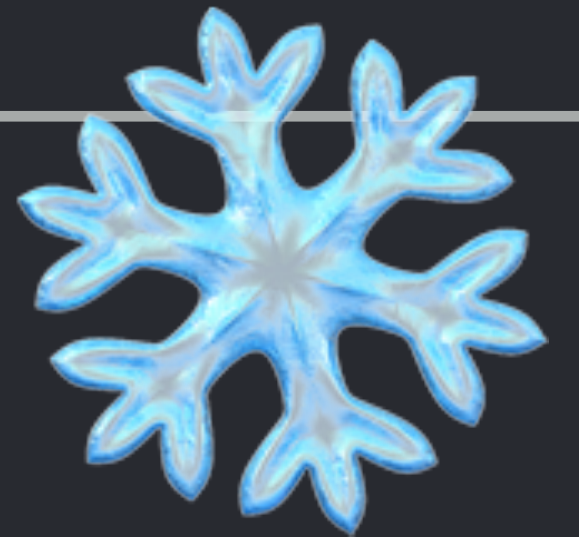




BLOCKCHAIN COMMONS

LEARNING FROST

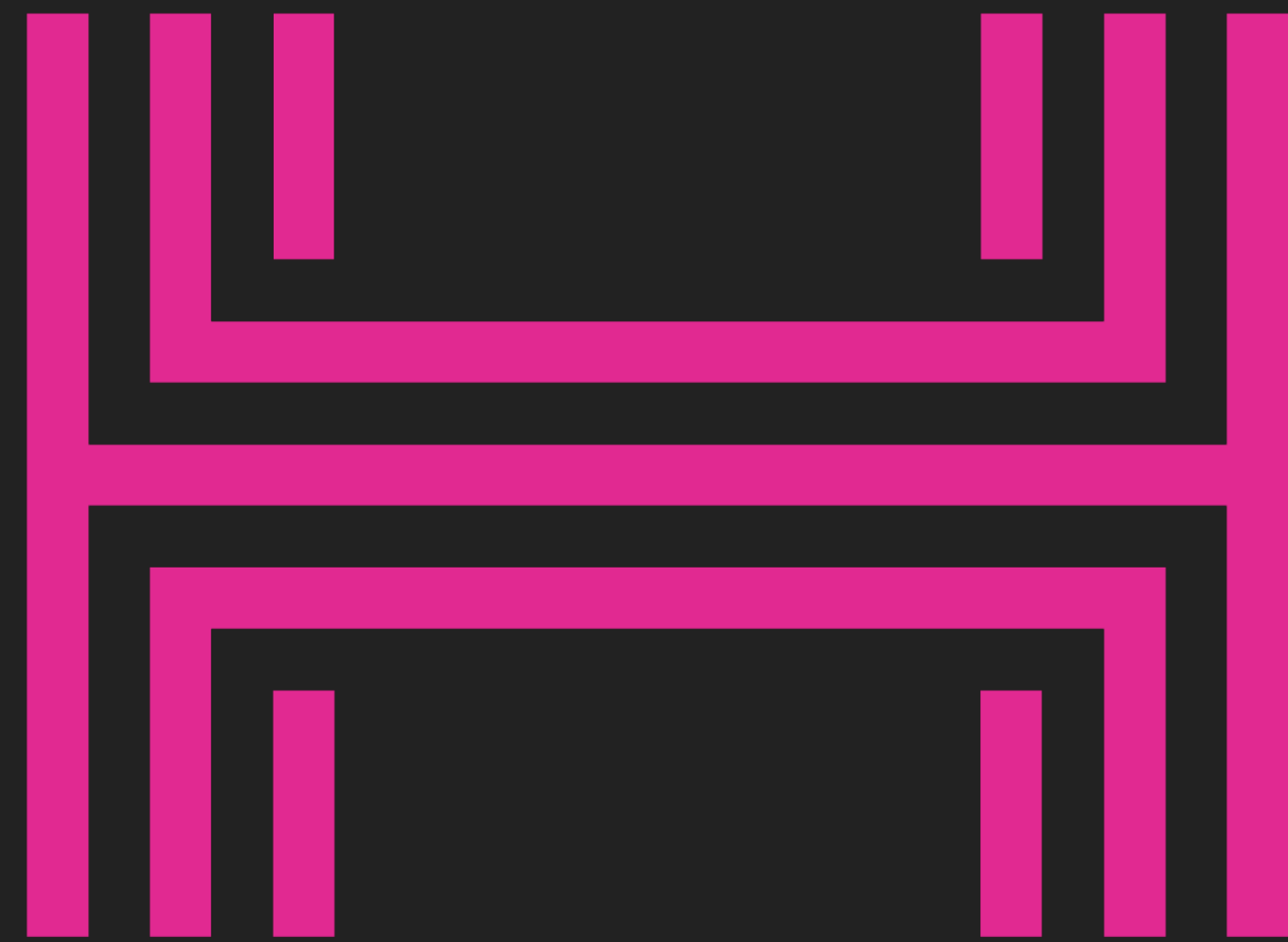


WHAT IS BLOCKCHAIN COMMONS?

- ▶ We are a community that brings together stakeholders to collaboratively build open & interoperable, secure & compassionate infrastructure.
- ▶ We design decentralized solutions where everyone wins.
- ▶ We are a neutral “not-for-profit” that enables people to control their own digital destiny.



