Cross-group Σ-protocols

Yevhen Hrubiian

May 2, 2025

Distributed Lab

Introduction

Schnorr identification protocol

Suppose $\mathbb{G}$ is a cyclic group of order q with a generator G. Then, the relation and language being considered are:

$$\mathcal{R} = \{(Q, x) \in \mathbb{G} \times \mathbb{Z}_q : Q = [x]P\},\ \mathcal{L}_{\mathcal{R}} = \{Q \in \mathbb{G} : \exists x \in \mathbb{Z}_q : Q = [x]P\}$$

Problem #1
$\mathcal{P}$ wants to convince $\mathcal{V}$ that it knows the discrete log of $Q \in \mathcal{L}_{\mathcal{R}}$. That is, he knows *witness* x such that $(Q, x) \in \mathcal{R}$.

Problem #2
We want to obtain the following properties:

- **Completeness**: $\forall (Q, x) \in \mathcal{R}$ verifier outputs **true**
- **Soundness**: $\forall (Q, x) \notin \mathcal{R}$ verifier outputs **true** with negligible probability.
- **Zero-knowledge**: Interaction between Prover and Verifier gives no new information about witness.

Protocol Flow

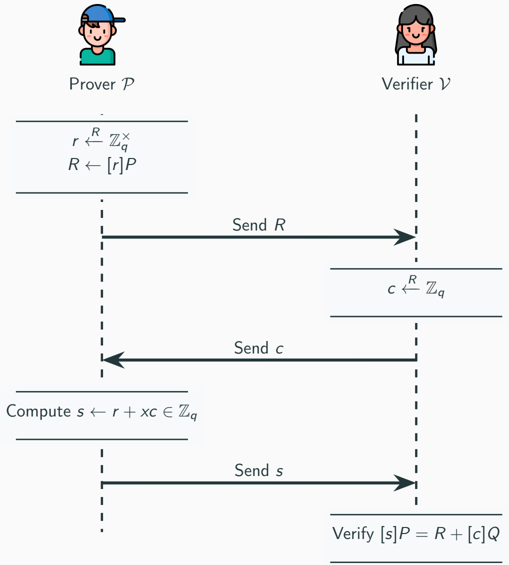


Figure 1: Prover convinces Verifier that he knows $x : Q = [x]P$

Properties of Protocol

Schnorr identification protocol has the following properties

- **Completeness**: $\forall (Q, x) \in \mathcal{R}$ verifier outputs **true**
- **Special(knowledge) Soundness**: There exist efficient *extractor* algorithm $\mathcal{E}$ such that given statement Q and two accepting conversations $(R, c, s), (R, c', s'), c \neq c'$ $\mathcal{E}$ can extract witness x such that $(Q, x) \in \mathcal{R}$ except with probability ϵ
- **Honest verifier zero-knowledge**: There exists an efficient *simulator* algorithm *Sim*, which on input Q and a challenge $c \in C$, outputs transcripts of the form (R, c, s) whose distribution is indistinguishable from accepting protocol transcripts generated by real protocol runs on public input Q and with challenge c
- Schnorr identification protocol is a member of a larger class of interactive protocols called Σ-**protocols**

Note:
Honest verifier zero-knowledge is not **zero-knowledge**! Malicious verifier has ability to extract the witness from transcript.

Σ-protocols

Definition
Let $\mathcal{R}$ be a binary relation and let $(Y, w) \in \mathcal{R}$. An interactive two-party protocol specified by algorithms (P_1, P_2, V) is called a **Σ-protocol** for $\mathcal{R}$ with challenge set C, public input Y , and private input w, if and only if it satisfies the following conditions:

- **3-move form**: The protocol is of the following form:
  1. The prover computes $(R, r) \leftarrow P_1(w, Y)$ and sends R to the verifier, while keeping st secret.
  2. The verifier draws $c \xleftarrow{R} C$ and returns it to the prover.
  3. The prover computes $s \leftarrow P_2(w, Y, c, r)$ and sends s to the verifier.
  4. The verifier accepts the protocol run, if and only if $V(Y, R, c, s) = \mathit{true}$, otherwise it rejects.
- Has **Correctness, Special Soundness, Honest verifier zero-knowledge**

