption-part-2/)

IND-CCA2 via ECIES, HPKE

Getting IND-CCA2 requires what is called an Integrated Encryption System (IES) that provides a public key encryption scheme. An IES that uses an elliptic curve is called an ECIES.

A notable example of an ECIES that provides a public key encryption scheme with IND-CCA2 is the NaCL/Libsodium *sealed box*.

What is special about an ECIES like NaCL's *sealed box* is that it provides the performance of symmetric encryption but with the end-only viewability of asymmetric public key encryption.

What we mean by end-only viewability is that only the recipient to whom the box is sealed can decrypt the ciphertext. The sender cannot.

The way *sealed box* does this is that it creates a non-interactive Diffie-Hellman (DH) en-de-cryption key using an ephemeral X25519 key pair with the long-term X25519 public key of the recipient.

The sender never sees the ephemeral X25519 private key so it can't reconstruct the DH en-de-decryption key. The *sealed box* includes the ephemeral X25519 public key so that the recipient can reconstruct the DH en-de-cryption key using its own long-term X25519 private key and the provided ephemeral X25519 public key.

The ciphertext is encrypted using symmetric encryption so it's high performant and only the holder of the long-term X25519 private key for the long-term X25519 public used in the *sealed box* to encrypt, can decrypt the ciphertext.

IETF RFC-9180 has generalized ECIES with HPKE (https://www.rfc-editor.org/rfc/rfc9180.html).

RFC-9180 extends basic HPKE with 4 modes of operation in a standardized approach.

The Libsodium sealed box has equivalent properties to RFC-9180 base mode

IND-CCA2 via ECIES, HPKE

Baseline for IND-CCA2 is the Libsodium Crypto Box Seal or Sealed Box.

This uses the X25519 key pairs and the venerable high-performant XSalsa20 stream cipher with the Poly1305 MAC that together provide the AEAD that underlies the sealed box HPKE.

https://libsodium.gitbook.io/doc/advanced/stream_ciphers/xsalsa20

https://en.wikipedia.org/wiki/ChaCha20-Poly1305

HPKE is not Authenticity

IND-CCA2, by itself, is insufficient to provide strong authenticity of the ciphertext.

As section 9.1.1 of RFC-9180 points out, all variants are vulnerable to key-compromise impersonation attacks that may arise from a compromise of the recipient's private key.

This also allows the attacker to purport that the confidential information was meant for the impersonated key pair thus also breaking authenticity.

RFC-9180 suggests that applications that require resistance against key-compromise impersonation should take extra steps to prevent this attack, such as, using a digital signature generated with the sender's private key over the HPKE container, which means signing the *sealed box* (when using Libsodium).

https://eprint.iacr.org/2006/252.pdf

https://www.rfc-editor.org/rfc/rfc9180.html#name-key-compromise-impersonatio

https://www.cryptologie.net/article/372/key-compromise-impersonation-attacks-kci/

https://www.usenix.org/system/files/conference/woot15/woot15-paper-hlauschek.pdf

Best Combined Authenticity and Confidentiality

For combined strong authenticity and confidentiality we need both hybrid public key encryption to the receiver's key pair (with IND-CCA2) and public key signing from the sender's key pair (with SUF-CMA).

There are several ways to combine a cipher text and a signature.

In order to protect against key-compromise impersonation attacks we must sign the encryption with the recipient's public key in plaintext. This is called encrypt-then-sign.

But merely signing the encryption does not provide complete protection against all attacks.

An attacker can strip the signature and resign the ciphertext with the attacker's key but has never seen the plaintext of the cipher. In some circumstances, this might lead the recipient to falsely believe that the sender had knowledge of the plain text when it did not. For that, we must not only bind the recipient's public key to the signature in plaintext but also bind the sender's public key inside the ciphertext.

ESSR (Encrypt Sender Sign Receiver)

An approach that protects against both KCI and sender impersonation of the ciphertext is called ESSR.

https://eprint.iacr.org/2001/079)

https://neilmadden.blog/2018/11/14/public-key-authenticated-encryption-and-why-you-want-it-part-i/

https://neilmadden.blog/2018/11/26/public-key-authenticated-encryption-and-why-you-want-it-part-ii/

https://neilmadden.blog/2018/12/14/public-key-authenticated-encryption-and-why-you-want-it-part-iii/

The properties that protect against key compromise impersonation attacks are called

TUF-PTXT (Third-party UnForgeability of PlainText),

TUF-CTXT (Third-party UnForgeability of CipherText),

RUF-PTXT (Receiver UnForgeability of PlainText),

RUF-CTXT (Receiver UnForgeability of CipherText).

Simply using the encrypt then sign approach provides TUF-PTXT, TUF-CTXT, and RUF-CTXT but not RUF-PTXT.

scheme	TUF-PTXT	TUF-CTXT	RUF-PTXT	RUF-CTXT
Encrypt-then-Sign	X	X		X
ESSR	X	X	X	X

As a result of binding the sender's public key inside the ciphertext and binding the receiver's public key in the enclosing signed plain text an adversary is prevented from forging messages that compromise either authenticity or confidentiality. Thus ESSR provides both strong authenticity and strong confidentiality.

ESSR (Encrypt Sender Sign Receiver)

An approach that protects against both KCI and sender impersonation of the ciphertext is called ESSR.

https://eprint.iacr.org/2001/079)

https://neilmadden.blog/2018/11/14/public-key-authenticated-encryption-and-why-you-want-it-part-i/

https://neilmadden.blog/2018/11/26/public-key-authenticated-encryption-and-why-you-want-it-part-ii/

https://neilmadden.blog/2018/12/14/public-key-authenticated-encryption-and-why-you-want-it-part-iii/

The properties that protect against key compromise impersonation attacks are called

TUF-PTXT (Third-party UnForgeability of PlainText),

TUF-CTXT (Third-party UnForgeability of CipherText),

RUF-PTXT (Receiver UnForgeability of PlainText),

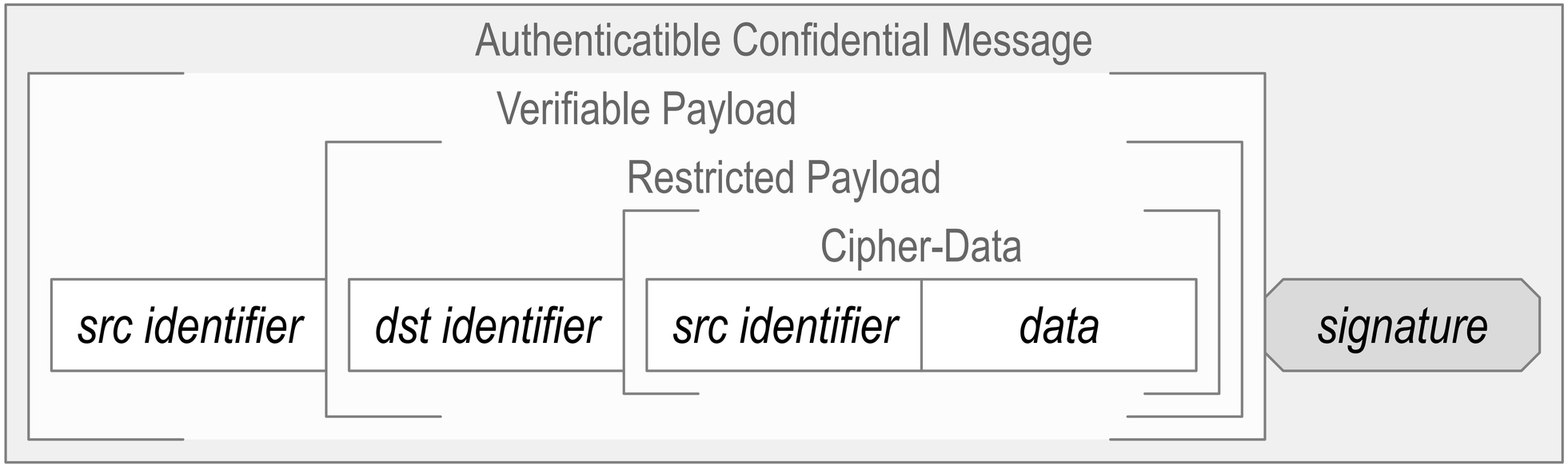
RUF-CTXT (Receiver UnForgeability of CipherText).

Simply using the encrypt then sign approach provides TUF-PTXT, TUF-CTXT, and RUF-CTXT but not RUF-PTXT.

scheme	TUF-PTXT	TUF-CTXT	RUF-PTXT	RUF-CTXT
Encrypt-then-Sign	X	X		X
ESSR	X	X	X	X

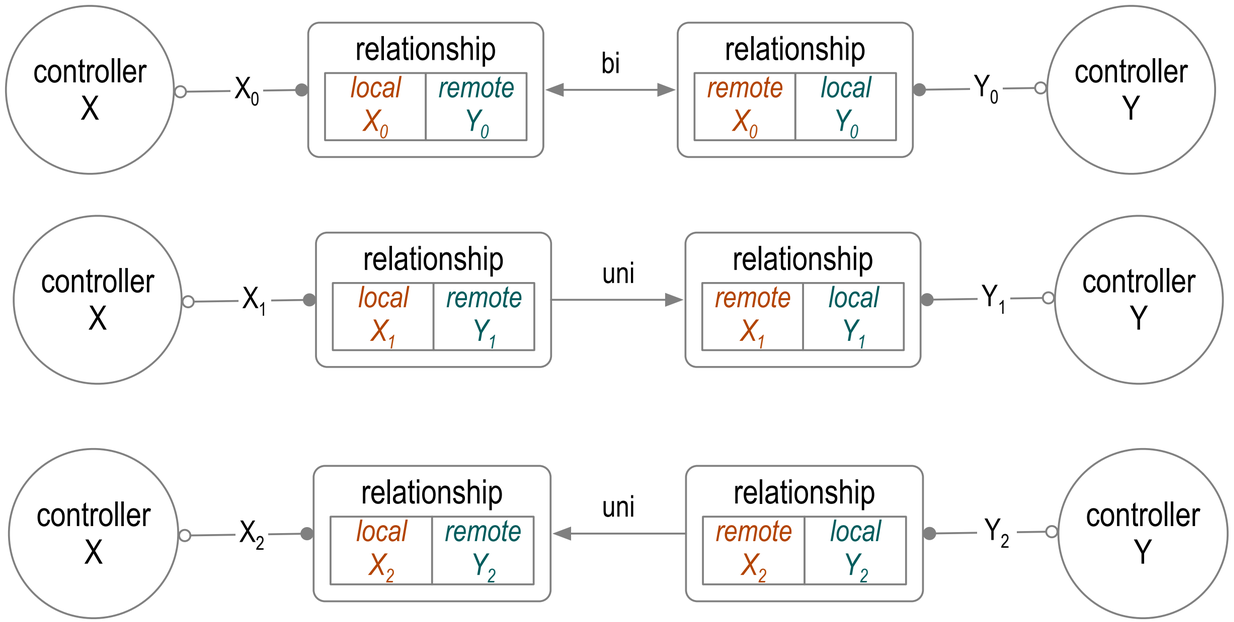
As a result of binding the sender's public key inside the ciphertext and binding the receiver's public key in the enclosing signed plain text an adversary is prevented from forging messages that compromise either authenticity or confidentiality. Thus ESSR provides both strong authenticity and strong confidentiality.

