Assertions and Tokens
+
Path tracing

SPIFFE/SPIRE

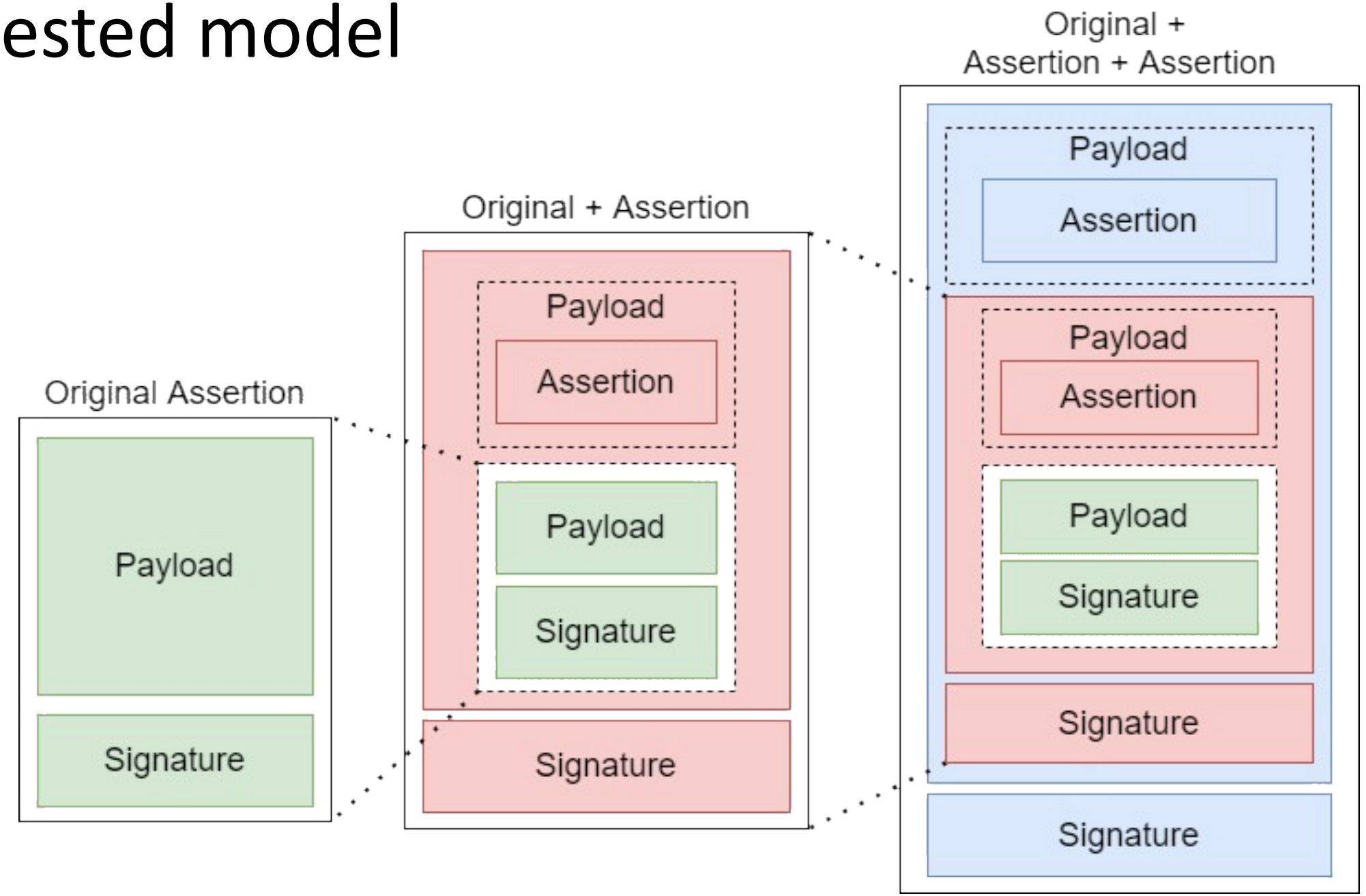
Out/2022

Recap

- **Nested model:** Allows appending new assertions to existing tokens

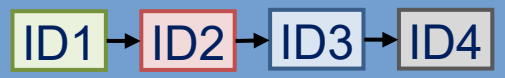
- **Token path tracing**: token path identification and validation
  - **Anonymous mode**: No ID associated to keys. Just path validation
  - **ID mode**: Each signature must be followed by necessary certificates to perform identification and validation