THANKS TO OUR FROST SPONSOR



Human
Rights
Foundation





AN OVERVIEW

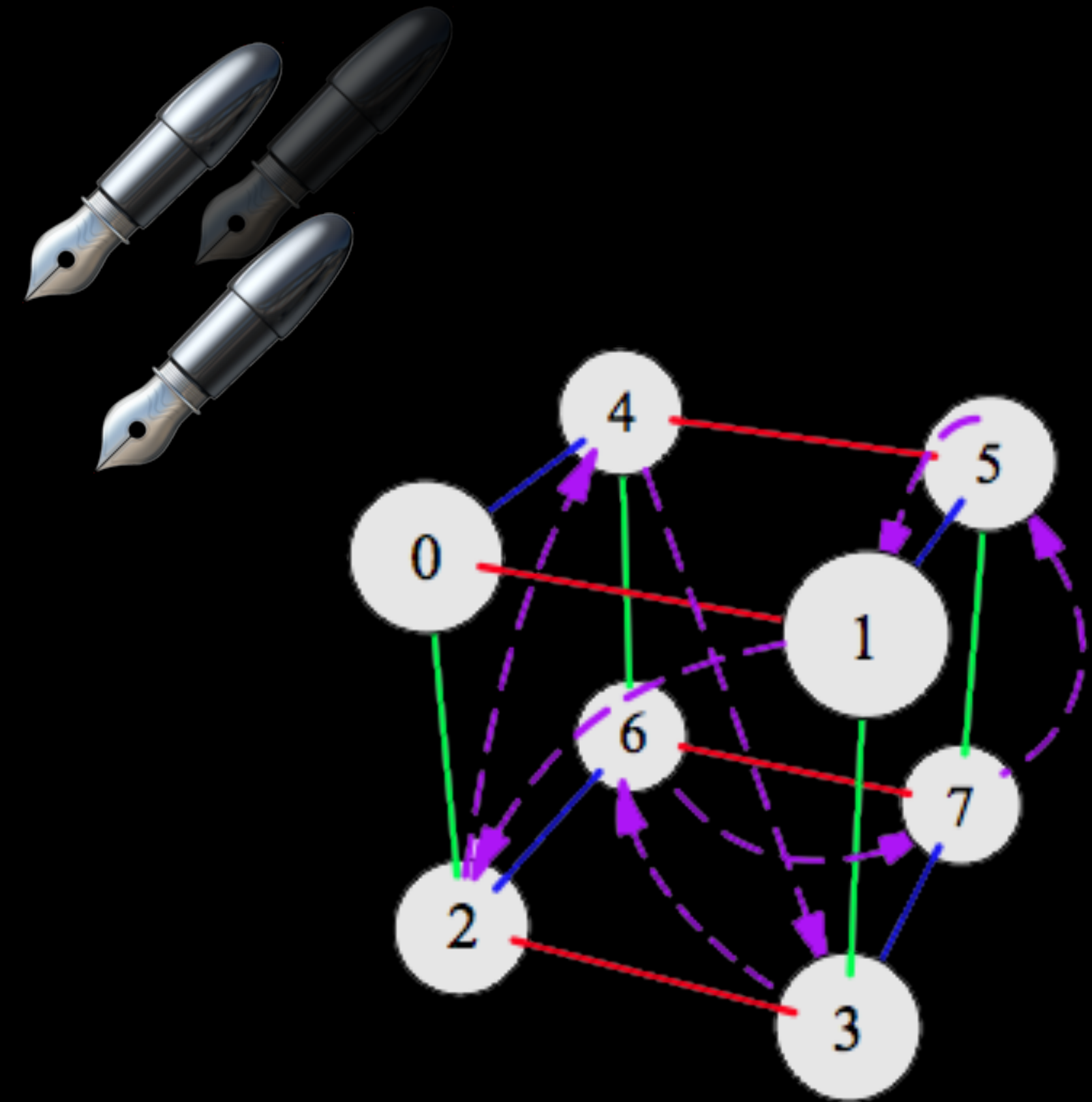
WHAT IS FROST?

WHAT IS FROST?

Flexible Round-Optimized Schnorr Threshold

It's a:

- **Threshold Signature** Scheme
- Using the **Schnorr** algorithm
- Built on Discrete Logs over Finite Fields
- With a specific methodology for signing
- That works with **Distributed Key Generation**



ABOUT SIGNATURES

Digital signatures are used to verify messages:

- **Private** key signs a message
- **Public** key verifies that signature

You know that:

- A specific person authorized the message
- The message is not changed

Signatures are not the same as encryption.



ABOUT MULTISIGS & THRESHOLD SIGNATURES

Signatures can involve:

- A group of **signers** (multisig)
- A lower **threshold** for how many signers are required for authorization (threshold sig)

These are often defined as:

- **m-of-n**: a subset (m) of the group (n) may sign
- **n-of-n**: all (n) of the group (n) must sign