Baseline ESSR Message

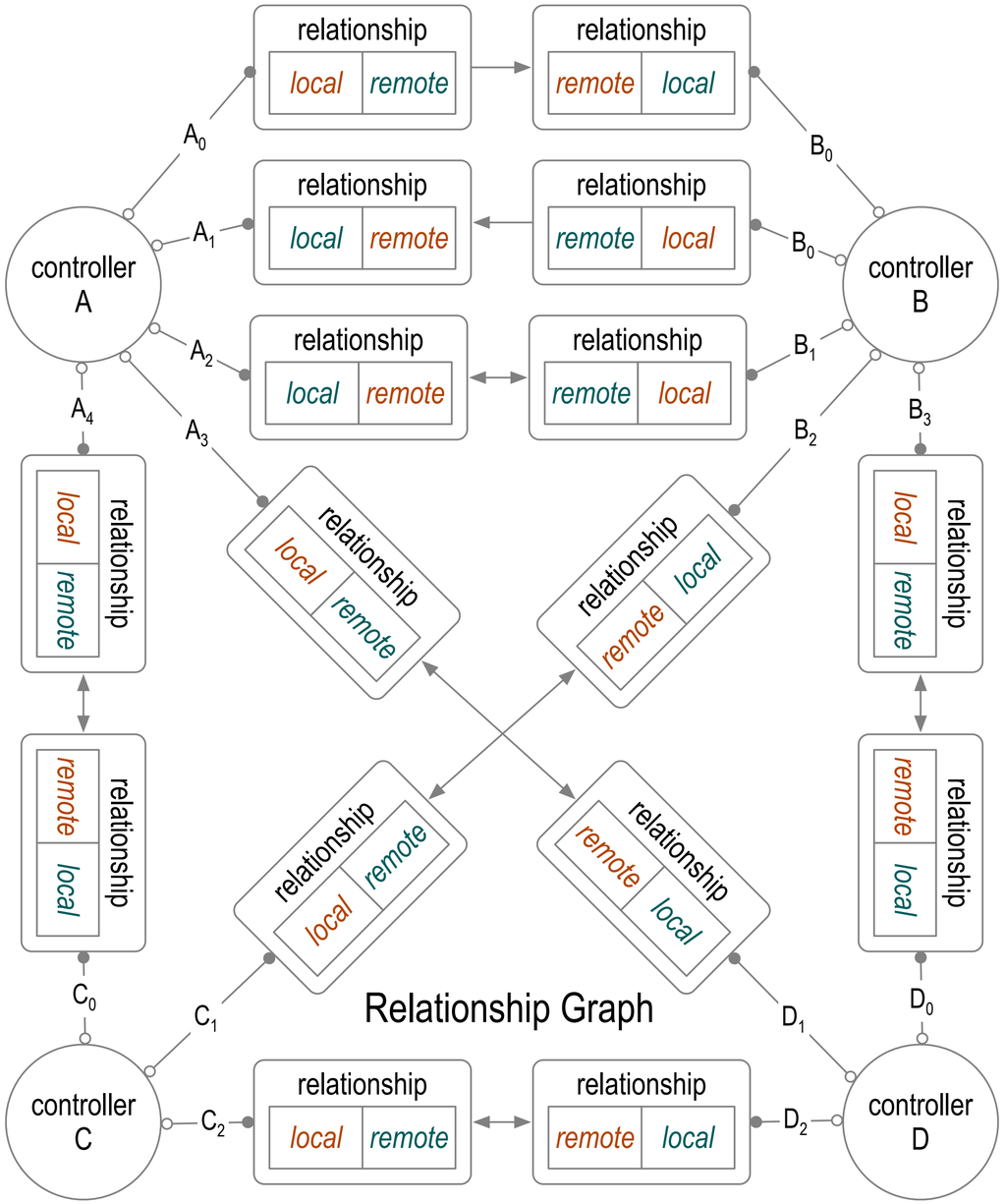


Relationships

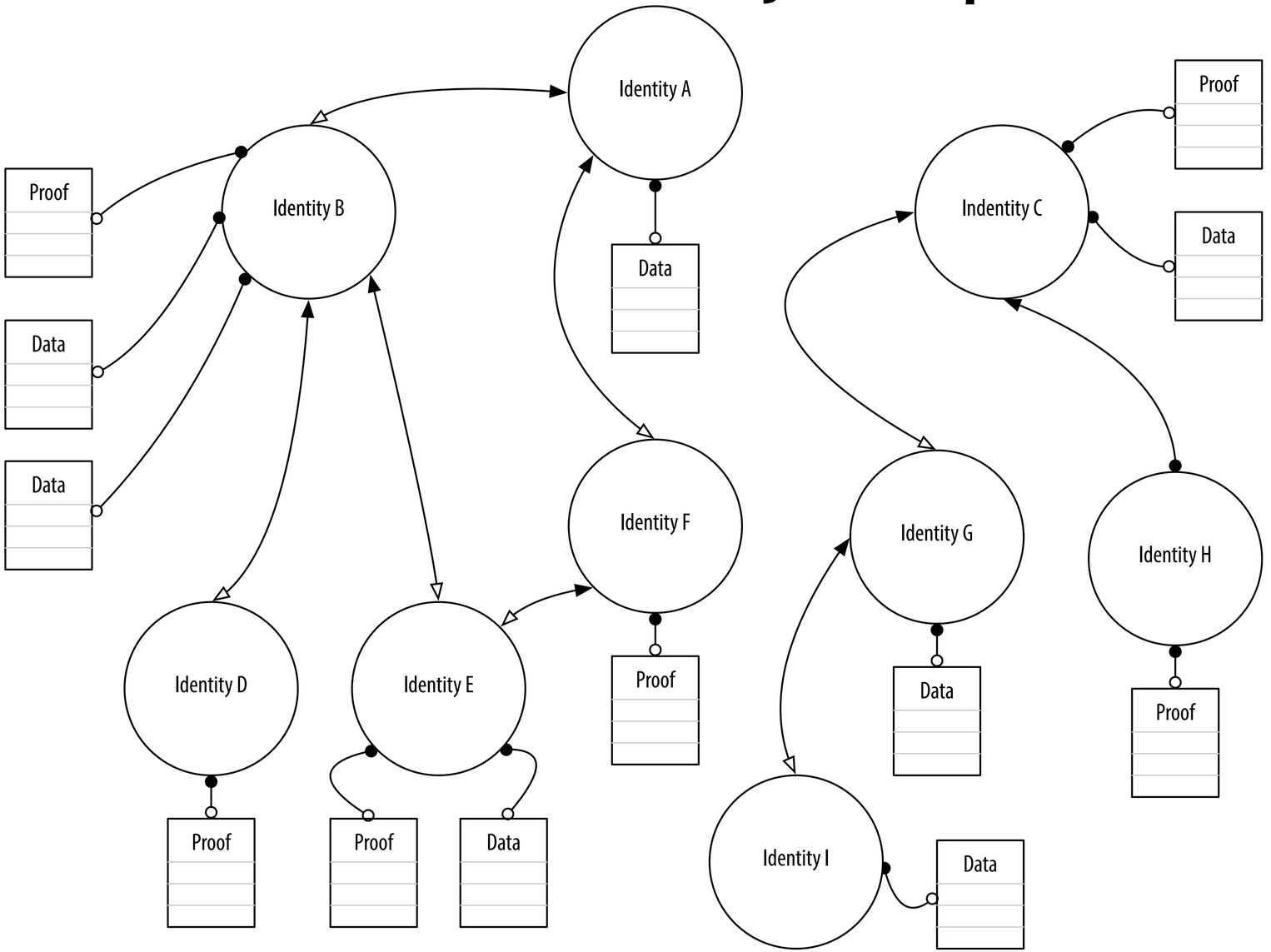
```
(A, <>, B)
(A, ->, B)
(A, <-, B)
```



Relationship Graph (global)



Self-Identity Graph



Relationship graph is the intersection of identity graphs.

Relationship Properties

A ***cryptonym*** is a cryptographically derived pseudonym with at least 128 bits of entropy in the derivation.

An ***AID*** is a cryptonym that is also securely attributable to one or more key pairs.

A ***relationship*** is a pairing of two AIDs, one each from a different *controller*.

*Two different **AIDs** are by themselves uncorrelated in an information-theoretic security sense in that knowledge of one by itself provides no information about the other.*

A ***context*** is a set of events.

*Two **contexts** are disjoint with respect to an AID when that AID appears in one or more events in one set but does not appear in any event in* the *other set.*

A relationship is not a *communication channel*, but the events sent over a communication channel may be a ***context*** for the aids in a ***relationship***.

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

*Any set of relationships using **ORIs (One Relationship Identifiers)** may form a partition.*

Any two member *contexts* of a *partition* may be correlatable due to other information associated with those contexts, but the partitioned relationships by themselves provide no correlatable information, i.e., *partitioned relationships* are by themselves not mutually correlatable

An *AID* common to any subset of events within a context provides a *perfectly correlatable* feature across those common events in that context.

Within a context, the *secure attributability* of an AID, together with its *perfect correlatability,* enables secure *reputational trust* (good or bad) in that AID within that context.

Partitions balance the concerns of:

strong authenticity and strong confidentiality within a context with,

sufficiently strong privacy between contexts.

OOB Setup

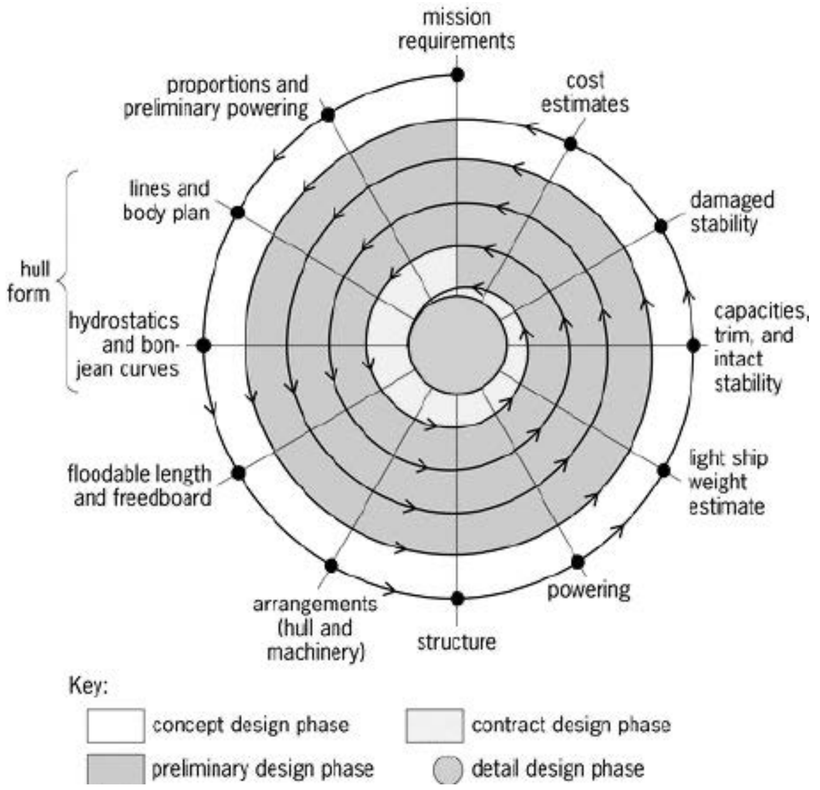
Assume at least one OOBA (out-of-band-authentication) factor to protect from MITM (man-in-the-middle) attack.

OOBA setup also provides protection from cheap pseudonymity.

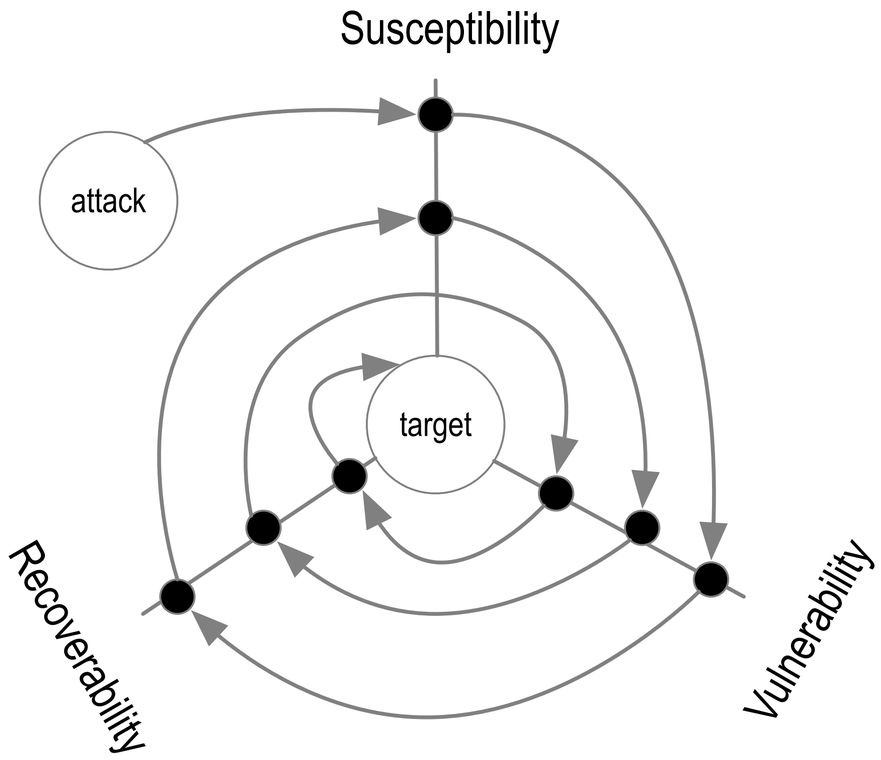
An OOB setup can simultaneously be strongly authentic, confidential, and private.

