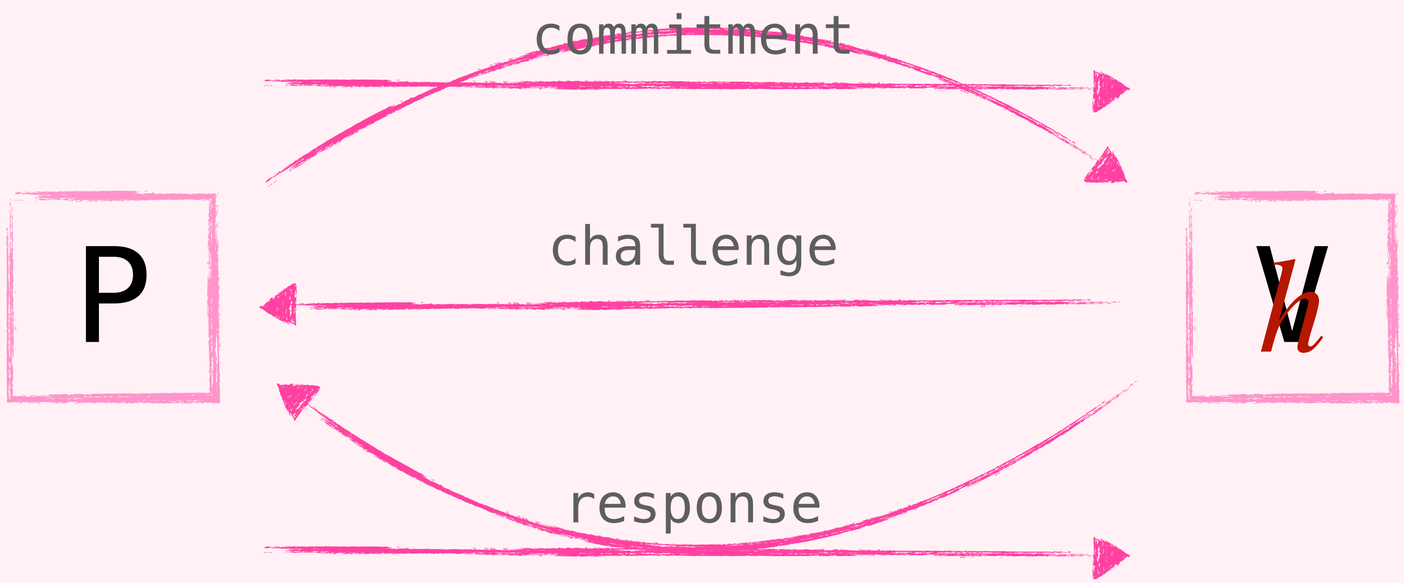
Σ-Protocols and Fiat-Shamir

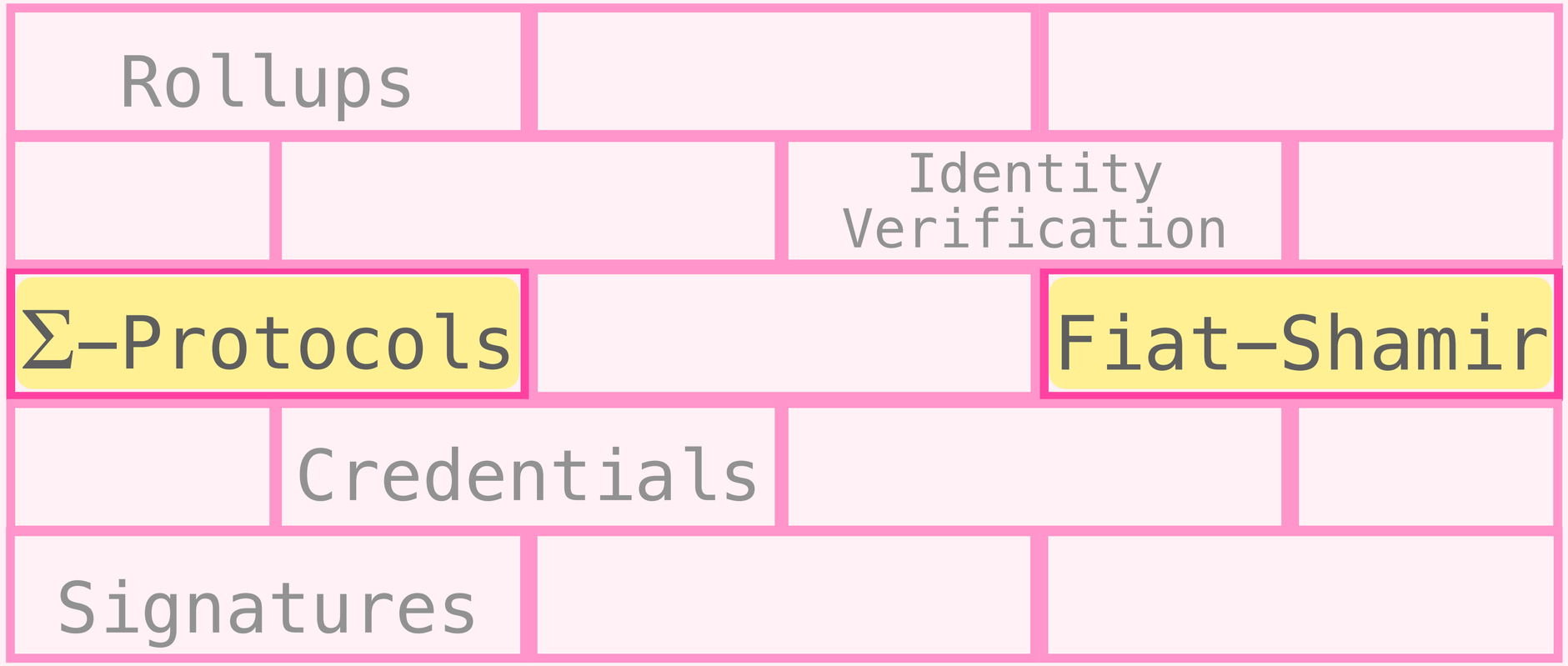
Michele Orrù, Cathie Yun, Vishruti Ganesh
IETF 124 - Crypto Forum

Interactive Proofs



- draft-irtf-cfrg-sigma-protocols-01
- draft-irtf-cfrg-fiat-shamir-01

Zero-Knowledge Proofs



Specs & Implementations

Interactive Sigma Proofs
draft-irtf-cfrg-sigma-protocols-01

Interactive Σ-Protocol Spec

Fiat-Shamir Transformation
draft-irtf-cfrg-fiat-shamir-01

Fiat-Shamir Transform Spec

mmaker / **draft-irtf-cfrg-sigma-protocols**

Python/SAGE Reference Implementation

arkworks-rs / **spongefish**

Fiat-Shamir Implementation

sigma-rs / **sigma-proofs**

Spec-compatible Rust Implementation
with extended functionalities

What happened since IETF123

Security Review

By OpenZeppelin Security: 3 Notes, 4 Low, 1 Medium

Anonymous Rate-Limited Credentials
draft-yun-privacypass-crypto-arc-00

Abstract

This document specifies the Anonymous Rate-Limited Credential (ARC) protocol, a specialization of keyed-verification anonymous credentials with support for rate limiting. ARC credentials can be presented from client to server up to some fixed number of times, where each presentation is cryptographically bound to client secrets and application-specific public information, such that each presentation is unlinkable from the others as well as the original credential creation. ARC is useful in applications where a server needs to throttle or rate-limit access from anonymous clients

Adoption

Anonymous Credit Tokens
draft-schlesinger-cfrg-act-00

Abstract

This document specifies Anonymous Credit Tokens (ACT), a privacy-preserving authentication protocol that enables numerical credit systems without tracking individual clients. Based on keyed-verification anonymous credentials and privately verifiable BBS-style signatures, the protocol allows issuers to grant tokens containing credits that clients can later spend anonymously with that issuer.