FROST is an m-of-n multisig system



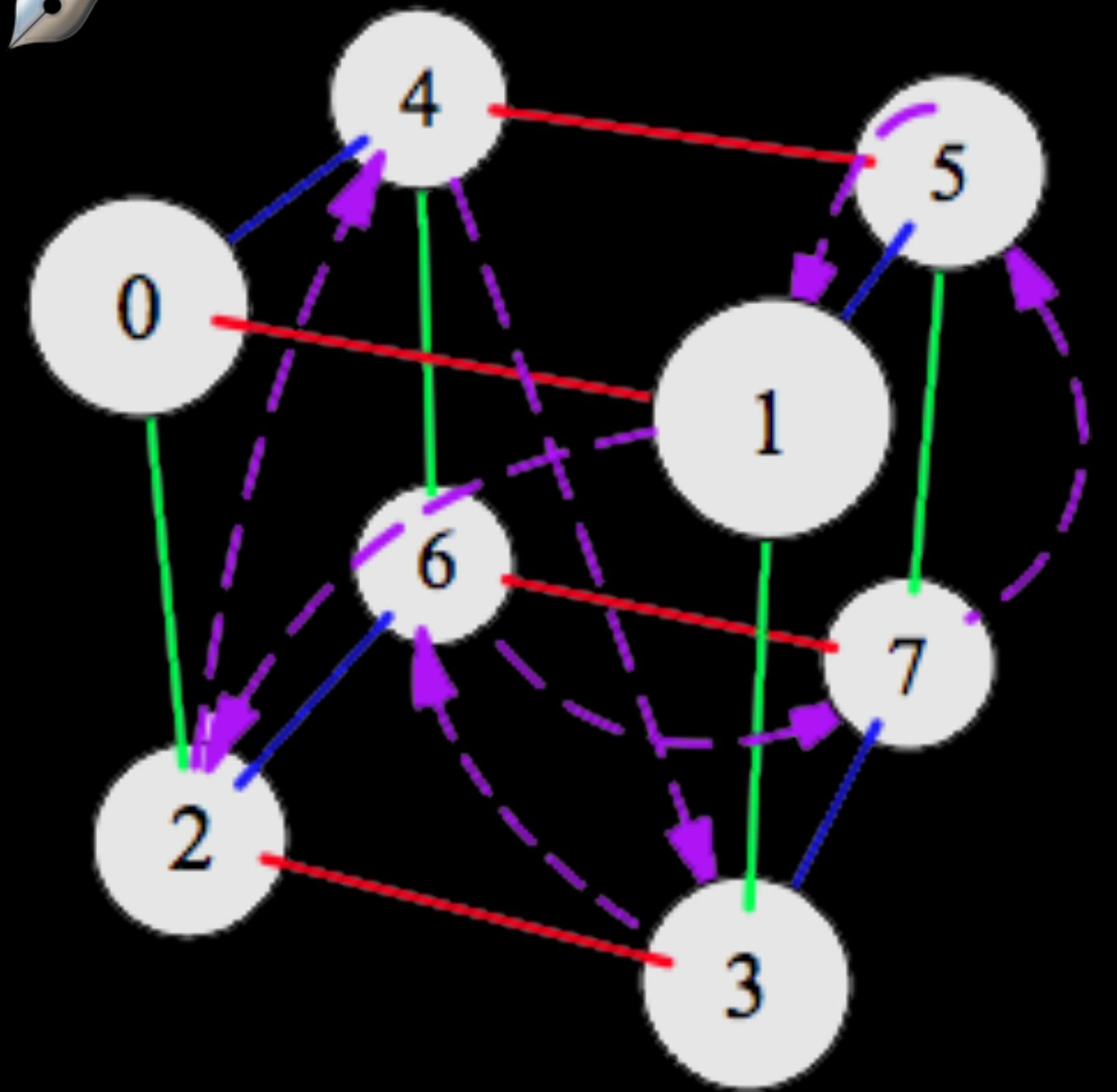
ABOUT SCHNORR SIGNATURES

Schnorr is a particular signature algorithm

- Focused on discrete logs & finite fields
- But that's not relevant to understanding it.

Its big advantage?

- Signatures are **aggregatable**
- Can be added together
- But always the **same size**



ABOUT DISTRIBUTED KEY GENERATION (DKG)

DKG is a Multiparty Computer (**MPC**) method

- Multiple computers together create a key
- The key never exists in one place!
- Each user just has a **share** (a fragment)

DKG is one method for FROST key generation

- Trusted Dealer Generation is the other
- But DKG is much more secure





ADVANTAGES OVER OTHER MULTISIG SYSTEMS

WHY USE FROST?

FROST VS BITCOIN MULTISIG

Classic **Bitcoin** Multisig

- Pay to Script Hash (P2SH)
- OP_CHECKMULTISIG
- Explicitly says it's a multisig.
- The signature can be long



FROST VS BITCOIN MULTISIG

New **FROST** Signature

- Pay to Taproot (P2TR)
- Just a signature!
 - *(and maybe a Merkle Tree hash)*
- Can't tell it's a multisig
- Can't tell how many people signed
- Can't tell who signed



FROST VS MUSIG2

MuSig2 is Another Schnorr Signature Scheme

- Only n-of-n threshold (natively)
- Accountable: you know who signed

vs **FROST**:

- m-of-n threshold
- Deniable: you don't know who signed



MULTISIGNATURE COMPARISONS

	FROST	MuSig2	Multisig
Scheme	Schnorr	Schnorr	ECDSA
Threshold	m-of-n	n-of-n	m-of-n
Privacy	Deniable	Accountable	Accountable
Signing	2 Rounds or Preprocess	2 Rounds	1 Serial Round
Size	64 bytes	64 bytes	72 bytes/sig

THE ADVANTAGES OF FROST (IN SUMMARY)

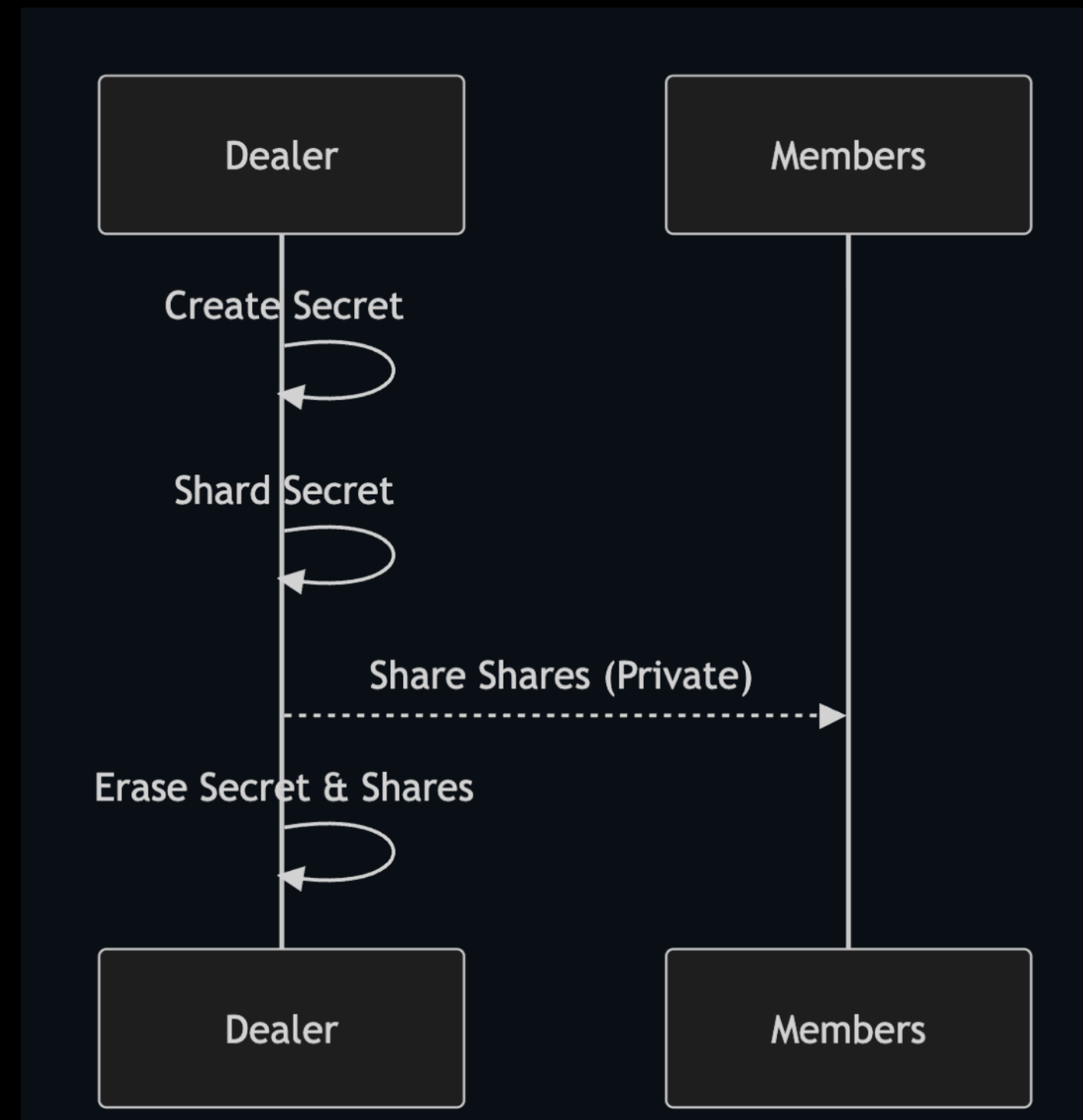
- Small Signatures
- Private Signatures
- Efficient Communication
- Strong Security (with DKG)
- Refresh & Repair Capabilities