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



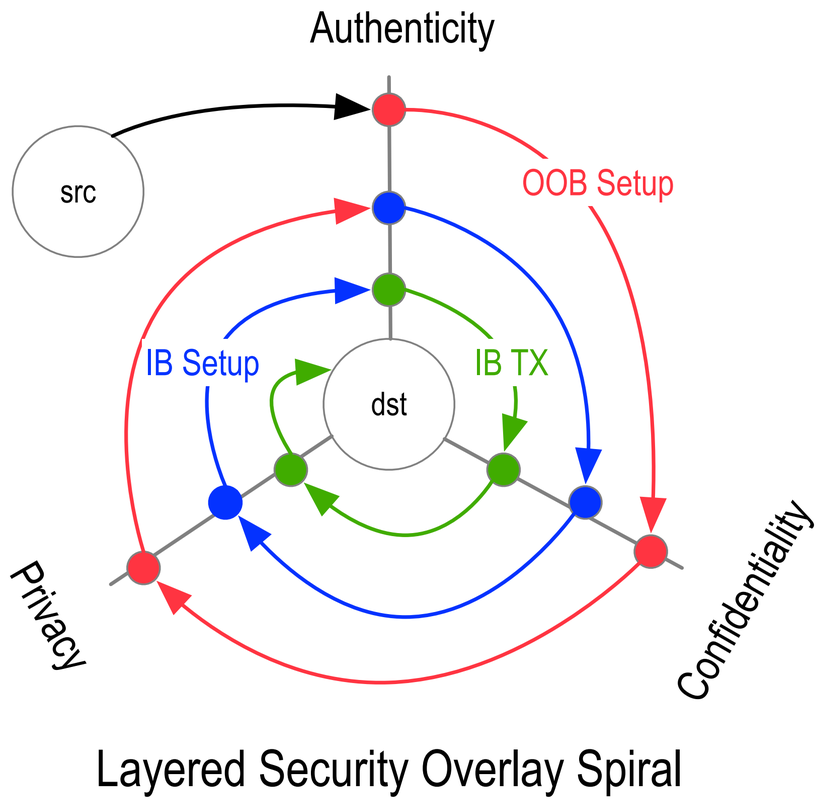
Naval Architecture Design Spiral



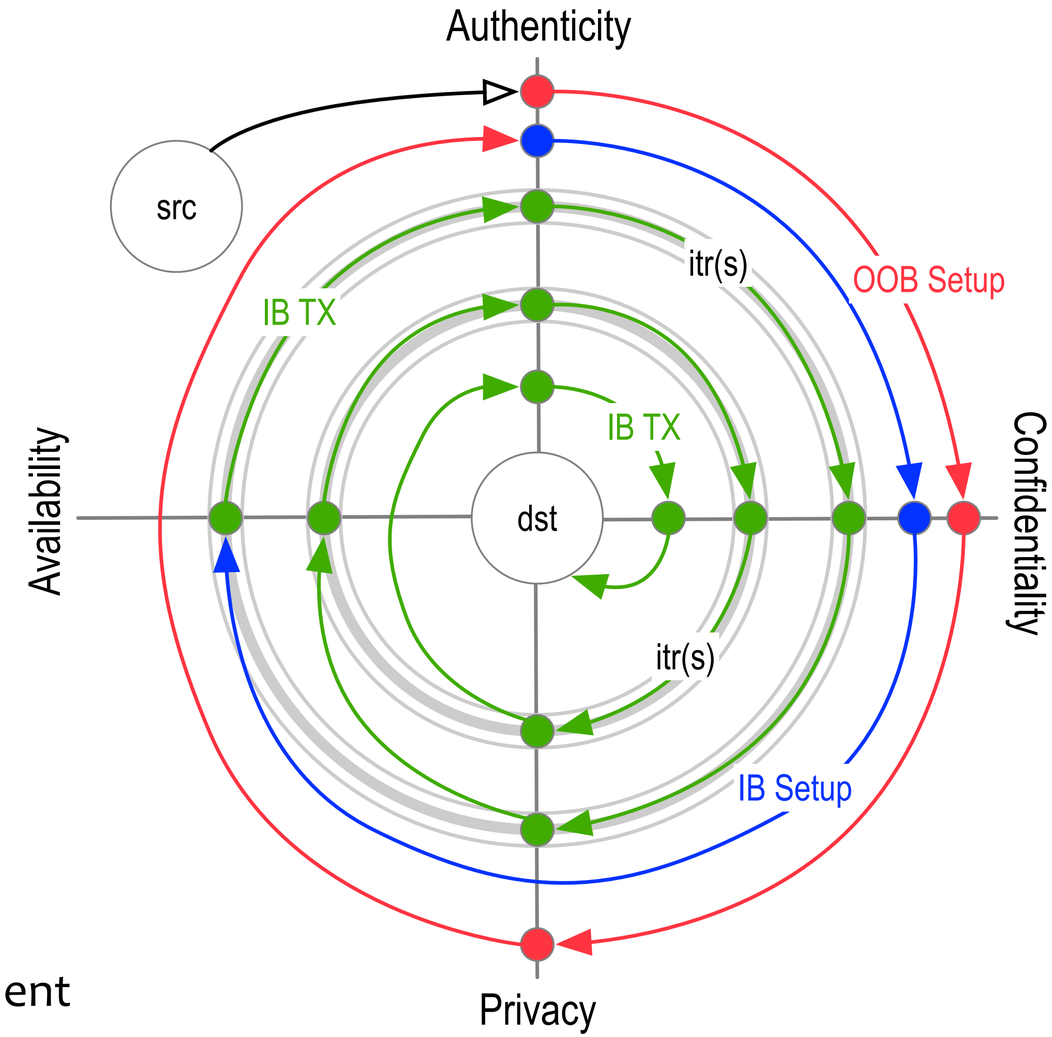
Layered Survivabilty Overlay Spiral

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

Protocol Syntax

AIDs:

X_0, X_1, X_2

Relationships:

$(X_0, <>, Y_0)$

$(X_1, ->, Z_0)$

$(X_2, <-, Y_1)$

Signatures:

$<data>\ X_0$

$<data>\ (X_0, X_1)$

CipherText:

$\{data\}Y_2$

Messages:

$[src\ X_0, dst\ Y_0, data]$

$<[src\ X_0, dst\ Y_0, \{src\ X_0, data\}Y_0]>X_0$

Interaction Context ID:

$i^0X_0Y_0$

Header:

(version, protocol type, packet type, interaction ID, sequence number, ….)

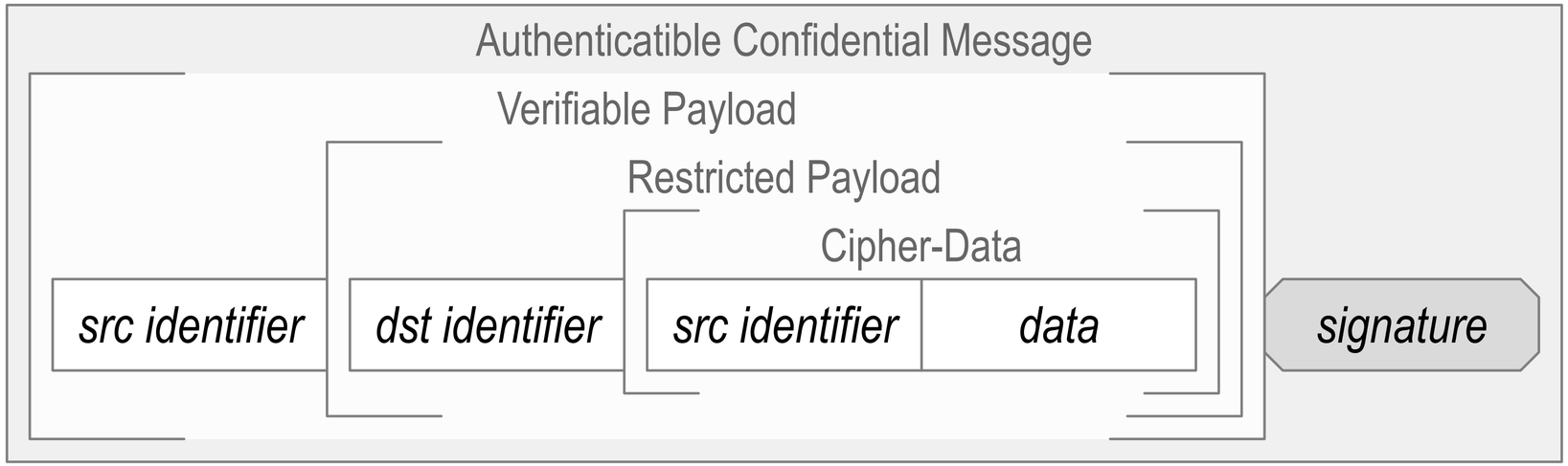
$<[header, src\ X_0, dst\ Y_0, \{src\ X_0, data\}Y_0]>X_0$

Transport (IP) Header: (IP addresses are public by nature with inherently correlatable global context)

$|src\ ip\ X_0, dst\ ip\ Y_0|$

Example:

$|src\ ip\ X_0, dst\ ip\ Y_0|<[header, src\ X_0, dst\ ip\ Y_0, dst\ Y_0, \{src\ X_0, data\}Y_0]>X_0$



Direct Protocol

OOB:

entities: A, B

relationship: (A_0, <>, B_0)

service endpoints: ip A_0, ip B_0

IB:

|src ip A_0, dst ip B_0|<[src A_0, dst B_0, {src A_0, data}B_0]>A_0

|src ip B_0, dst ip A_0|<[src B_0, dst A_0, {src B_0, data}A_0]>B_0

Legal Recourse Variant

|src ip A_0, dst ip B_0|<[src A_0, dst B_0, {<src A_0, data>A_0}B_0]>A_0

|src ip B_0, dst ip A_0|<[src B_0, dst A_0, {<src B_0, data>B_0}A_0]>B_0

Correlatable Metadata Identifiers:

ip A_0, ip B_0, A_0, B_0

Confidential Information:

data

Strongly Authentic and Confidential but not Private

Direct Protocol as Wrapper

OOB:

entities: A, B

relationship: (A_0, <>, B_0)

service endpoints: ip A_0, ip B_0

relationship: (A_1, <>, B_1)

IB:

|src ip A_0, dst ip B_0|<[src A_0, dst B_0, {src A_0, <[src A_1, dst B_1, data]>A_1}B_0]>A_0

|src ip B_0, dst ip A_0|<[src B_0, dst A_0, {src B_0, <[src B_1, dst B_1, data]>B_1}A_0]>B_0

Correlatable Metadata Identifiers:

ip A_0, ip B_0, A_0, B_0

Confidential Information:

data, A_1, B_1

Private Metadata Identifiers

A_1, B_1

Strongly Authentic and Confidential and Somewhat Private

Relationship Formation Protocol

OOB:

entities: A, B

relationship: (A_0, <>, B_0)

service endpoints: ip A_0, ip B_0

IB:

relationship: (A_1, <>, B_1), $i^0A_1B_0$, $i^1A_1B_0$

|src ip A_0, dst ip B_0|<[src A_0, dst B_0, {src A_0, <[$i^0A_1B_0$, A_1, bi]>A_1}B_0]>A_0

|src ip B_0, dst ip A_0|<[src B_0, dst A_0, {src B_0, <[$i^1A_1B_0$, A_1, bi, B_1]>B_1}A_0]>B_0

Correlatable Metadata Identifiers:

ip A_0, ip B_0, A_0, B_0

Confidential Information:

data, A_1, B_1, iA_1B_0

Private Metadata Identifiers

A_1, B_1, $i^0A_1B_0$, $i^1A_1B_0$

Strongly Authentic and Confidential and Somewhat Private

One Shared Intermediary

OOB:

entities: A, B, C

relationships: (A_1, <>, B_1), (A_2, <>, C_0), (B_2, <>, C_1)

service endpoints: ip A_2, ip B_2, ip C_0, ip C_1

C database:

<[C_0,<[A_2, ip A_2]>A_2]>C_0

<[C_1,<[B_2, ip B_2]>B_2]>C_1

C has communication relationships with 1000+ other entities

IB:

Hop1 A to C

|src ip A_2, dst ip C_0|<[src A_2, dst C_0, {src A_2, dst C_1, <[src A_1, dst B_1, {src A_1, data}B_1]>A_1 }C_0]>A_2

Hop2 C to B

|src ip C_1, dst ip B_2|<[src C_1, dst B_2, {src C_1, <[src A_1, dst B_1, {src A_1, data}B_1]>A_1 }B_2]>C_1

Correlatable Metadata Identifiers:

(ip A_2, ip C_0, A_2, C_0)

(ip B_2, ip C_1, B_2, C_1)

Confidential Information:

data, A_1, B_1,

Private Metadata Identifiers

A_1, B_1

Strongly Authentic and Confidential and Mostly Private

One Shared Intermediary

The AIDS in (A_1, <>, B_1) are never stored on disk by C. They are only ever in memory. This means that an attacker who compromises the disk storage of C cannot leak (A_1, <>, B_1).