Σ-protocol for knowledge of preimage of homomorphism

Σ-protocols could prove much larger class of binary relations, more specifically they could prove knowledge of preimage of arbitrary homomorphism $\phi : \mathbb{Z}_q^m \to \mathbb{G}^n$:

$$\mathcal{R} = \{(\mathrm{Y}, \mathrm{w}) \in \mathbb{G}^n \times \mathbb{Z}_q^m : \mathrm{Y} = \phi(\mathrm{w})\}, \mathrm{Y} = (Y_1, \ldots, Y_n), \mathrm{w} = (w_1, \ldots, w_m)$$

Protocol algorithms:

- $P_1(\mathrm{w}, \mathrm{Y})$ samples $\mathrm{r} = (r_1, \ldots, r_m) \overset{R}{\leftarrow} Z_q^m$, computes $\mathrm{R} = \phi(\mathrm{r})$, outputs (R, r)
- $P_2(\mathrm{w}, \mathrm{Y}, c, \mathrm{r})$ outputs $\mathrm{s} = \mathrm{w} + c \cdot \mathrm{r}$ $(\forall i = 1..m : s_i = w_i + cr_i)$
- $V(\mathrm{Y}, \mathrm{R}, c, \mathrm{r})$ checks $\mathrm{R} + c\mathrm{Y} \overset{?}{=} \phi(\mathrm{s})$

Σ-protocols could be made non-interactive using Fiat-Shamir heuristic: $c = \mathcal{H}(\mathcal{D}, \mathrm{Y}, \mathrm{R})$ where $\mathcal{D}$ - domain separator, $\mathcal{H}$ - random oracle modeled with collision-resistant hash function. Zero-knowledge property holds for non-interactive Σ-proofs

Σ-protocols standardization

There is two main types of non-interactive Σ-proofs:

- Batchable proof: $\pi = (\mathsf{R}, \mathsf{s})$
- Compact proof: $\pi = (c, \mathsf{s})$

Σ-protocols currently are being under standardization process:
`https://sigma.zkproof.org`

Most common notation for Σ-protocols is Camenisch-Stadler notation.
For example Chaum-Pedersen protocol for DH-triplets:

```
DLEQ(A, B, G, H) = {
    (x): A = (x * G), B = (x * H)
}
```

Moreover, we can prove knowledge of witness for arbitrary arithmetic circuit(possibly encoded in R1CS) with Σ-protocol, however proof size is $O(n + m)$, where n-number of constraints, m-number of variables

Cross-group Σ protocols

Cross-group protocols

Often, it is useful to prove knowledge of the same witness across different groups:

- In credential linking: anonymous credentials(KVAC) issued to the same user in different cryptographic groups, *Signal* group system uses KVAC.
- Proof of assets between different chains: often privacy-based solutions use Pedersen commitments so proving that committed value is equal would be useful. Also linking on-chain pairing-unfriendly cryptography to some pairing-friendly group is useful when proving on-chain statements with pairing-based SNARKs.
- Linking commited value to native scalar in proving systems like *bulletproofs*. It's very useful when proving consistency of update of *Assymetric Ratchet Tree*

Proving equality of committed witness

Suppose that one want to prove relation with witness-committed schemes as *bulletproofs or halo*:

$$\mathcal{R}_{R1CS} = \{(A, B, C, V; \mathrm{z}) | (A\mathrm{z}) \circ (B\mathrm{z}) = (C\mathrm{z})\}$$

where $V = [x]G + r[H] \in \mathbb{G}_1$ - committed witness in proving system group, $\mathrm{z} = (x, \mathrm{w})$ - extended witness, suppose that x should also satisfy $Q = [x]G_2 \in \mathbb{G}_2$ where $\mathbb{G}_2$ - some proving-native elliptic curve group(e.g. defined over F_n where $n = |\mathbb{G}_1|$). So we have to prove

$$\mathcal{R}_{comeq} = \{((Q, V); (x, s)) | Q = [x]G_1, V = [x]G_2 + [s]H_2\}$$

Where $x \in \mathbb{Z}, s \in \mathbb{Z}_q, G_1 \in \mathbb{G}_1, G_2, H_2 \in \mathbb{G}_2$

Cross-group Σ-protocols

Let us formalize our cross-group protocol. Briefly, idea of the protocol came from `https://eprint.iacr.org/2022/1593.pdf`.