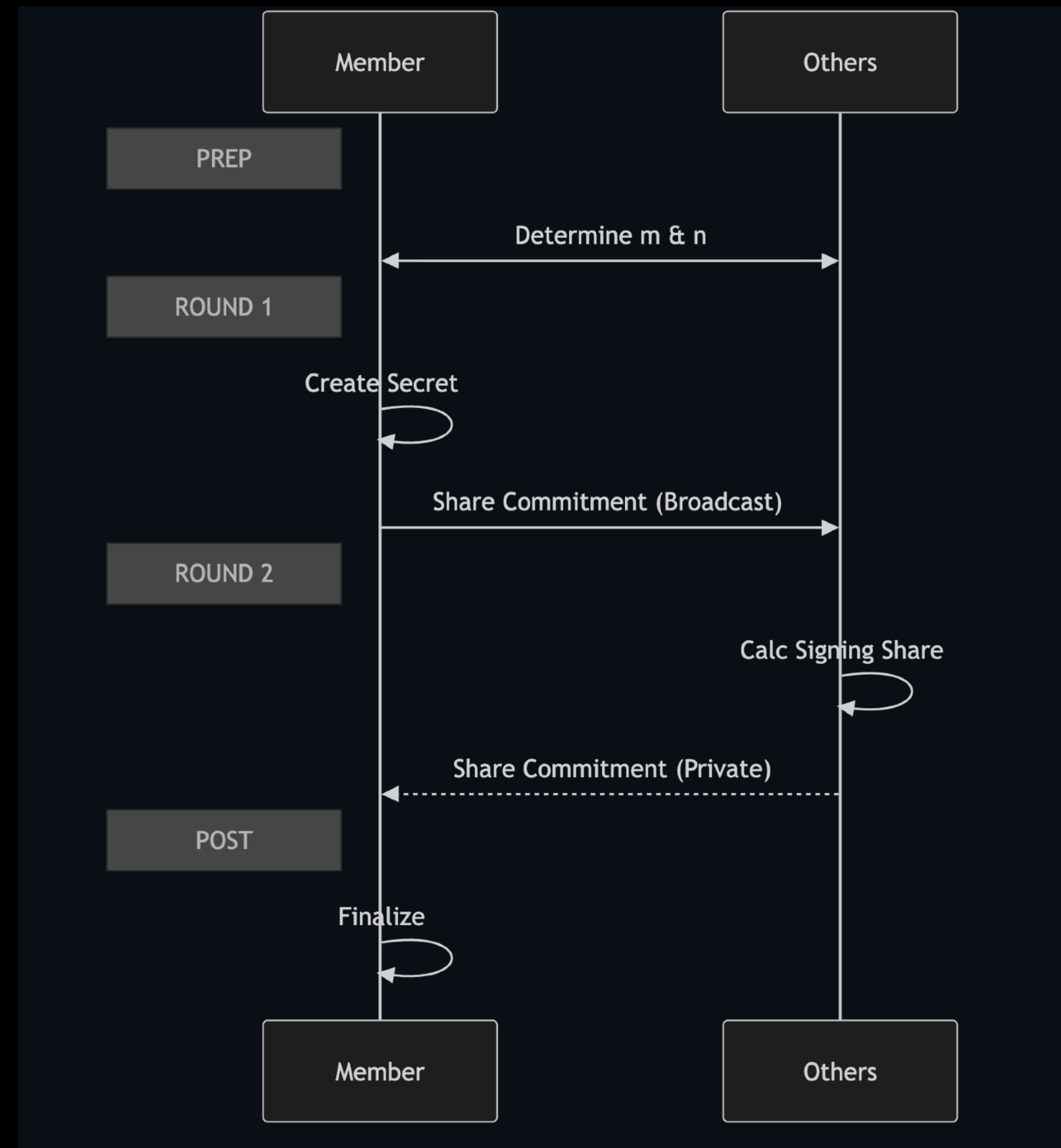
A detailed microscopic image of frost crystals, showing intricate, branching, and star-like structures. The crystals are densely packed and vary in size and orientation, creating a complex, textured appearance. The overall color is a muted, dark blue-grey, with the frost crystals appearing as lighter, almost white, structures against the darker background.