An attack on C's protected memory processes is much more difficult than an attack on C's disk storage. This is more than sufficient to protect against 3rd party correlation for advertising aggregation.

A does not need to know the B side of B's relationship with C, and B does not need to know the A side of A's relationship with C. So neither can leak that information. An attacker has to get that by attacking C, not either A or B.

Thus A and B can isolated their independent hop-wise communications contexts with C from their joint end-wise routing context. The three contexts form a partition with respect to 3rd party correlatability of the AIDs.

The primary limitations of this approach are as follows:

- Both A and B must trust C.
- The ISP of C has access to both the source and destination IPs of all C's packets. With some extra work, the ISP could correlate observable statistics (but not AIDs) about the packets, such as time-of-arrival and time-of-departure, packet size, packet type, protocol types, etc.

Non Shared Intermediaries

OOB:

entities: A, B, C, D

relationships: (A_1, <>, B_1), (A_2, <>, C_0), (B_2, <>, D_0), (C_2, <>, D_2)

service endpoints: ip A_2, ip B_2, ip C_0, ip D_0, ip C2, ip D_2

C database: <[C_0,<[A_2, ip A_2]>A_2]>C_0 <[A_2, D_2, ip D_2]>A_2

D database: <[D_0,<[B_2, ip B_2]>B_2]>D_0 <[B_2, C_2, ip C_2]>B_2

C and D each have communication relationships with 1000+ other entities

IB:

Hop1 A to C

|src ip A_2, dst ip C_0|<[src A_2, dst C_0, {src A_2, dst D_2, dst D_0, <[src A_1, dst B_1, {src A_1, data}B_1]>A_1}C_0]>A_2

Hop2 C to D

|src ip C_2, dst ip D_2|<[src C_2, dst D_2, {src C_2, dst D_0, <[src A_1, dst B_1, {src A_1, data}B_1]>A_1}D_2]>C_2

Hop3 D to B

|src ip D_0, dst ip B_2|<[src D_0, dst B_2, {src D_0, <[src A_1, dst B_1, {src A_1, data}B_1]>A_1}B_2]>D_0

Correlatable Metadata Identifiers:

(ip A_2, ip C_0, A_2, C_0, D_2, D_0)

(ip B_2, ip D_0, B_2, Do, C_2, C_0)

Confidential Information:

data, A_1, B_1, A_2, B_2

Private Metadata Identifiers

A_1, B_1, A_2, B_2

Strongly Authentic and Confidential and Very Private

Non Shared Intermediaries

All any 3rd party ISP sees is that A_2 has a relationship with C and B_2 has a relationship with D but can't correlate those relationships to (A1, <>, B1) given sufficient herd privacy of C's and D's relationships with others.

A and B both trust that C and D do not leak (A1, <>, B1).

The AIDS in (A1, <>, B1) are never stored on disk by either C or D. They are only ever in memory. This means that an attacker who compromises the disk storage of C or D cannot leak (A1, <>, B1). An attack on C's or D's protected memory processes is much more difficult than an attack on C's or D's disk storage.

A does not need to know the B side of B's relationship with D, and B does not need to know the A side of A's relationship with C. So neither can leak that information.

An attack on C's disk does not expose that A has a relationship with any of D's relationships because the farside destination is only exposed in memory. Likewise, an attack on D's disk does not expose that B has any relationships with A because the farside destination is only exposed in memory.

This approach is fully zero-trust because all table lookups by C and D are signed at rest, so an attacker can't misdirect traffic between the two or between their associated edge relationships.

A, B, C, and D have isolated their independent hop-wise communication contexts with each other from the A and B joint end-wise context (A1, <>, B1). The end-wise relationship provides both an end-to-end routing context at the AID level and an end-to-end interaction context. These four contexts (independent hop-wise and joint end-wise) are mutually partitioned wrt 3rd party correlation.

We can conclude that we can have secure authenticity, confidentiality, and privacy (identifier correlation), as well as DDOS protection.

The only OOB setups are one-time setups between A and B, A and C, and B and D.

Non Shared Intermediaries as Wrapper

OOB:

entities: A, B, C, D

relationships: (A_3, <>, B_3) (A_1, <>, B_1), (A_2, <>, C_0), (B_2, <>, D_0), (C_2, <>, D_2)

service endpoints: ip A_2, ip B_2, ip C_0, ip D_0, ip C2, ip D_2

C database: <[C_0,<[A_2, ip A_2]>A_2]>C_0 <[A_2, D_2, ip D_2]>A_2

D database: <[D_0,<[B_2, ip B_2]>B_2]>D_0 <[B_2, C_2, ip C_2]>B_2

C and D each have communication relationships with 1000+ other entities

IB:

Hop1 A to C

*|src ip A_2, dst ip C_0|<[src A_2, dst C_0, {src A_2, dst D_2, dst D_0, <[src A_1, dst B_1, {src A_1, **<[src A_3, dst B_3, i^0AB, data]>A_3}B_1]>A_1}C_0]>A_2*

Hop1 A to C Variant

*|src ip A_2, dst ip C_0|<[src A_2, dst C_0, {src A_2, dst D_2, dst D_0, <[src A_1, dst B_1, {src A_1, **<[src A_3, dst B_3, {src A_3, i^0AB, data}B_3]>A_3}B_1]>A_1}C_0]>A_2*

Correlatable Metadata Identifiers:

(ip A_2, ip C_0, A_2, C_0, D_2, D_0)

(ip B_2, ip D_0, B_2, Do, C_2, C_0)

Confidential Information:

*data, A_1, B_1, A_2, B_2, **A_3, B_3, i^0AB***

Private Metadata Identifiers

*A_1, B_1, A_2, B_2, **A_3, B_3, i^0AB***

Strongly Authentic and Confidential and Private

Non Shared Intermediaries as Wrapper

This approach inherits all the properties of Non-Shared intermediaries but with the following additions:

C and D can not leak (A3, <>, B3)

The aids in (A3, <>, B3) are never viewed by C or D.

A, B, C, and D have isolated their independent hop-wise communication contexts with each other from the A and B joint end-wise context (A1, <>, B1) and from the end-wise interaction context (A3, <>, B3) .

The end-wise relationship (A1, <>, B1) only provides a generic end-to-end routing context at the AID level

The end-wise relationship (A3, <>, B3) provides an end-to-end interaction context. These five contexts (independent hop-wise and joint end-wise) are mutually partitioned wrt 3rd party correlation.

The interaction context (A3, <>, B3) is also partitioned with respect to C and D.

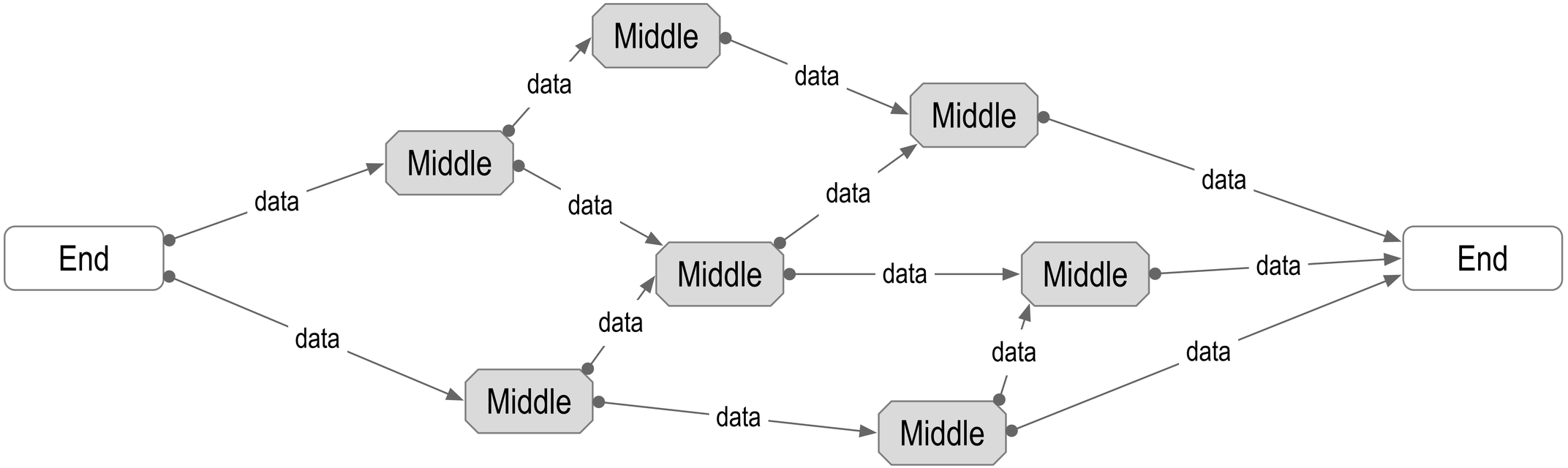
We can conclude that we can have secure authenticity, confidentiality, and privacy (identifier correlation), as well as DDOS protection.

The only OOB setups are one-time setups between A and B, A and C, and B and D.

Dual End Verifiability and End-Only Viewability

End-to-End Verifiability *of* Authenticity

Only-at-End Viewability via Confidentiality



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

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

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

Zero-Trust-Computing

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

Strong End-Only-Viewable Confidentiality

Use the 3-party model where:

1st party is the sender,

2nd party is the intended receiver

3rd party is any unintended receiver.

A system with strong end-only viewable confidentiality may have the following properties:

- 3rd party non-viewability
- 1st party non-viewability (this protects 1st party from liability wrt to 2nd party and protects 2nd party from exploit by 1st party) (see appendix for a more detailed description of this property)
- 2nd party partition-ability by 1st party
- Detectable leakage (2nd party to 3rd party)
- Detectable collusion of 2nd party against 1st party (2nd party with 2nd party, or 2nd party with 3rd party)
- Detectable collusion of 1st party against 2nd party (1st party with another 1st party relative to same 2nd party or 1st party with 3rd party)

Identity Theft Threats

Multi-step recursive privilege elevation attack (authentication and authorization).

The attacker compromises weak access credentials on low-value targets that allow it to elevate its privileges in order to compromise a higher-value target until it finally has the access credentials of a treasure trove.

An example of this would be to compromise the VPN login of some customer or employee of some company that allows an attacker to compromise some other employee's VPN login to a connected company and so on until it captures access credentials to a connected company's treasure trove.

It starts with what is called an edge attack on the weakest company.

The elevation mechanism often leverages what is called a BOLA or BUA.

These are not privacy weaknesses but authentication-based authorization weaknesses.

“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”

To summarize, a given trust domain is only as strong as the weakest connected trust domain (in which connected is recursively applied).

https://www.apple.com/newsroom/pdfs/The-Rising-Threat-to-Consumer-Data-in-the-Cloud.pdf)

https://heimdalsecurity.com/blog/what-is-broken-object-level-authorization-bola/

https://github.com/OWASP/API-Security/blob/master/2019/en/src/0xa2-broken-user-authentication.md) https://www.traceable.ai/blog-post/a-deep-dive-on-the-most-critical-api-vulnerability-bola-broken-object-level-authorization

https://owasp.org/ww-project-api-security/)

Clandestine and Covert Operations

Covert: A covert operation is an operation that is planned and executed in secrecy so that the identity of the agency or organization remains unknown or is plausibly deniable.

Clandestine: A clandestine operation is an operation that is carried out in such a manner that the operation remains in secrecy or is concealed.

Confusion of operative vs. organization in the application of privacy protection mechanisms from spy craft.

Exclusively clandestine measures are unsustainable for operative. Instead use covert measures to minimize the susceptibility of infrequent clandestine measures.

Use Covert to hide in plain sight.

Nest clandestine contexts inside covert contexts

Example:

Using a vpn on Saturday to trade on a Cyprus cryptocurrency exchange.

vs.