Cross-group relation
Let $\mathbb{G}_1, \mathbb{G}_2$ - cryptographic groups of prime orders respectively p, q and $\phi : \mathbb{Z}_p^{n_1} \to \mathbb{G}_1^{m_1}, \psi : \mathbb{Z}_q^{n_2} \to \mathbb{G}_2^{m_2}$ - homomorphisms. Denote $\iota_p : \mathbb{Z} \to \mathbb{Z}_p, \iota_p(a) = a \pmod{p}$.

Assume prover wants to convince verifier that he knows preimage of each homomorhism given some elements of each preimage might be equal: $\mathrm{w}_1 = (\iota_p(\mathrm{x}), \mathrm{r}_1), \mathrm{w}_2 = (\iota_q(\mathrm{x}), \mathrm{r}_2)$ where x - common part of witness in both groups. We build proving system Π_{cross} for the following relation:

$$\mathcal{R}_{cross} = \{((\mathrm{Y}_1, \mathrm{Y}_2), (\mathrm{w}_1, \mathrm{w}_2)) \in (\mathbb{G}_1^{n_1} \times \mathbb{G}_2^{n_2}) \times (\mathbb{Z}_p^{m_1} \times \mathbb{Z}_q^{m_2}) : \\ \mathrm{Y}_1 = \phi(\mathrm{w}_1), \mathrm{Y}_2 = \psi(\mathrm{w}_2)\}$$

Parametrization

The construction of our protocol is based on combination of Σ-protocols with aborts and range-proofs(we implemented it with *bulletproofs*). Intuitively, it will abort if prover «leak» any information about the witness, if so, the parties begin protocol from scratch.

Let $B_g = \lceil \log_2(min(p, q)) \rceil$ - bitlength of minimal prime. Our protocol is parametrized by B_x, B_c, B_f so that $B_x + B_c + B_f < B_g$ (no modular reduction occurs in $\mathbb{Z}_p, \mathbb{Z}_q$ during proving!). Here B_x - maximum bitlength of cross-group scalar, B_c - bitlength of challenge which controls the soundness error ϵ, B_f - value that defines the probability of abort: $Pr[\mathcal{P} \text{ aborts}] = 1/B_f$.

For example, convenient choice for parameters when $B_g \approx 256$ (elliptic curve groups for 128bit security level) is:

$$B_x = 64, B_c = 128, B_f = 56$$

Π_{cross} protocol

Prover $\mathcal{P}$

Verifier $\mathcal{V}$

$\pi \leftarrow \mathcal{P}_{range}(\mathsf{x}, B_x)$

$\mathsf{r} \xleftarrow{R} \mathbb{Z}_p^{n_1}, \mathsf{R} \leftarrow \phi(\mathsf{r})$

$\mathsf{s} \xleftarrow{R} \mathbb{Z}_q^{n_2}, \mathsf{S} \leftarrow \psi(\mathsf{s})$

Send $(\pi, \mathsf{R}, \mathsf{S})$ $\longrightarrow$

$c \xleftarrow{R} \mathbb{Z}^*_{2^{B_c}}$

$\longleftarrow$ Send c

$\sigma_p \leftarrow \mathsf{r} + \mathsf{w}_1 c \in \mathbb{Z}_p$

$\sigma_q \leftarrow \mathsf{s} + \mathsf{w}_2 c \in \mathbb{Z}_q$

Abort if $(\sigma_p, \sigma_q) \notin \{2^{B_c+B_x}, \ldots, 2^{B_c+B_x+B_f} - 1\}^{n_1 \times n_2}$