Ciphersuites

```
class NISigmaProtocol:
    Protocol: SigmaProtocol
    Hash: DuplexSpongeInterface
    Codec: Codec
```

Ciphersuites

```
class NISchnorrProofShake128P256(NISigmaProtocol):
    Protocol = SchnorrProof
    Hash = SHAKE128
    Codec = P256Codec
```

```
class NISigmaProtocol:
    Protocol: SigmaProtocol
    Hash: DuplexSpongeInterface
    Codec: Codec
```

```
class NISchnorrProofShake128Bls12381(NISigmaProtocol):
    Protocol = SchnorrProof
    Hash = SHAKE128
    Codec = Bls12381Codec
```

```
class NISchnorrProofKeccakDuplexSpongeBls12381(NISigmaProtocol):
    Protocol = SchnorrProof
    Hash = KeccakDuplexSponge
    Codec = Bls12381Codec
```

In other specs (simplified):

ARC:

```
// Proof of knowledge that x1 such that
// X1 = x1 * generatorH
statement.append_equation(
    X1Var, [(x1Var, genHVar)]
)
prover = NISchnorrProofShake128P256(statement)
```

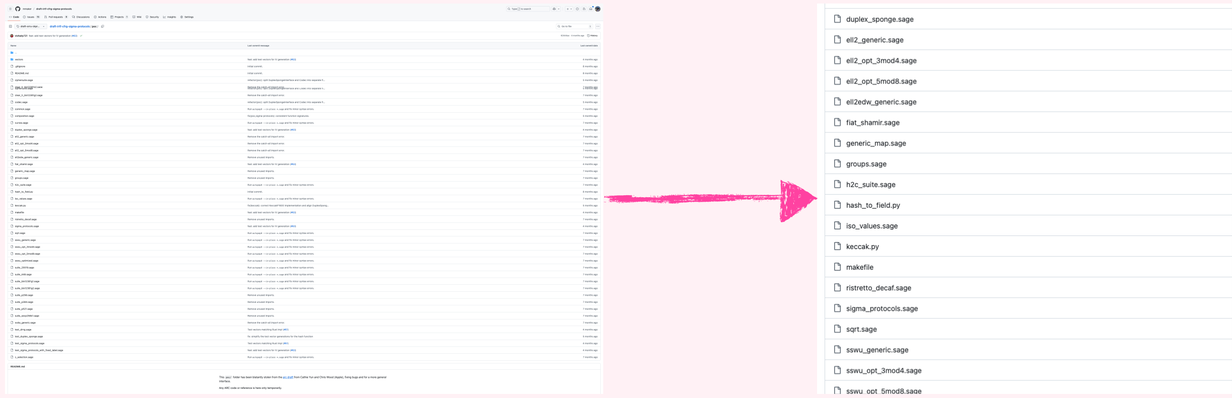
ACT:

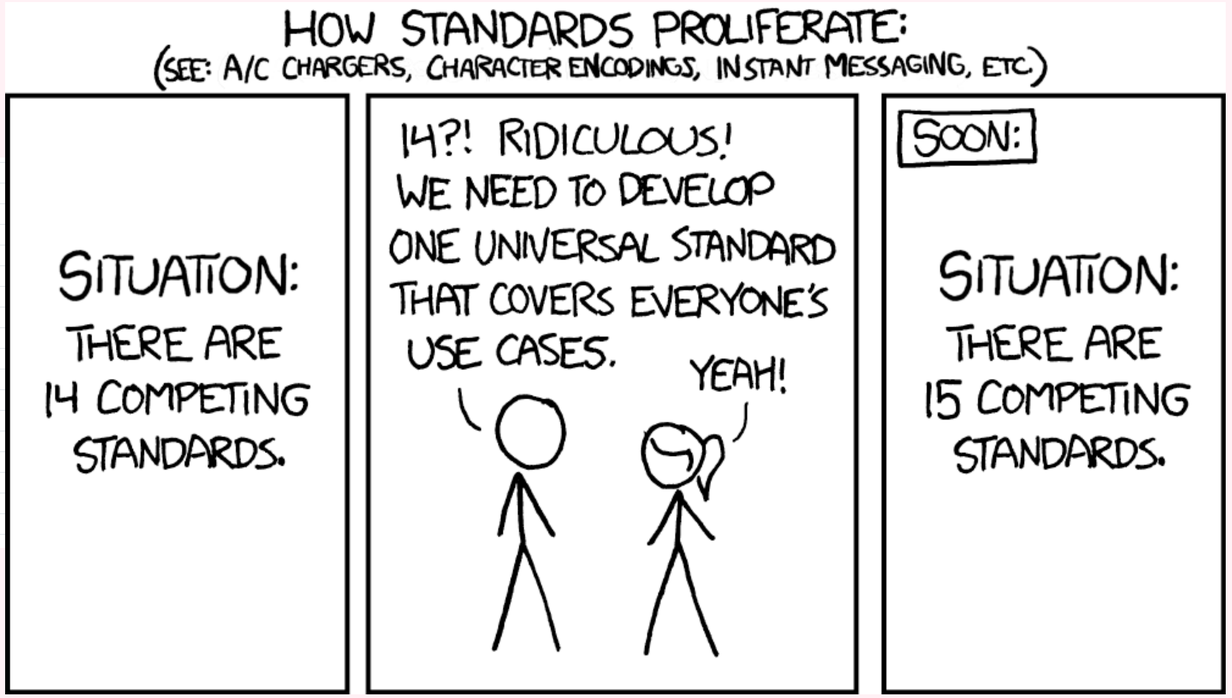
```
// Proof of knowledge of (k, r) such that
// K = H2 * k + H3 * r
statement.append_equation(
     KVar, [(kVar, genH2Var), (rVar, genH3Var)]
)
prover = NISigmaProtocol(statement)
```

You can define your own ciphersuite according to your use-case!

Clearing up the technical debt

- Elliptic Curve definitions:
  - The BBS Signature Scheme (draft-irtf-cfrg-bbs-signatures)
  - The ristretto255 and decaf448 Groups (RFC 9496)
  - Hashing to Elliptic Curves (RFC 9380)
- Reference implementations inherit SAGE





- A python replacement? PR#76

```
git+https://github.com/mmaker/draft-irtf-cfrg-sigma-protocols#subdirectory=poc
```

Next Steps and Open Questions

Next: Formal Verification

We want your help!

How would you like to use Σ-Protocols and Fiat-Shamir?

See Rust implementation for extensions and optimizations

Σ-Protocols and Fiat-Shamir

- draft-irtf-cfrg-sigma-protocols
- draft-irtf-cfrg-fiat-shamir

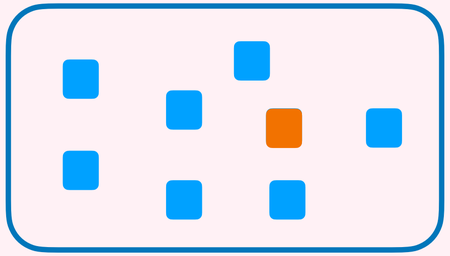
cathieyun@gmail.com
m@orru.net
vishruti721@gmail.com

Appendix

Feedback Thread: OR Composition of Σ-Protocols

(NOT in spec)

P



$z_{i\neq5}$ = simulate_response()

$y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8$

$c_{i\neq5}$ = sample_random()

c

V

$c_5 = c - \sum_{i\neq5} c_i$

$z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8$

$y_{i\neq5}$ = simulate_commitment()

Feedback Thread: Compressed Σ-Protocols

(NOT in spec)