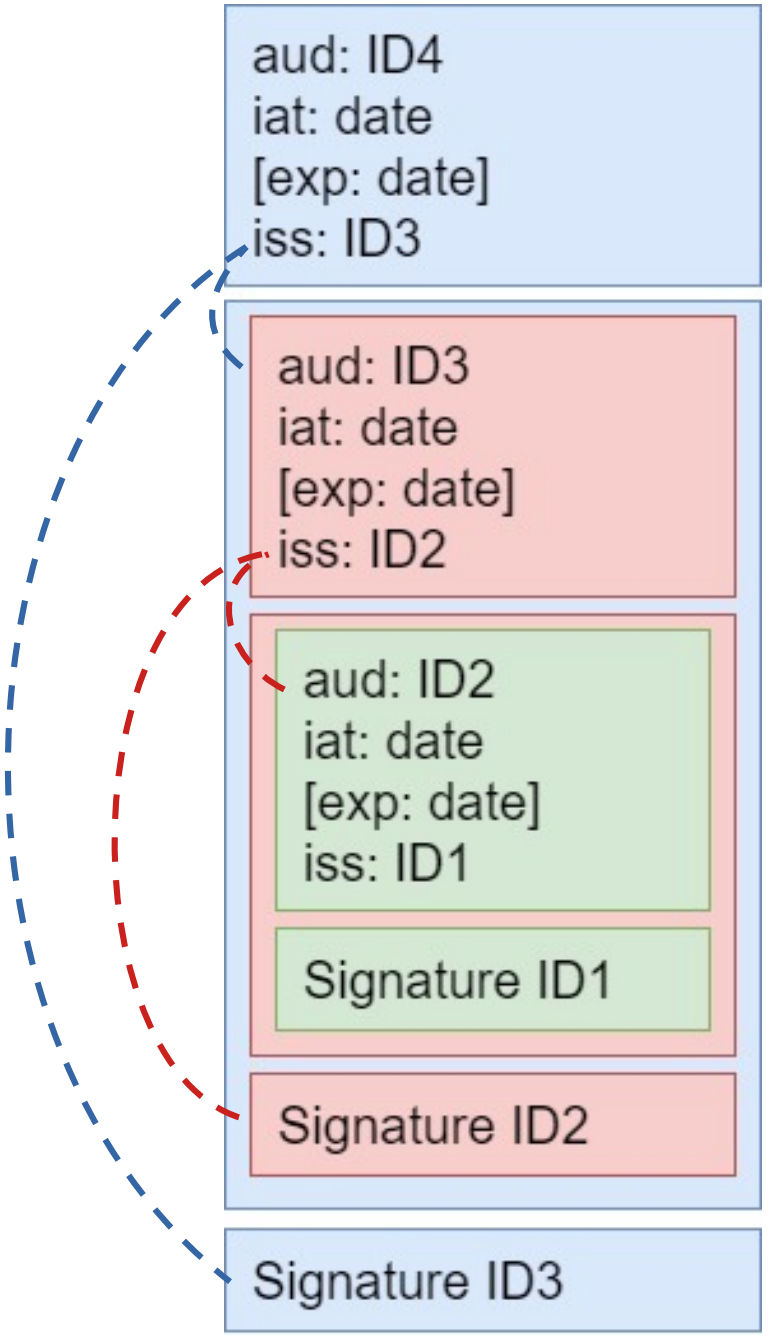
Nested model



Token tracing

Link between issuer and audience





Work so far...

Prototypes developed:

- **ECDSA-SVID:** uses SVID to sign/validate. Can use SPIFFE-ID or SVID as ID, adding to token or using IdP (available)
- **EdDSA-Schnorr**: uses Schnorr EdDSA standard signatures/validation
- **Schnorr Tracing model**: Schnorr EdDSA with issuer/audience validation This prototype uses a secret key as private key generator
- **Concatenated Schnorr**: uses part of previous signature as next private key, and Galindo-Garcia validation model

ECDSA – SVID (ID/Anonymous mode)

Developed solution: **Sign with SVID key, add SPIFFE-ID, SVID or public key to token**

- **Pros**
  - Off-line validation: token contain necessary certs
  - ID and Anonymous mode available
- **Cons**
  - ID mode requires more bandwidth
- **Possibilities**
  - Remove SVID from token and send it apart (ID artifact)
  - Use lightweight SVID
  - Also support anonymous mode (sending token without ID artifact)

EdDSA – Schnorr (anonymous mode)