Send (σ_1, σ_2) $\longrightarrow$

Verify range proof $\mathcal{V}_{range}(\pi, B_x)$

Verify $\phi(\sigma_1) = \mathsf{R} + c \cdot \mathsf{Y}_1$

Verify $\psi(\sigma_2) = \mathsf{S} + c \cdot \mathsf{Y}_2$

Completeness

Theorem (completeness)
Protocol Π_{cross} is 2^{-B_f}-**complete** for relation $\mathcal{R}_{cross}$.

Proof idea: each response σ_i is uniformly distributed in $Z_0 = \{cw_i, cw_i + 1 \dots, cw_i + 2^{B_c+B_x+B_f} - 1\}$ so the probability of non-aborting is $Pr[\mathcal{P}$ not aborts$] = \frac{|\{2^{B_c+B_x}, \dots, 2^{B_c+B_x+B_f}-1\}|}{|Z_\mathbf{0}|} = 1 - \frac{1}{2^{B_f}}$ and $Pr[\mathcal{P}$ aborts$] = 1/|2^{B_f}|$

If prover does not abort all verification equations pass since $0 \leq c < 2^{B_c}$, so that protocol Π_{cross} has completeness error $1/2^{B_f}$ $\square$

Soundness

Here we give a scetch of a soundness proof for a little weaker class of Σ-cross protocols, but we believe that it is possible to prove it for all.

Theorem (soundness)
Protocol Π_{cross} is $(2^{-B_c+1} + \epsilon_{range} + \epsilon_{DL})$-**computationally special sound** for relation $\mathcal{R}_{cross}$ where ϵ_{range} is the knowledge error of π_{range}, $\epsilon_{DL} = n_1 \cdot \epsilon_{DL_{\mathbb{G}_1}} + n_2 \cdot \epsilon_{DL_{\mathbb{G}_2}}$ is adversary advantage in solving discrete logs in both groups if for each Y_i every cross-group witness variable x_j in preimage is bound by the Pedersen vector commitment Y_i.

Proof idea: proving soundness implies building the knowledge extractor, so it's not hard to check that knowledge error of each part of Π_{cross} is 2^{-B_c} by building witness extractor separately for w_1 and w_2 in both groups, so summing them up gives knowledge error 2^{-B_c+1}. Adding knowledge error of rangeproofs: $2^{-B_c+1} + \epsilon_{range}$. Finally, we must check that cross-group witness variables x is consistent among all Pedersen commitments openings which gives us additionally error ϵ_{DL}: $(2^{-B_c+1} + \epsilon_{range} + \epsilon_{DL})$ $\square$

Zero knowledge

Theorem (sHVZK)
Protocol Π_{cross} is **special honest verifier zero knowledge** for relation $\mathcal{R}_{cross}$.

Proof idea: similar to proof of completeness if prover does not abort we build the simulator for Π_{cross} producing full transcripts from both domains $\mathbb{G}_1, \mathbb{G}_2$: $(\mathsf{R}', \mathsf{S}', c, \sigma_1, \sigma_2)$, if prover aborts with probability $1/2^{B_f}$ we build truncated simulated transcript $(\mathsf{R}', \mathsf{S}', c, \perp)$ $\square$

Using Fiat-Shamir transform we can convert Π_{cross} to non-interactive protocol preserving described properties.

Implementation

Reference implementation

We have implemented a non-interactive variant of Π_{cross} in *Rust*: `https://github.com/juja256/zkp`. This implementation is a fork of `https://github.com/zkcrypto/zkp` with a little bit modified Camenisch-Stadler DSL as *Rust* macro.