Using a vpn for all communications not merely the cryptocurrency exchange.

Communications context = herd privacy (covert)

Routing context = herd privacy (covert)

Interaction context (clandestine)

Exploiters

State Actors (3rd party):

The primary incentives for state actors are political, but some like North Korea are also monetary.

State actors may use any combination of attacks to surveil, regulate, and persecute all first parties, second parties, and intermediaries with virtually unlimited resources and the ability to cause harm.

Because of the virtually unlimited resources and techniques that state actors can apply we will treat them as out of scope for mitigation for the time being. That does not mean we will not consider state-actor resistant mitigations but to attempt state-actor proof mitigations would likely take us down rabbit holes that we can never return from.

Identity Thieves (3rd party):

The primary incentive for identity thieves is monetization through asset theft. Typically either directly through the capture of access credentials to assets or indirectly through the sale of personally identifying information (PII) that may be used to capture access credentials. Another primary monetization strategy is ransomware of either access credentials or PII that may be used to capture access credentials or simply sensitive PII such as health data.

Aggregators (3rd party, 2nd party, 1st party):

The primary incentive for aggregators is to sell data to advertisers. This means profiling groups of potential customers in attractive demographics. This profiling may be entirely statistical. There are several classes of aggregators. These include:

Intermediaries (3rd party):

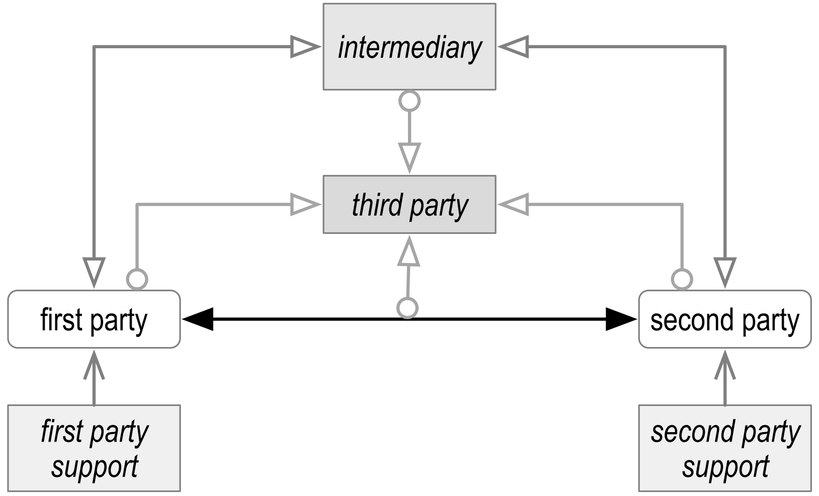
* Search as an aggregator
* Cloud provider as an aggregator
* ISP as an aggregator

Platforms (2nd party)

* Colluding 2nd parties as aggregators

Inadvertent (1st party)

* Inadvertent colluding 1st party as an aggregator via web cookies



Identity Theft Mitigations

* Use strong authentication by 2nd parties of any first party at any step along a multi-step recursive privilege elevation attack. In any multistep recursive privilege elevation attack any step that has strong authentication with individually signed requests for authorization will stop the attack from proceeding.

* Use strong authentication by any 2nd party of any 1st party for any exploitable use of first-party credentials, whether those credentials be credit card, bank account, or access to sensitive data.

* Use strong authentication (zero-trust data in motion and at rest) at any party holding a treasure trove. Strong authentication is also state-actor resistant.

* Get rid of treasure troves of access credentials by using decentralized trust domains (DPKI). This means passwordless security with strong key rotation mechanisms. This is also state actor resistant.

Third-Party Aggregator Threats and Mitigations 1

Search as an intermediary (3rd party).

In this case, an internet search engine is able to track the click-throughs of a successful search result. This enables the search engine to monetize the 1st party searcher by correlating the meta-data of the incoming client browser with the 2nd party endpoint resulting from the search.

* Mechanisms:

- 1st party permissioned surveillance of first party metadata collected by 3rd party search intermediary and sold to other 3rd parties

* Mitigations:

- Use VPN in combination with a partitioned 1st party identifier such as a random email address

- Use a search engine that contractually protects 1st party search metadata.

Third-Party Aggregator Threats and Mitigations 2

Cloud data storage provider as an intermediary (3rd party).

In this case, all 1st party content data stored in the cloud may be anonymized and re-correlated to provide monetizable advertising campaign pools.

* Mechanisms:

- First-party permissioned surveillance of first-party content data by 3rd party cloud provider intermediary and sold to other 3rd parties.

* Mitigations:

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

- Use cloud storage providers that contractually protect 1st party content data.

Third-Party Aggregator Threats and Mitigations 3

I**nternet Service Provider (ISP) as an intermediary** (3rd party).

In this case, ISP has access to all traffic routed through it for 1st party client connections to any 2nd party. ISP can anonymize and then re-correlate into monetizable advertising campaign pools.

* Mechanisms:

- Non-content Metadata (routing and billing) identifiers. In the USA for example any anonymized data (both metadata & content) is aggregation permissible for any ISP.

- Mobile device location tracking.

* Mitigations:

* Use an ISP that contractually protects user metadata

- Use VPN that contractually protects user metadata. A VPN provides herd privacy relative to ISP tracking of the source to 2nd party destinations.

- There is no mitigation for mobile device tracking by mobile service providers except to turn off the mobile device.

- Use distributed decentralized confidential network overlay to make metadata harder to correlate. This could include onion or mix network routing such as TOR, Elixxir, MainFrame, and DIDComm Routing. In general, a decentralized network overlay is difficult to incentivize and as a result, may have relatively low adoptability when compared to a VPN.

- Use information-theoretic secure routing. An example would be random walk routing. Of all the mitigations, this one is state actor resistant. Information-theoretic secure routing, however, has similar adoptability problems as a decentralized confidential network overlay.

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 3rd party surveillor (intermediary)

Statistical regression of anonymized content data by a large second party or by multiple colluding 2nd parties

Mitigation:

Don't use 2nd 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 1st 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 2nd parties or intermediaries

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.

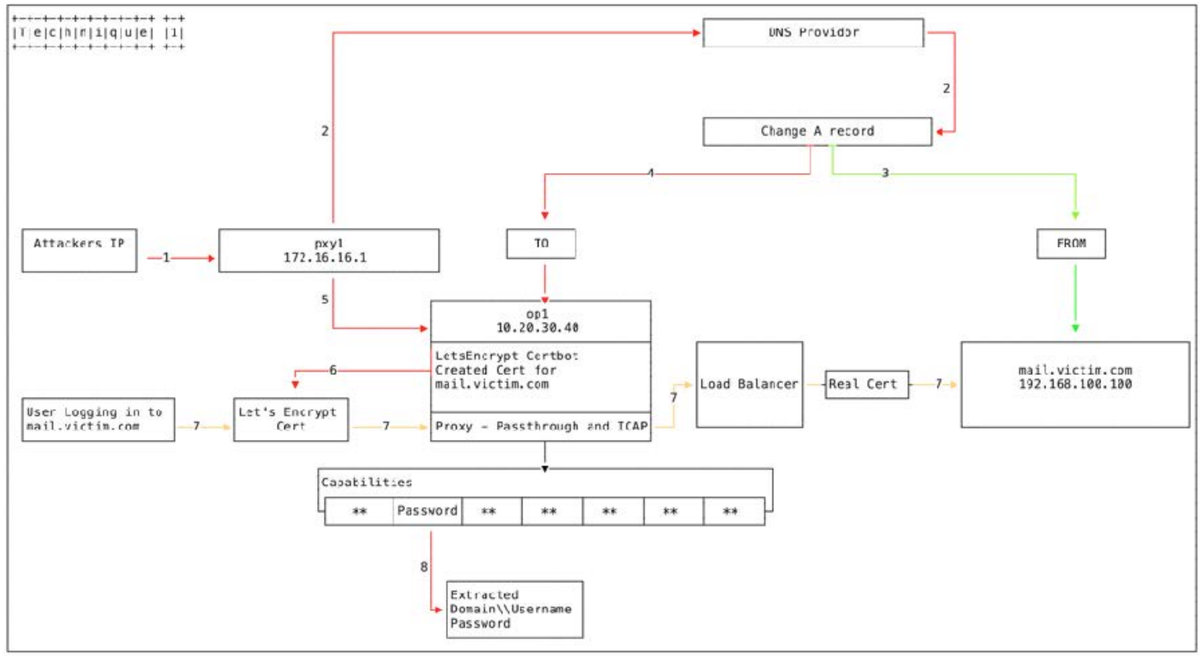
The infrastructure changes needed to fix identity theft will enable easier adoption of the changes needed to better protect against aggregators.