Developed solution: **Use standard Schnorr signature and validation**

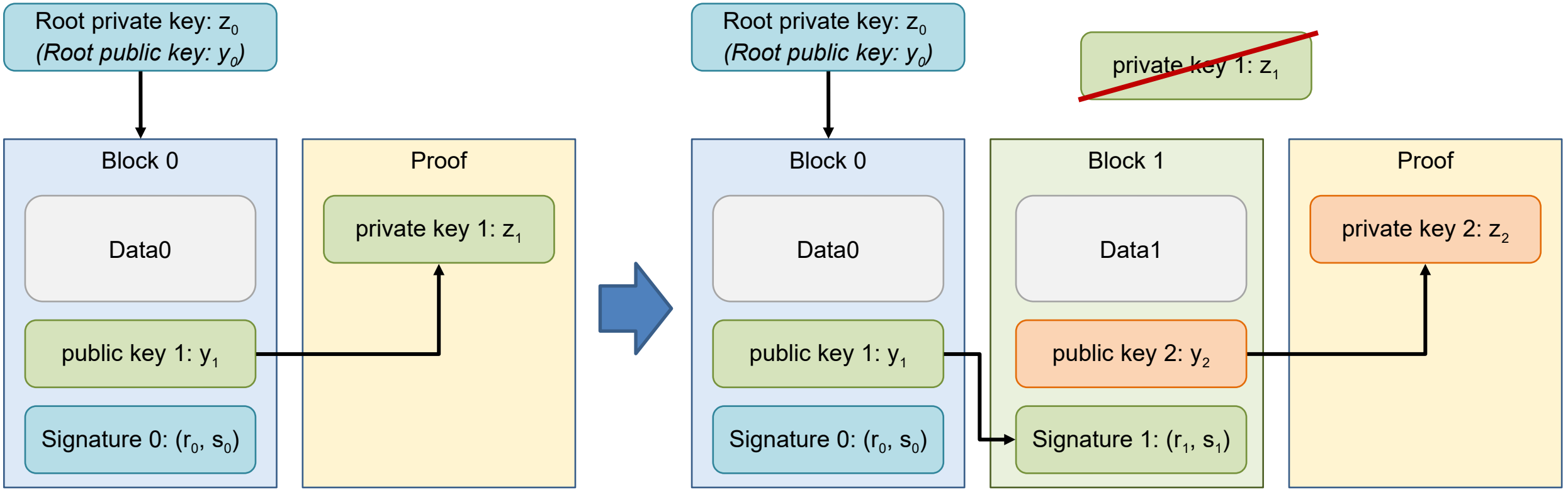
- **Pros**
  - Simpler construction
  - Slightly smaller signatures
- **Cons**
  - No identity model adopted
- **Possibilities**
  - Study of ECDSA – Schnorr possibility, allowing usage of SVID/Schnorr solution

Schnorr – Tracing model (anonymous mode)

Developed solution: **Each hop generates a private key, used by next hop to append assertions**

- **Pros**
  - Implement issuer/audience validation
- **Cons**
  - No identity model adopted
- **Possibilities**
  - Study of ECDSA – Schnorr possibility, allowing usage of SVID/Schnorr solution

Tracing model (biscuits-based model with Schnorr EdDSA)



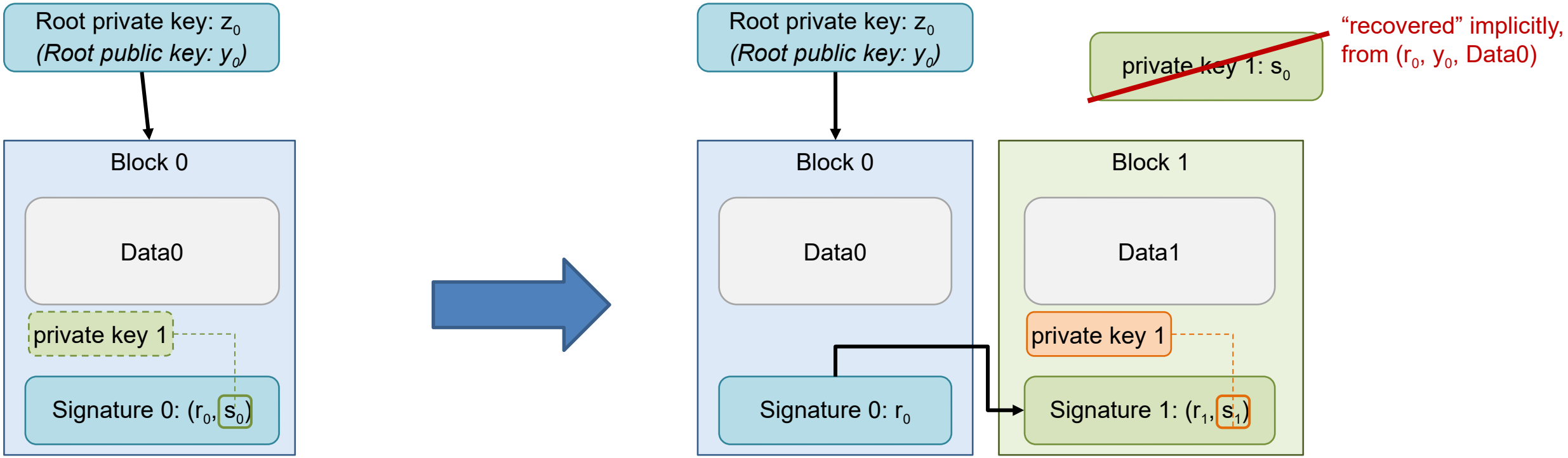
EdDSA – Concatenated Schnorr (anonymous mode)

Developed solution: **SchCo-biscuits**. Each hop uses part of previous signature as private key.