A QUICK LOOK AT FROST PROCESSES

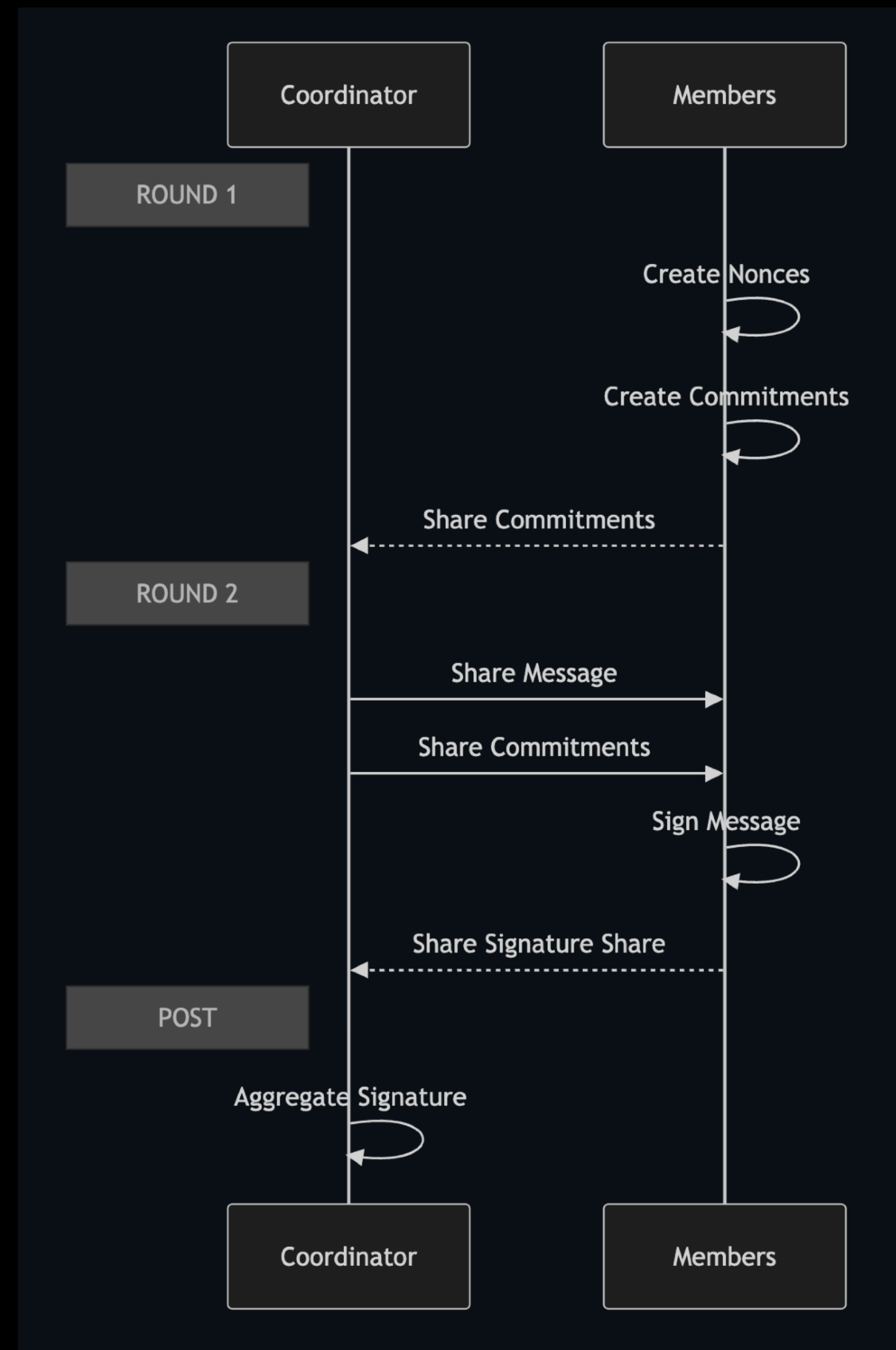
HOW DOES FROST WORK?



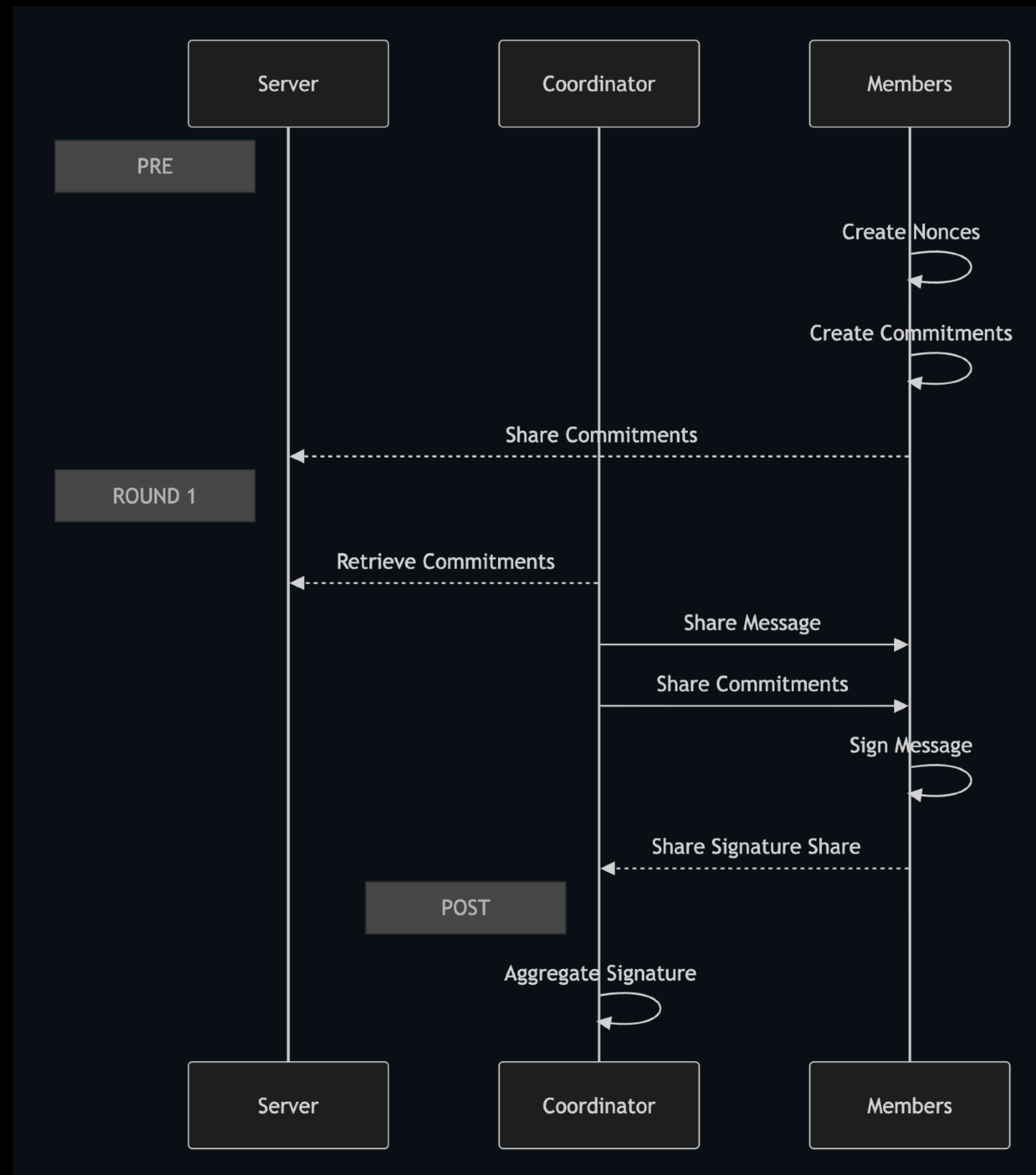
Key Generation: Trusted Dealer



Key Generation: Distributed Key Generation (DKG)