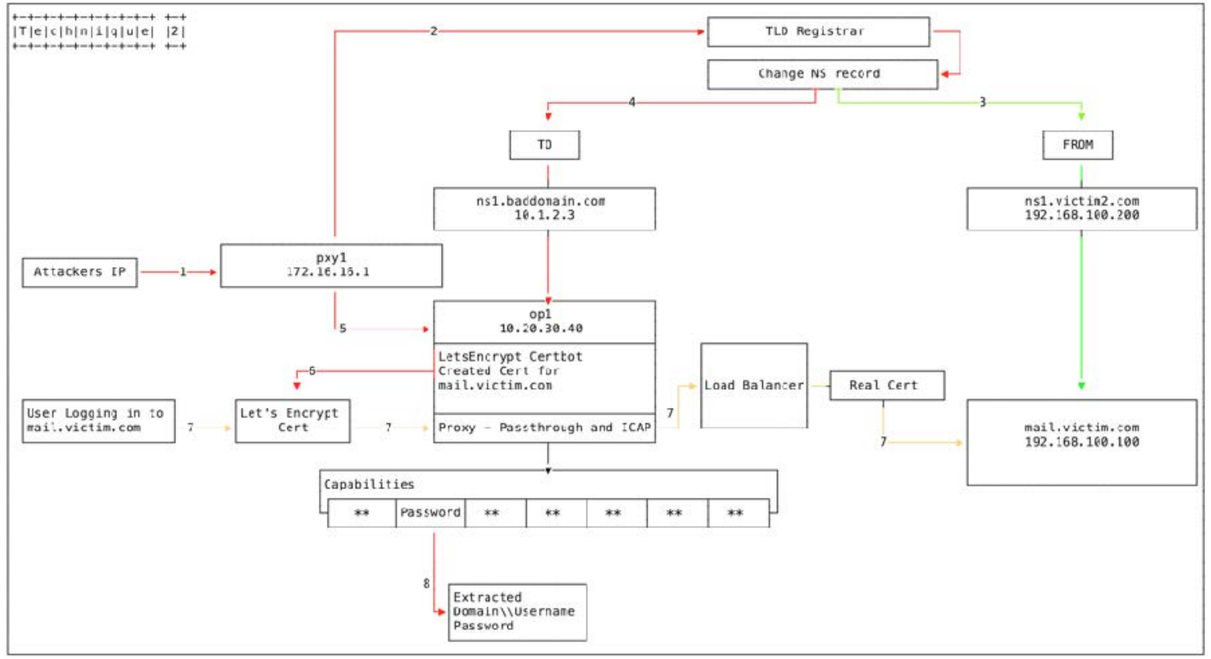
Questions

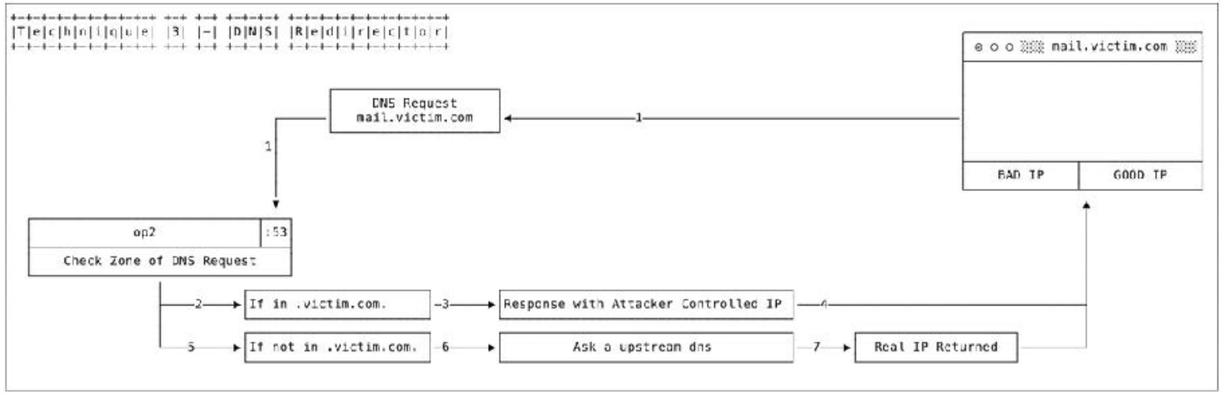
DNS Hijacking

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/







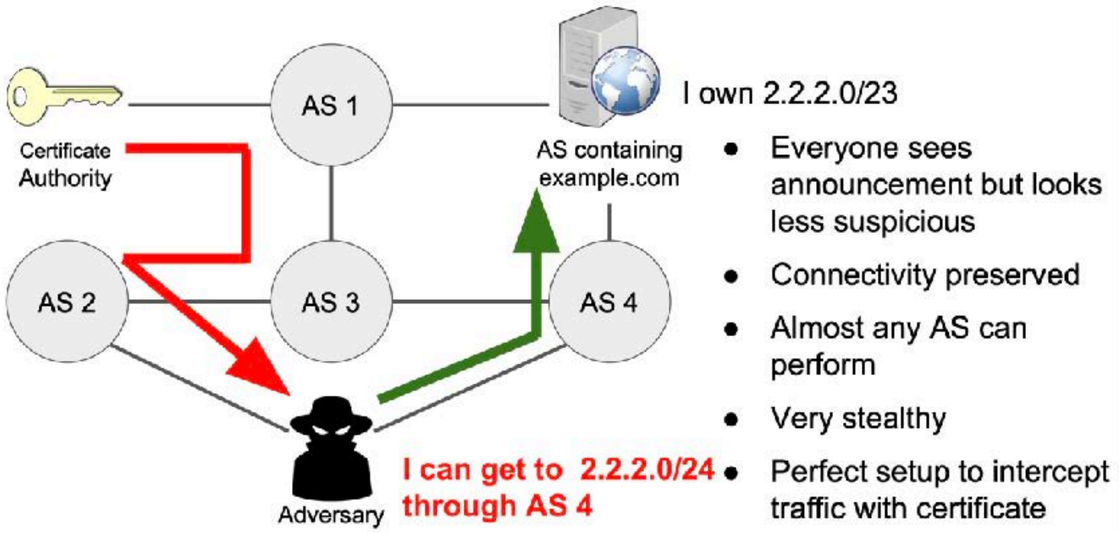
BGP Hijacking: AS Path Poisoning

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

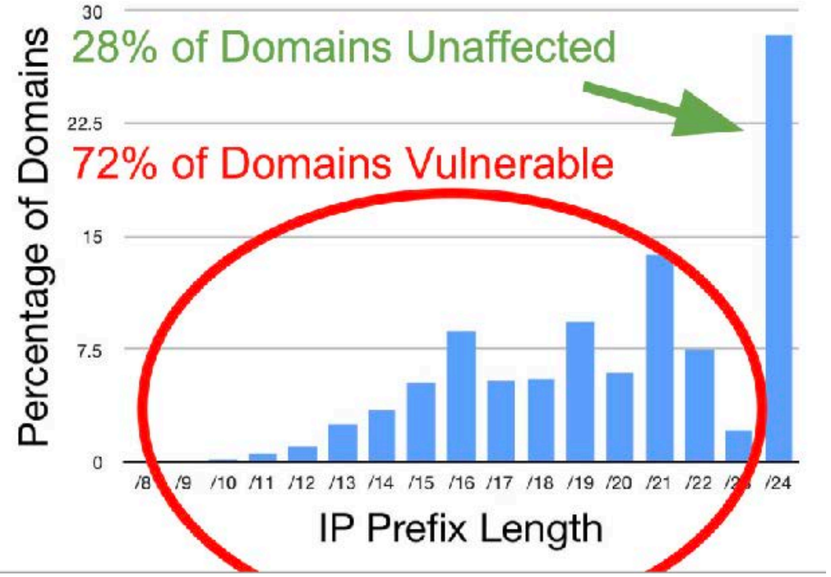


I own 2.2.2.0/23

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

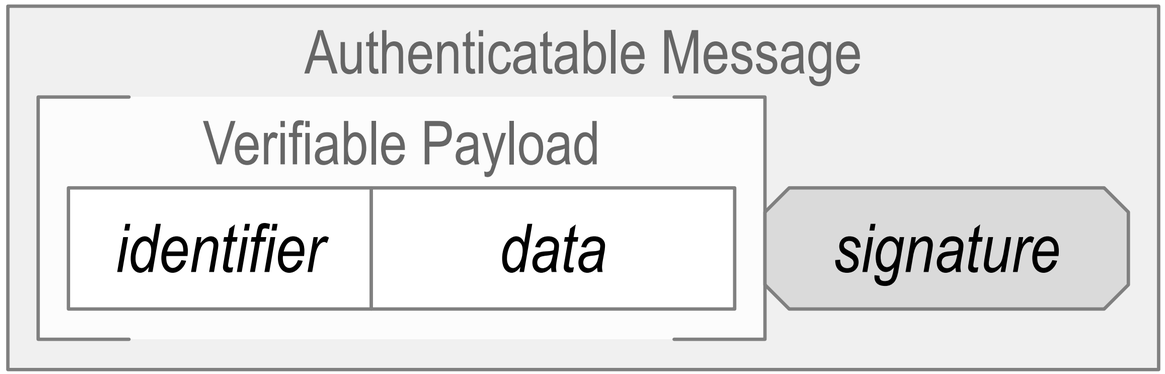
Vulnerability of domains: sub-prefix attacks

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



Identity (-ifier) System Security Overlay

Establish authenticity of IP packet's message payload.

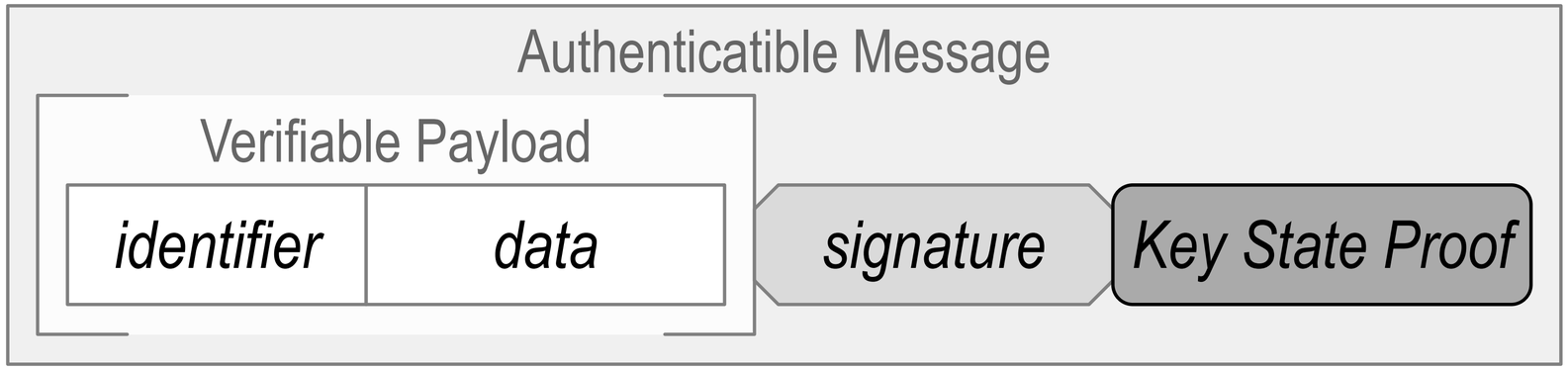


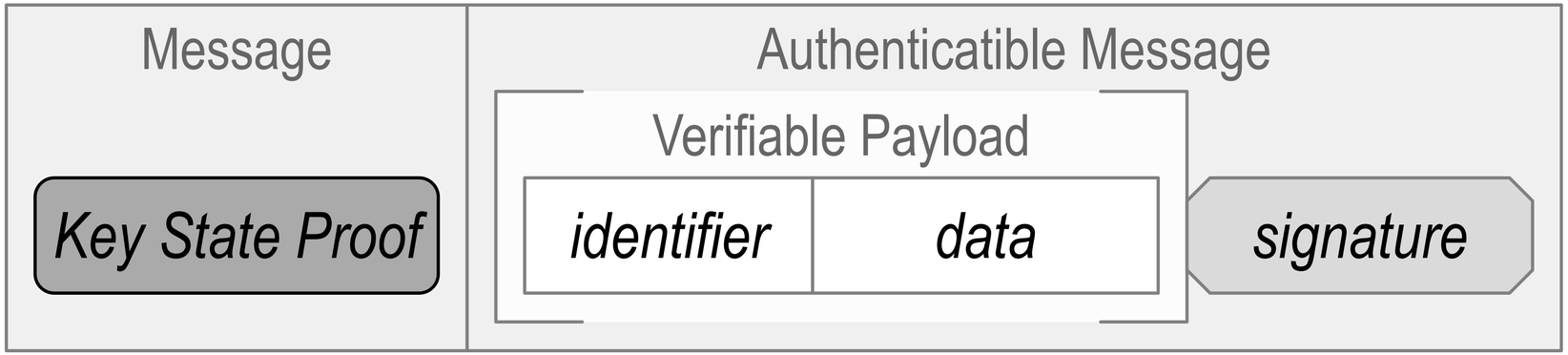
The overlay's security is contingent on the mapping's security.

Identity (-ifier) System Security Overlay

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

Establish authenticity of IP packet's message payload.