- **Pros**
  - Smaller token size when compared to std. model
  - Faster validation (using Galindo-Garcia) than sequencial model
  - Cryptographic-linked signatures
- **Cons**
  - No tracing capabilities
- **Possibilities**
  - Study aggregated signatures *state-of-art* to verify its viability

SchCo-Biscuits

(using concatenated Schnorr-based signatures: Galindo-Garcia-style)



Validation runtime and token size comparison

Token with 10 signatures	Std. Schnorr EdDSA	Concatenated Schnorr
1	15.666	8.806
2	16.057	15.548
3	19.031	7.823
4	8.724	12.274
5	18.621	14.156
6	15.904	8.223
7	17.341	11.199
8	13.056	14.249
9	10.706	9.473
10	9.559	8.149
Average runtime	**14.467**	**10.990**

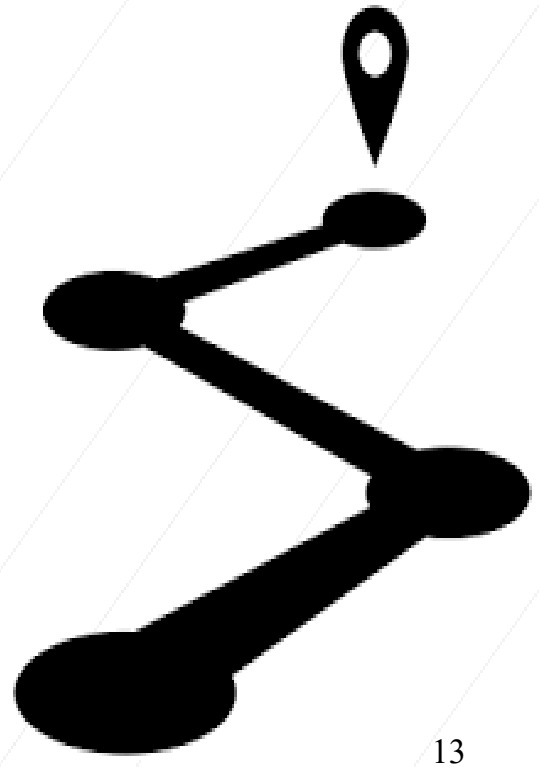
***milliseconds**

	Std. Schnorr EdDSA	Concatenated signatures
x1	196	196
x2	463	351
x3	730	506
x4	997	661
x5	1264	816
x6	1531	971

***bytes**

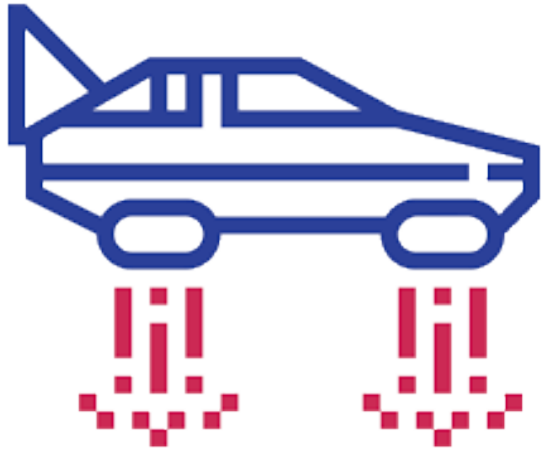
Next Steps

- Add proxy and more middle-tiers to PoC scenario
- Implement anonymous/ID mode in PoC application
- Generate assertions using SPIRE selectors
- General solution benchmarks



Future Work

- Specify and implement lightweight SVID
- Identity-based SVID: lightweight SVID with Galindo-Garcia
- Develop Biscuits prototype, with support to Galindo-Garcia
- Protobuf / JSON analysis




WHO? WHERE?
WHEN? WHY? HOW?
HOW? WHAT? WHO?
WHERE?
WHO? WHERE?
WHAT?
WHY? WHAT?
WHEN?
WHERE? WHO?
HOW? WHAT?
WHERE?
WHY?
WHO? WHEN?
WHEN? WHAT?
WHO? WHY? HOW?
HOW? WHERE?
WHY?
WHO? WHAT?
HOW?
WHAT? WHEN?
HOW? WHEN?
WHAT?
WHY? WHERE?
WHEN? HOW?
WHERE?