Signing: Two Rounds



Signing: One Round with Preprocessing

A detailed, high-magnification photograph of frost patterns on a dark surface. The frost forms intricate, branching, and cellular structures, resembling a complex network of fine lines and small, rounded shapes. The overall color palette is a mix of dark blues, greys, and light blues, with the frost appearing as a lighter, almost white, textured layer.

A NEW BLOCKCHAIN COMMONS COURSE

LEARNING FROST FROM THE COMMAND LINE

ABOUT THE COURSE

We have released the *Learning FROST from the Command Line* course.

- Written in the style of ...
 - *Learning Bitcoin from the Command Line*
- Sponsored by HRF
- The start of this presentation was Chapter 1

<https://learningfrost.blockchaincommons.com/>



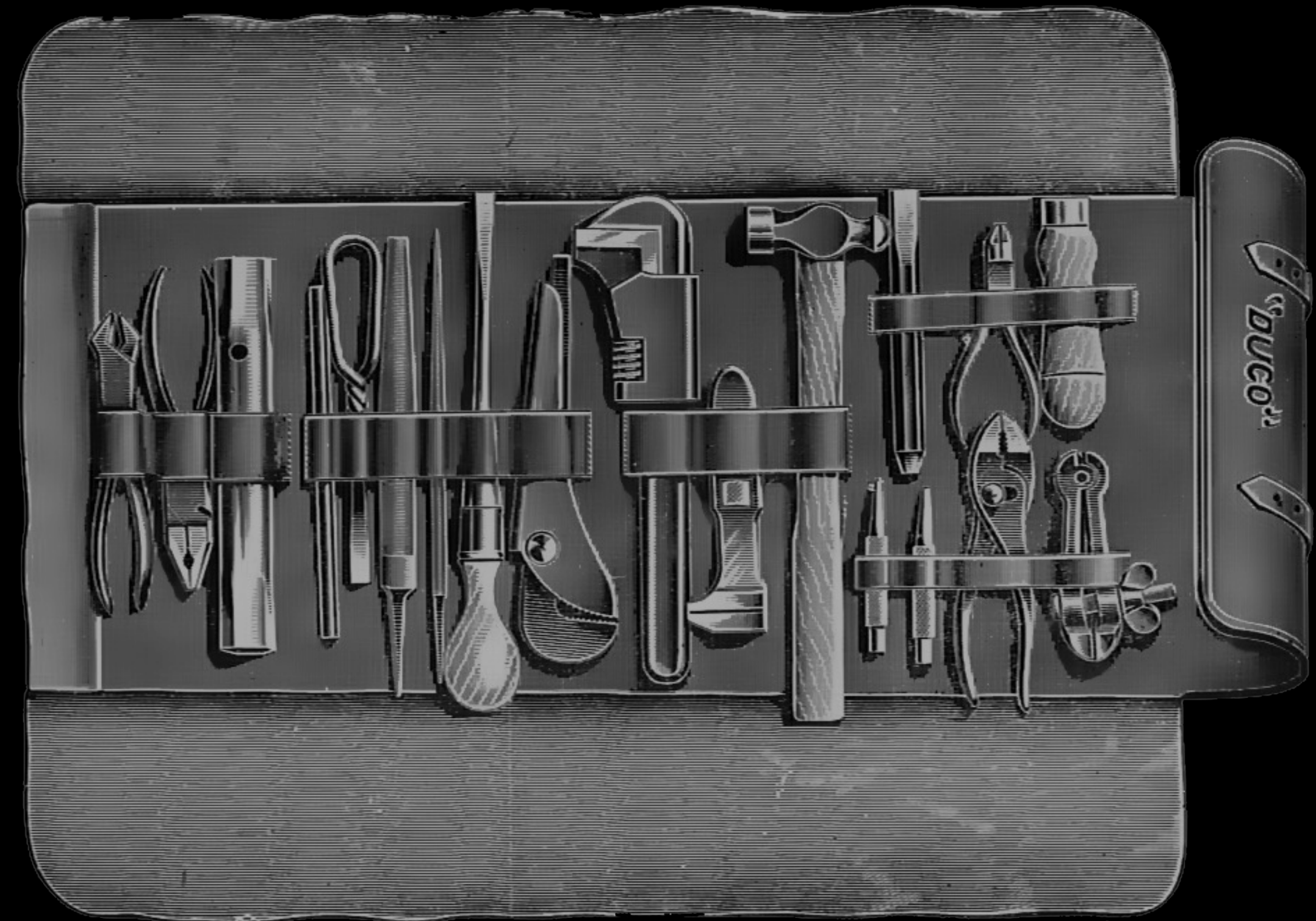
ABOUT THE FROST TOOLS

Course built on ZF FROST Tools:

- trusted-dealer
- dkg
- coordinator
- participant
- frostd server
- frost-client

Installable with **Cargo**.