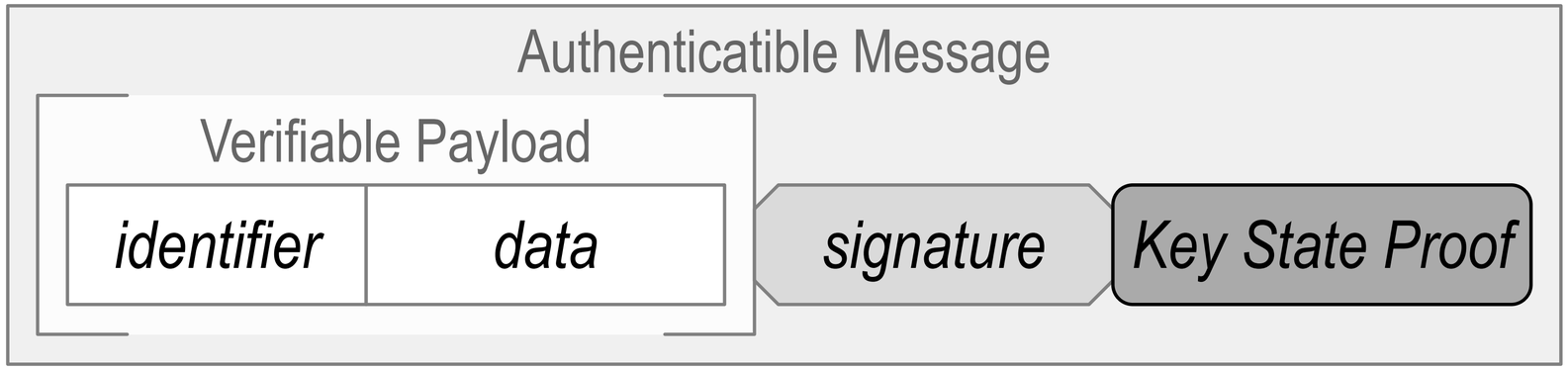


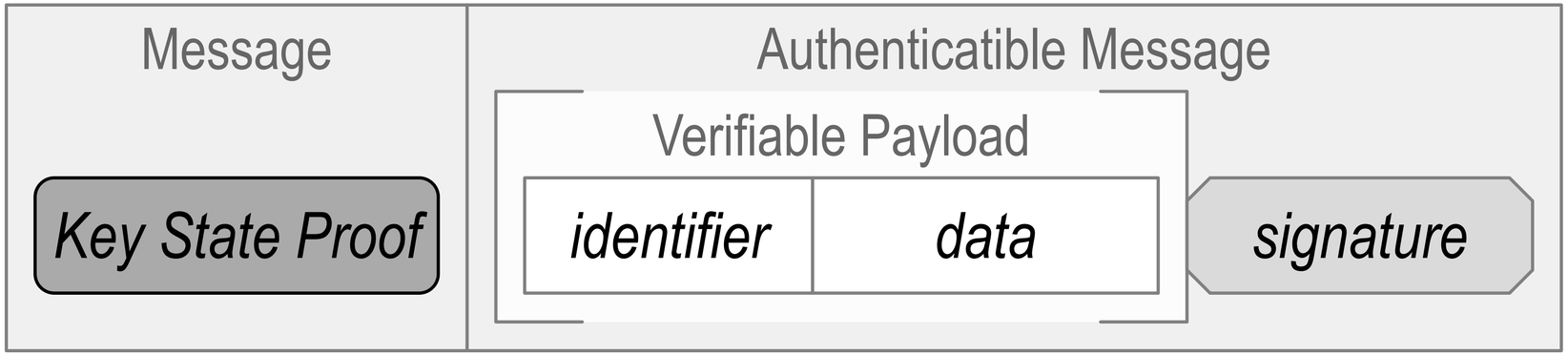


Identity (-ifier) System Security Overlay

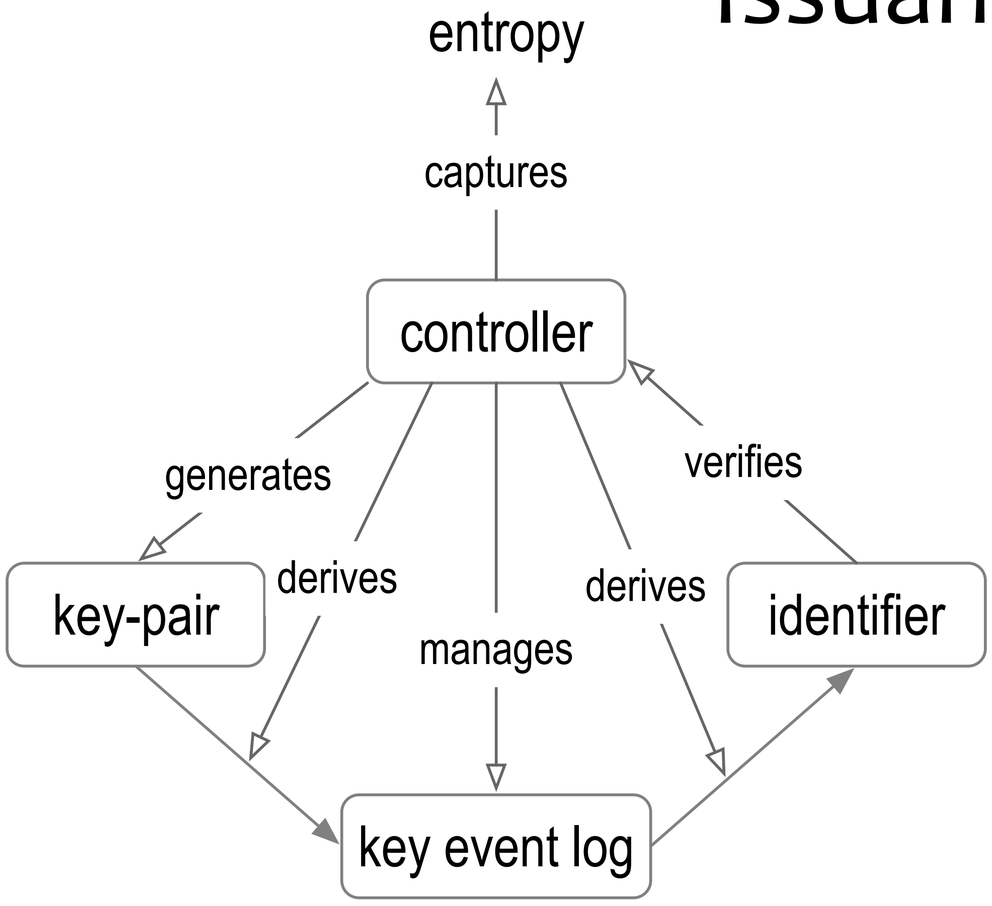
persistent (transferable) mapping = verifiable data structure of key state changes

Establish authenticity of IP packet's message payload.

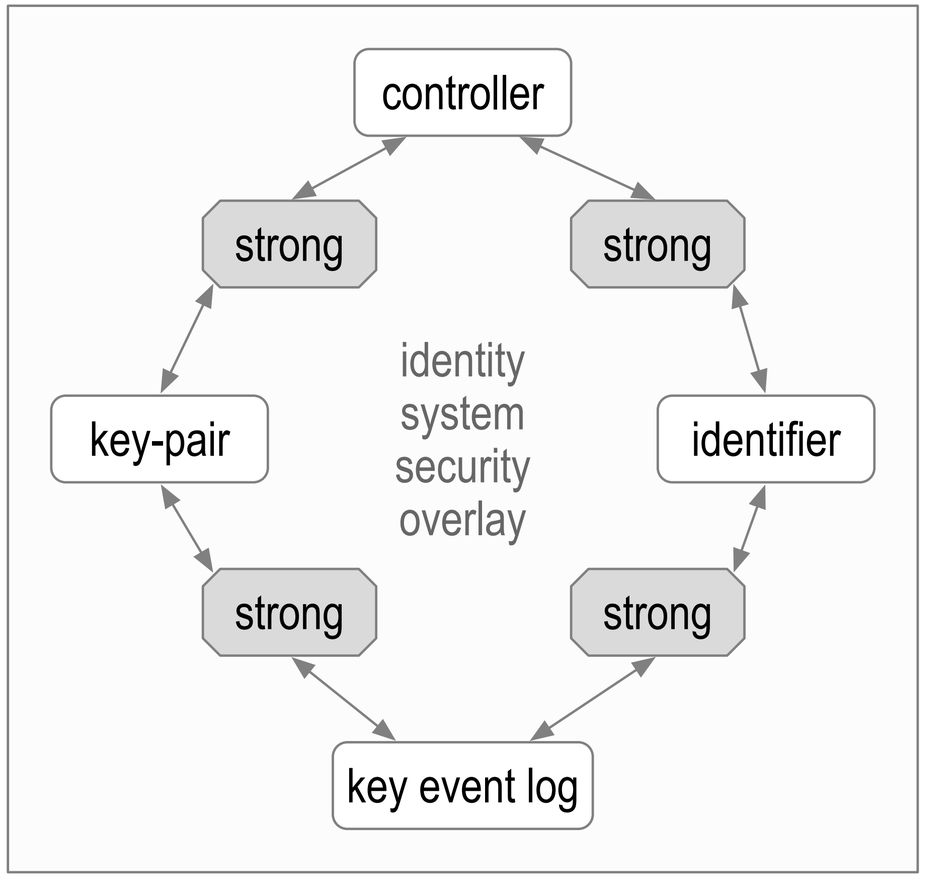




Autonomic Identifier (AID) (Persistent): Issuance and Binding



autonomic, self-managing

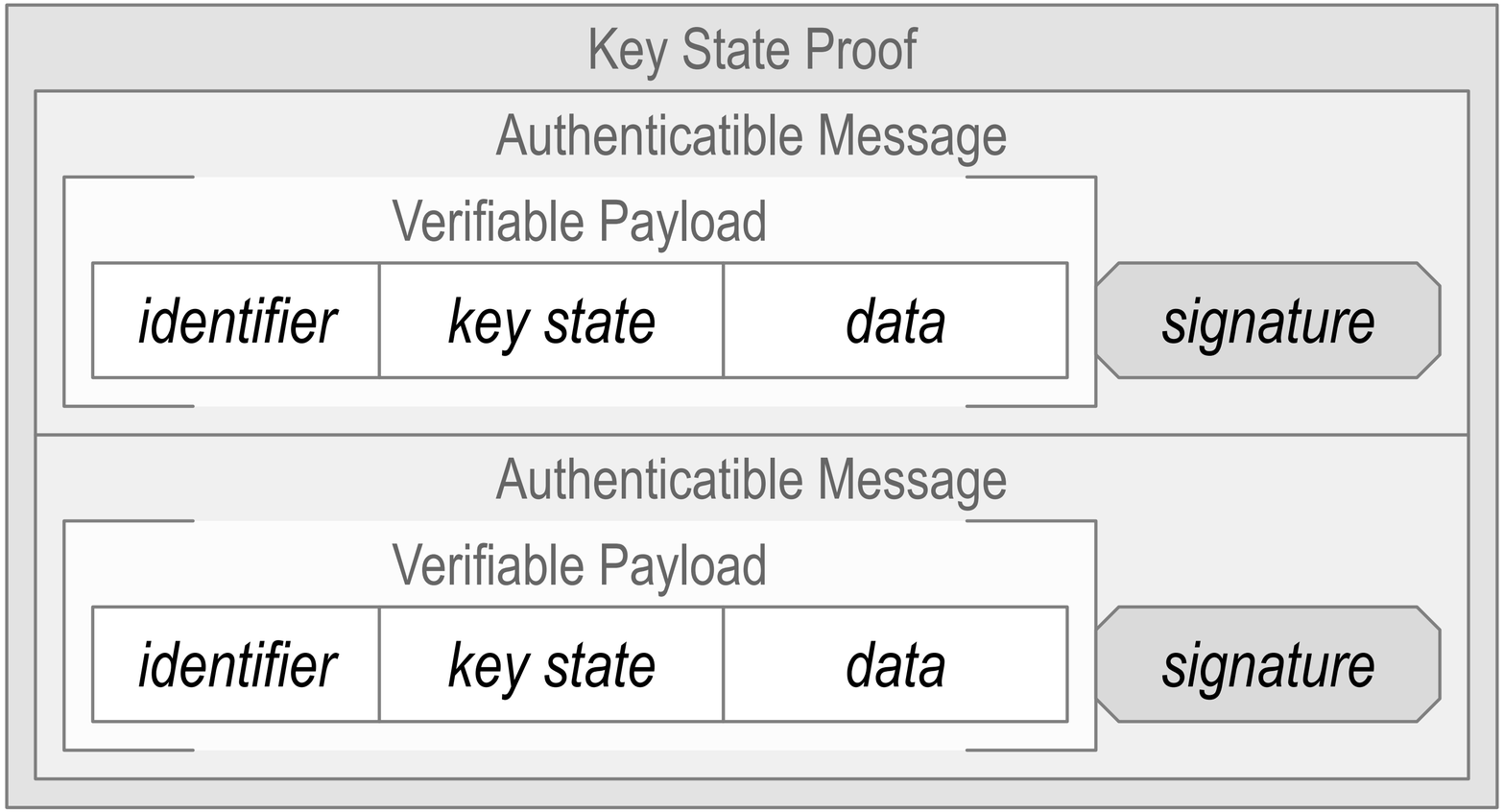


Autonomic Identifier Issuance Tetrad

cryptographic root-of-trust & verifiable persistent control

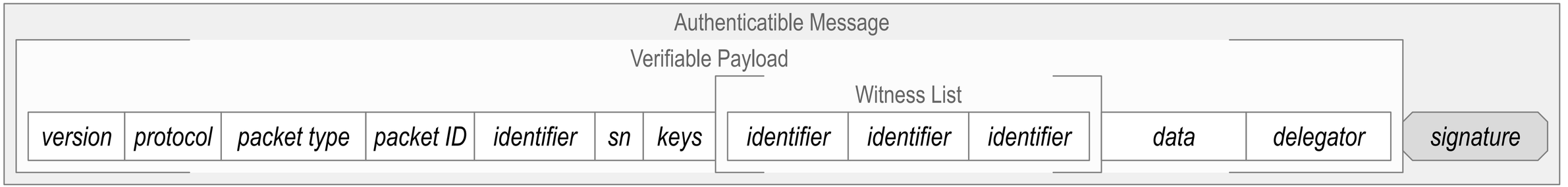
Key State Proof is Recursive Application of Overlay

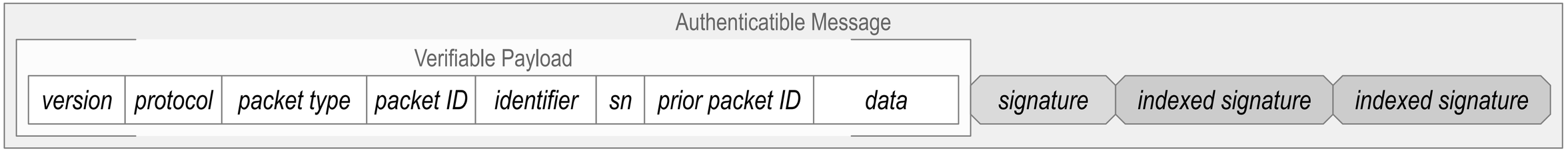
persistent (transferable) mapping = verifiable data structure of key state changes