Crate supports arbitrary elliptic curve groups from *ark-ec* crate which orders doesn't exceed 256bit. For range proofs (for 64bit values) we use the fastest *bulletproofs* crate which enabled if the second group is *ark_ed25519* (we also implemented a bridge between `dalek_curve25519::RistrettoPoint` and `ark_ed25519::EdwardsAffine`)

Proving $\mathcal{R}_{comeq}$ with Π_{cross}

Recall our example relation describing the equality of discrete log and committed value in different groups:

$$\mathcal{R}_{comeq} = \{((Q, V); (x, r)) | Q = [x]G_1 \in \mathbb{G}_1, V = [x]G_2 + [s]H_2 \in \mathbb{G}_2\}$$

In our instance $\mathbb{G}_1$ is `secp256k1` group and $\mathbb{G}_2$ is `curve25519` group. To prove it with Π_{cross} we use the following strategy:

- Set parameters $B_x = 64, B_c = 128, B_f = 56$
- Draw $r_i \xleftarrow{R} F_p^*$ for $i = 0..3$ and write x, s in base 2^{B_x} representation: $x = \sum_{i=0}^{3} 2^{i \cdot B_x} x_i, s = \sum_{i=0}^{3} 2^{i \cdot B_x} s_i$.
- Commit to each x_i using Pedersen commitments in both groups: $A_i = [x_i]G_1 + [r_i]H_1 \in \mathbb{G}_1, B_i = [x_i]G_2 + [s_i]H_2 \in \mathbb{G}_2$
- Add constraints for each A_i, B_i and contrain public key as a sum: $Q = \sum_{i=0}^{3} [2^{i \cdot B_x} G_1] x_i \in \mathbb{G}_1$
- Draw appropriate homomorphisms ϕ, ψ and use Π_{cross} to prove the relation.

Proving $\mathcal{R}_{comeq}$ with Π_{cross}

So we've got the following equivalent relation:

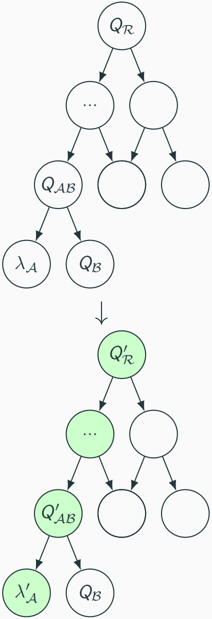
$$
\begin{aligned}
\mathcal{R}'_{comeq} = \{ & ((A_1, A_2, A_3, A_4, Q), (B_1, B_2, B_3, B_4)); \\
& ((x_0, x_1, x_2, x_3, r_0, r_1, r_2, r_3), (x_0, x_1, x_2, x_3, s_0, s_1, s_2, s_3)) | \\
& A_0 = [x_0]G_1 + [r_0]H_1, A_1 = [x_1]G_1 + [r_1]H_1, \\
& A_2 = [x_2]G_1 + [r_2]H_1, A_3 = [x_3]G_1 + [r_3]H_1, \\
& Q = [x_0]G_1 + [x_1]([2^{B_x}]G_1) + [x_2]([2^{2 \cdot B_x}]G_1) + [x_3]([2^{3 \cdot B_x}]G_1), \\
& B_0 = [x_0]G_2 + [s_0]H_2, B_1 = [x_1]G_2 + [s_1]H_2, \\
& B_2 = [x_2]G_2 + [s_2]H_2, B_3 = [x_3]G_2 + [s_3]H_2 \}
\end{aligned}
$$

Where $G_1, H_1 \in \mathbb{G}_1, G_2, H_2 \in \mathbb{G}_2$ - generators

$\mathcal{R}_{comeq}$ definition in DSL

```
define_cross_proof! {
    comeq,
    "{(Q, Com(x), G_1, G_2, H_2); (x, s) |
    Q = [x]G_1 \in G1 & Com(x) = [x]G_2 + [s]H_2 \in G2}",
    (x0, x1, x2, x3), (r0, r1, r2, r3), (s0, s1, s2, s3),
    (A0, A1, A2, A3, Q), (Com_x0, Com_x1, Com_x2, Com_x3),
    (G_1, G_1_1, G_1_2, G_1_3, H_1), (G_2, H_2) :
    A0 = (x0 * G_1 + r0 * H_1),
    Com_x0 = (x0 * G_2 + s0 * H_2),
    A1 = (x1 * G_1 + r1 * H_1),
    Com_x1 = (x1 * G_2 + s1 * H_2),
    A2 = (x2 * G_1 + r2 * H_1),
    Com_x2 = (x2 * G_2 + s2 * H_2),
    A3 = (x3 * G_1 + r3 * H_1),
    Com_x3 = (x3 * G_2 + s3 * H_2),
    Q = (x0 * G_1 + x1 * G_1_1 + x2 * G_1_2 + x3 * G_1_3)
}
```