<https://github.com/ZcashFoundation/frost-tools>



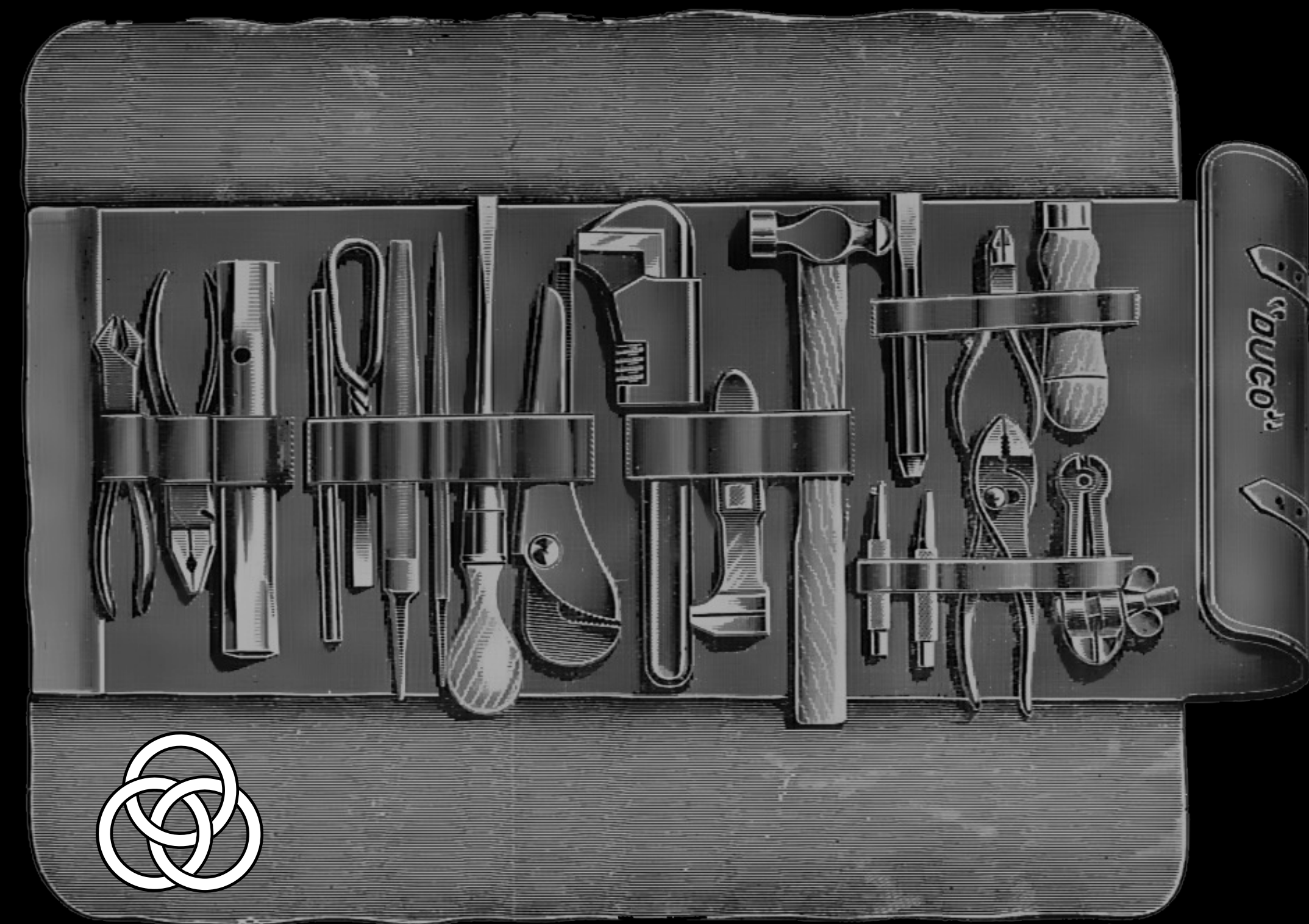
OUR EXPANSION TO THE FROST TOOLS

frost-verify-rust

- A tool for checking ZF FROST Signatures

Tools for **Signing Bitcoin transactions**

- Secp256K1-TR Ciphersuite for ZF FROST
- Taproot Tweak for ZF FROST
- Helper to Extract Sighash from PSBT
- Helper to Insert Signature into PSBT



A QUICK OVERVIEW

The course offers the hands-on experience!

- That's how you learn how stuff works
- But here's a quick look at what it contains

Also take a look at Wolf's two Bitcoin videos

- FROST-CLI Demo Meeting
- FROST DKG Bitcoin Signing Demo




```
ShannonA — bash — 80x25
~ — bash
[% more key-package-1.json | jq
{
  "header": {
    "version": 0,
    "ciphersuite": "FROST-ED25519-SHA512-v1"
  },
  "identifier": "0100000000000000000000000000000000000000000000000000000000000000",
  "signing_share": "f0a7b5aa58916c50a137a95cd36d500872f474c1388cf0226abc6db2aa50c30a",
  "commitment": [
    "40d8cd4dbd25e71e172a69b1dcc75151a46db5e9f3fe3ee92bbfa871c71b8351",
    "6a6ef7346dc3668f076ea6ec4f329a96a1e0acaf4dc8a5b5379e6d0bc9dec2e8"
  ]
}
%
```

CREATING FROST SIGNING SHARES: TRUSTED DEALER


```
ShannonA — dkg — 80x25
~ — dkg

[% dkg
The minimum number of signers: (2 or more)
2
The maximum number of signers:
3
Your identifier (this should be an integer between 1 and 65535):
1

=== ROUND 1: SEND PACKAGES ===

Round 1 Package to send to all other participants (your identifier: "0100000000
000000000000000000000000000000000000000000000000000000000000000000000000"):

{"header":{"version":0,"ciphersuite":"FROST-ED25519-SHA512-v1"},"commitment":["1
64a6cb3365dcaeab9e740eb73c8f0910d2a24dfe1c1f678f78e81c2206c51d6","131b5ec816bbb2
a3e260685e8ed9e4240da14eb6bf0db755d944295c8b194961"],"proof_of_knowledge":"e6fc8
1643215921bc4894b5a992c1002a857f6c9b47327855667178c552ba4f7163506183a20b1154c054
8b465d9fe0a31d71d5b5bb67b7cc0897c949f4c7804"}

=== ROUND 1: RECEIVE PACKAGES ===

Input Round 1 Packages from the other 2 participants.

The sender's identifier (hex string):
```

CREATING FROST SIGNING SHARES: DKG


```
ShannonA — dkg —
[% dkg
The minimum number of
2
The maximum number of
3
Your identifier (this
1

=== ROUND 1: SEND PACK

Round 1 Package to send
00000000000000000000000000000000

{"header":{"version":0
64a6cb3365dcaeab9e740e
a3e260685e8ed9e4240da1
1643215921bc4894b5a992
8b465d9fe0a31d71d5b5bb

=== ROUND 1: RECEIVE P

Input Round 1 Packages

The sender's identifie
```

12:25 ★ 📶 🔋

2 / 5 Commitments Exit

corresponding fields.