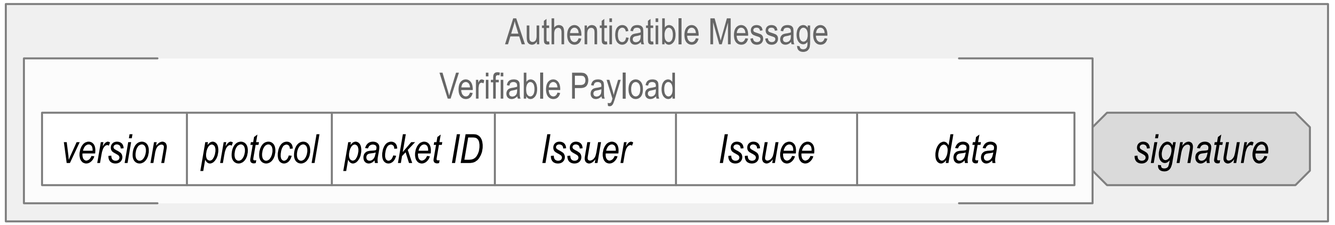


Wither *Destination* in Authentication Overlay

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







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)

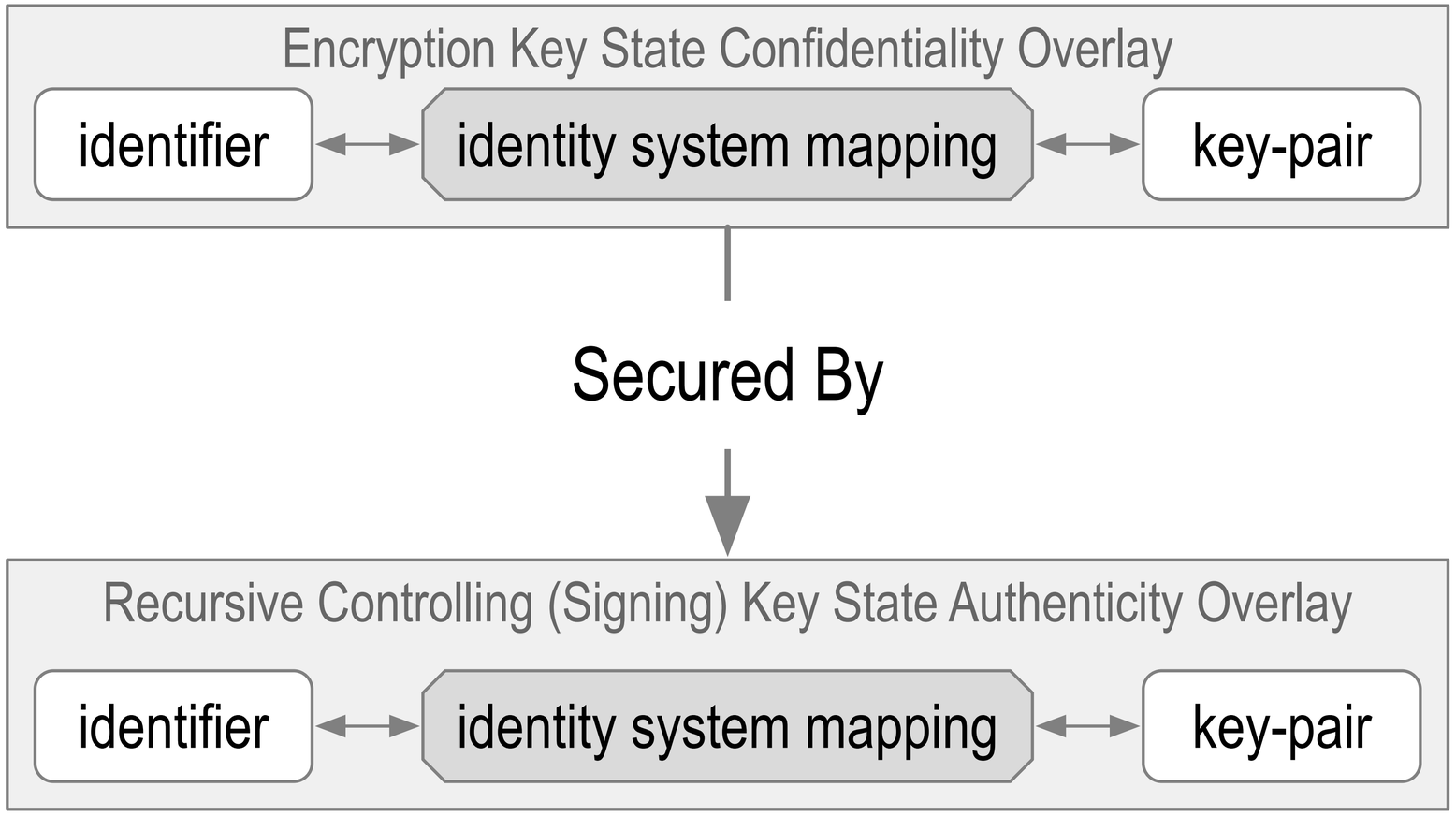
Can encrypt a message using the encryption public key to ensure that only the controller of the identifier can view it (confidentiality) (only-end-viewable restricted)

Confidentiality Overlay Dependency

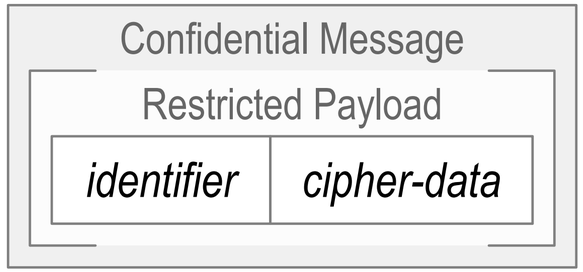
The encryption key state mapping can leverage the same persistent control mechanism as the signing key state mapping.

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.



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

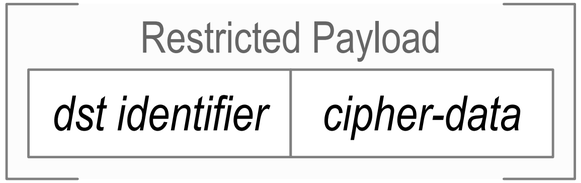
Detectable leakage (2nd party to 3rd party)

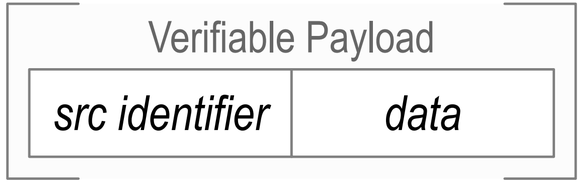
Detectable collusion (2nd party to 2nd party)

1st party non-view-ability (protects 1st party from liability, protects 2nd party from exploit of 1st party via shared secret)

1st party non-collude-ability by 2nd party (1st party to 1st party or 1st party to 3rd party)

Layering Confidentiality & Authenticatability Overlays

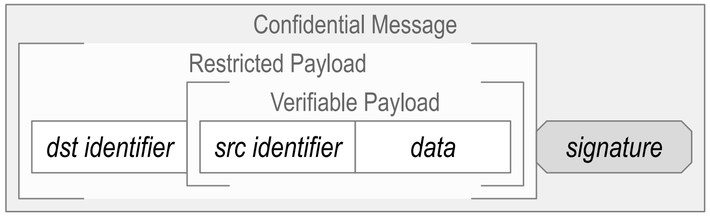


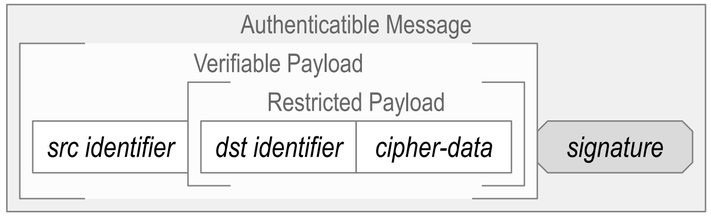


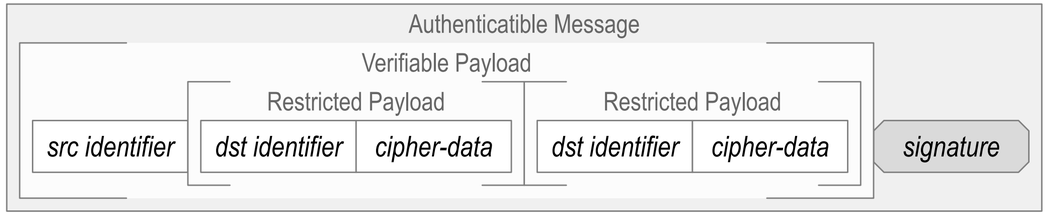
Setups matter! Order of layering is setup dependent, including the setups of any trusted intermediaries

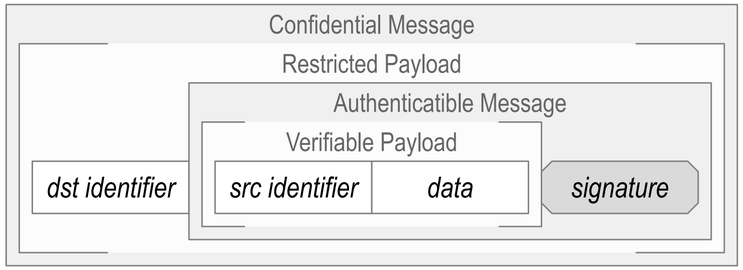
The presence or absence (explicit vs. implicit) of src and dst identifiers in a message is setup dependent

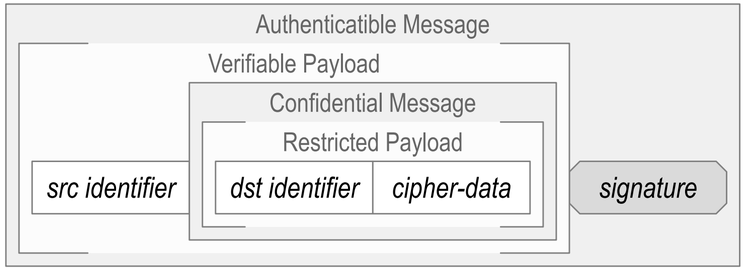
The hard problem of setups for persistent identifiers is to avoid repeating non-scalable manual setups like OOBA in order to ensure persistent control (solved by pre-rotated provenanced key state on cryptographic root-of-trust)











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

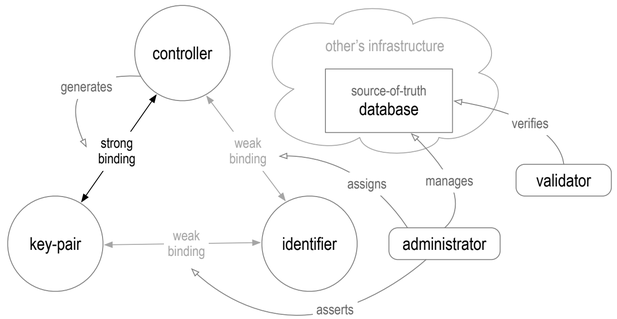
Given that maximal trust transitivity happens in an interaction when the trust bases of all parties are mutually appraisable with the same degree of trustability and at relative low appraisal cost …

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.

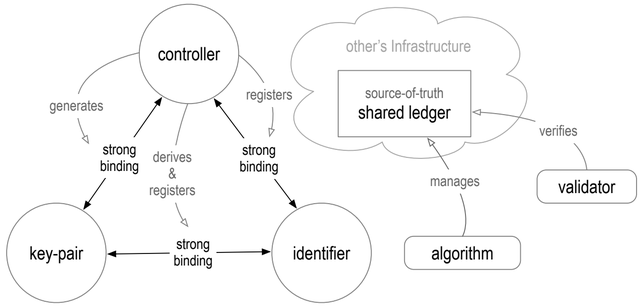
Is it not ludicrous to have a signed agreement where some parties use digital signatures, and some use HMACs?

Administrative
DNS/CA



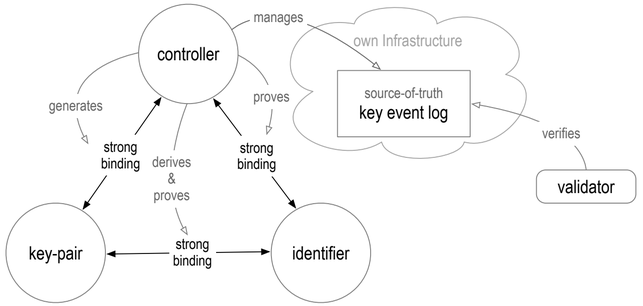
Opaque
Impractically High Appraisal Cost

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