Application: Proving Diffie-Hellman on ART



One intriguing application of cross-group Σ-protocols is proving the correctness of update of *Assymetric Ratchet Tree(ART)* - a novel approach to large group e2e encryption.

Suppose Alice $\mathcal{A}$ wants to convince other group members that she correctly updated her path in *ART*, e.g. correctly calculated all the shared DH secrets λ'. Thus she wants to prove the following relation for the every level of *ART*:

$$\mathcal{R}_{\mathcal{ART}} = \{(Q'_{\mathcal{A}}, Q_{\mathcal{B}}, Q'_{\mathcal{AB}}; \lambda'_{\mathcal{AB}}, \lambda'_{\mathcal{A}}) | \\ Q'_{\mathcal{A}} = [\lambda'_{\mathcal{A}}]P, \lambda'_{\mathcal{AB}} = \iota([\lambda'_{\mathcal{A}}]Q_{\mathcal{B}}), Q'_{\mathcal{AB}} = [\lambda'_{\mathcal{AB}}]P\}$$

Where $P \in \mathbb{G}_1, \mathrm{ord}(P) = q, \lambda'_{\mathcal{A}} \in \mathbb{Z}_q$, $\iota : \mathbb{G}_1 \to \mathbb{Z}_q$ - group to integer mapping, for example $\iota(Q) = x(Q) \mod q$ when $\mathbb{G}_1$ is elliptic curve group.

Proving $\mathcal{R}_{\mathcal{ART}}$ with bulletproofs

One may notice that proving the knowledge of preimage of ι is extremely complicated using classic Σ-protocols. Thus we have proposed to prove $\mathcal{R}_{\mathcal{ART}}$ using *bulletproofs* proving system. Here comes the power of cross-group protocols. Suppose $\mathbb{G}_2$ - `curve-25519` (or *Ristretto*) group using in the fastest *bulletproofs implementation* $\mathbb{G}_1$ - elliptic curve group defined over $\mathbb{F}_q$ where q is the prime order of $\mathbb{G}_1$.

So we introduce the following strategy:

1. Prove that committed λ-value $\mathrm{Com}(\lambda, s) = [\lambda]G_2 + [s]H_2 \in \mathbb{G}_2$ is equal to the λ in $Q = [\lambda]P \in \mathbb{G}_1$ using Π_{cross} (this is our $\mathcal{R}_{comeq}$)
2. Prove the correctness of ι computation using *bulletproofs* ($\mathcal{R}_{\iota}$) *.
3. Combining described protocols using the same transcript in non-interactive fashion we get the proving system for $\mathcal{R}_{\mathcal{ART}}$

* `https://github.com/distributed-lab/project-m/tree/main/src/zk`

Performance & future considerations

We achieved the following performance for $\mathcal{R}'_{comeq}$:

- In crate we use batchable range proofs and compressed Σ-proofs so the overall proof size is 1248 bytes.
- Prove-and-verify roundtrip takes only $7ms$ on *Ryzen 7840HS*

For $\mathcal{R}'_{\iota}$ using *bulletproofs* we achieved:

- Proof size 1121 bytes.
- Prover time around $200ms$ on *Ryzen 7840HS*

Future work:

- Formalize the soundness proof and extend the potential class of provable relations
- Extend the protocol to arbitrary number of groups
- Find new potential usages of Π_{cross} protocol, which we believe, there is a plenty of.