My name

Alice

My commitment

At7qYW9psdpSwdgr/YwLtVE4sVQzJ1Ylup
qEtaG2GxgDA1Ntp550Rey8rva6Hae+vOqi
UUuAGCxS2GiLfZUk/+50Apdjn20vZsGoA
mTh2fsV0iE8qCeDJBb421qBcoSTMN2bylt
3kmK3TThl8w9nqUM+LsNwKoAy41qZwx
WQbxYAI8AC4EOqQllqmSPtLDhqau9NCyY
vnXgq7XZ31C/yRt4/Vmg=

📄

 View QR code

bob

Enter bob's commitment

📄

📄

carol

Enter carol's commitment

📄

📄

☐ I have verified that everyone has my commitment

Generate shares

```
+
}
and 65535):

ur identifier: "010000000000
"):

HA512-v1"},"commitment":["1
c2206c51d6","131b5ec816bbb2
"proof_of_knowledge":"e6fc8
2ba4f7163506183a20b1154c054
```

CREATING FROST SIGNING SHARES: STACK WALLET


```
frost-dkg — -zsh — 80x25
...-key localhost+3-key.pem  ~/tmp/frost-dkg — -zsh  ~/tmp/frost-dkg — -zsh  ~/tmp/frost-dkg — -zsh  +

ShannonA@Shannons-MacBook-Pro frost-dkg % frost-client dkg -d "DKG: Alice, Bob, Eve" \
-s 127.0.0.1:2744 \
-S $BOB_PUBKEY,$EVE_PUBKEY \
-t 2 \
[-c alice.toml]
Logging in...
Creating DKG session...
Getting session info...
Waiting for other participants to send their Round 1 Packages.....
.....
Waiting for other participants to send their broadcasted Round 1 Packages.....
Waiting for other participants to send their Round 2 Packages....
Group created; information written to alice.toml
ShannonA@Shannons-MacBook-Pro frost-dkg %
```

CREATING FROST SIGNING SHARES: DKG WITH SERVER

FROST FACES TYPICAL KEY MANAGEMENT ISSUES

You now have a file with your signing share!

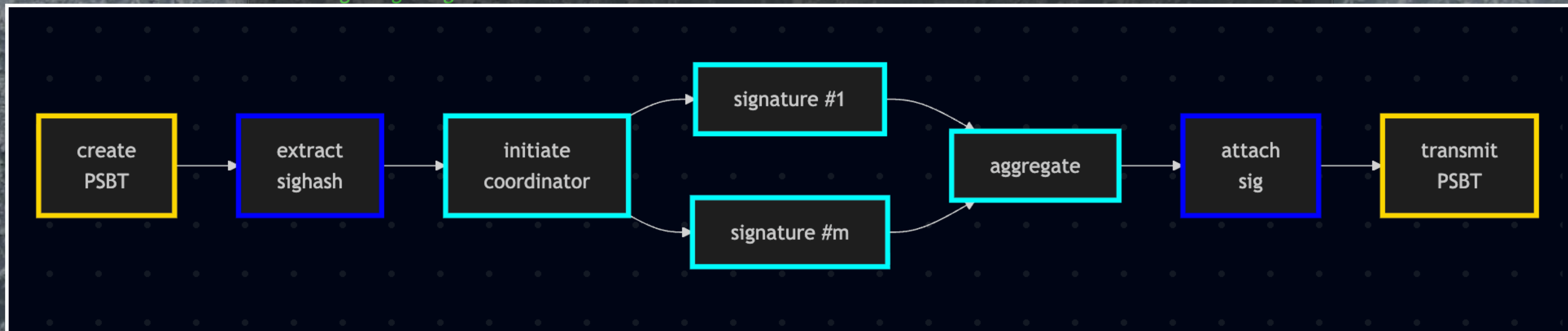
- How do you secure your signing share?
- How do you record its intended usage?
- How do you keep it from being lost?

But **FROST** has big advantages due to thresholds:

- if $m > 1$, one stolen key doesn't steal signature
- if $m < n$, one lost key doesn't lose signature




```
frost-dkg — -zsh — 80x25
...-key localhost+3-key.pem  ...g-250917.sig -c alice.toml  ...7.0.1:2744 -c alice.toml  ... ~/tmp/frost-dkg — -zsh +
ShannonA@Shannons-MacBook-Pro frost-dkg % GROUP_ID=$(grep -oE '^\[group\.[0-9a-f
]+' bob.toml | head -1 | sed 's/^\[group\./\[')
[frost-client participant -g $GROUP_ID -s 127.0.0.1:2744 -c bob.toml
Logging in...
Joining signing session...
```



```
Sending signature share to coordinator...
Done
[ShannonA@Shannons-MacBook-Pro frost-dkg %
[ShannonA@Shannons-MacBook-Pro frost-dkg %
[ShannonA@Shannons-MacBook-Pro frost-dkg %
[ShannonA@Shannons-MacBook-Pro frost-dkg %
ShannonA@Shannons-MacBook-Pro frost-dkg %
```

SIGNING BITCOIN WITH FROST

THAT'S FROST IN A NUTSHELL

This is a capstone of two years' work.

- We've held five **FROST** meetings
- For implementers & developers

We're now moving from discussion to implementation:

- Learning **FROST** is just the start
- We are also bringing **FROST** into our stack

Our ultimate goal?

- Wider usage of **FROST**!

<https://developer.blockchaincommons.com/frost/>





WHAT'S NEXT?

FUTURE WORK

SIGNING EXAMPLES

So what do you sign with FROST?

- Gordian Envelope
- Gordian Club Updates



COORDINATION EXAMPLES

ZF FROST offers two types of coordination

- By Hand (ugh!)
- By Coordinator (centralized!)

How do we get advantages of coordination without centralization?

- Hubert, the Dead-Drop Hub

<https://developer.blockchaincommons.com/hubert/>





ANY QUESTIONS?



Learning FROST from the Command Line

<https://learningfrost.blockchaincommons.com>

Shannon Appelcline

shannon.appelcline@gmail.com



@BlockchainComns

“Advocating for the creation of open, interoperable, secure & compassionate digital infrastructure to enable people to control their own digital destiny and to maintain their human dignity online”

