No More Guesswork: Ready-to-Use Distributed Key Generation

Jonas Nick, Tim Ruffing

Threshold Signatures

Threshold Signatures

t-of-n

Threshold Signatures

Correctness

t honest signers
can produce
a valid signature.

t-of-n

Threshold Signatures

Correctness

t honest signers
can produce
a valid signature.

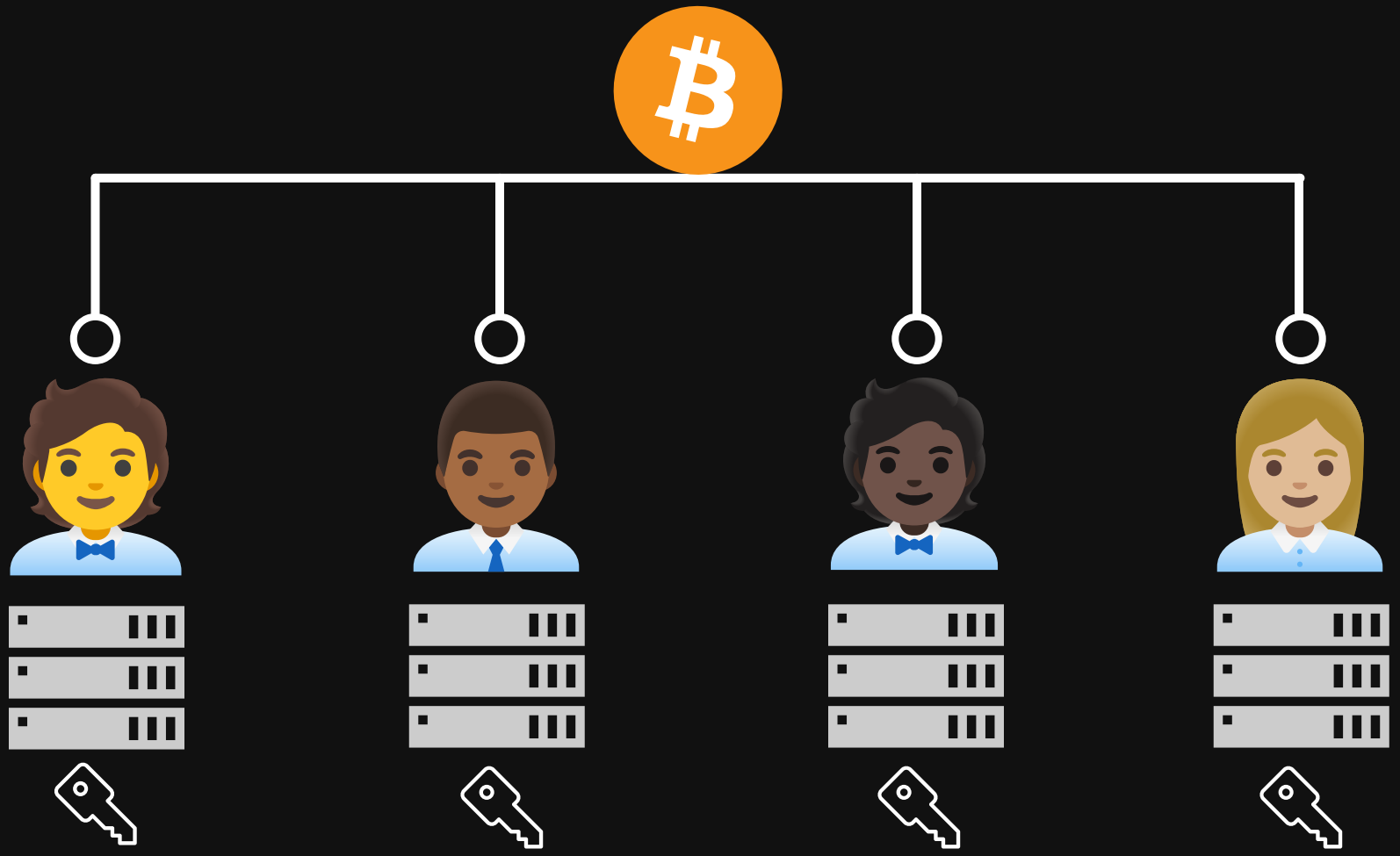
t-of-n

Unforgeability

$t - 1$ malicious signers
cannot produce
a valid signature.

Example Application: 2-of-3 Personal Wallet

Example Application: 3-of-4 Shared Wallet

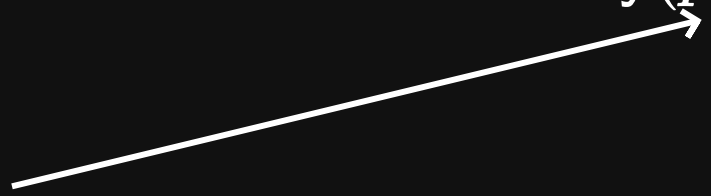


Schnorr-compatible Threshold Signatures

SchnorrVerify(pk, σ, m)

Schnorr-compatible Threshold Signatures

SchnorrVerify(pk, σ, m)



- ordinary Schnorr public key

Schnorr-compatible Threshold Signatures

- ordinary Schnorr public key
- obtained via interactive Distributed Key Generation (DKG) protocol

Schnorr-compatible Threshold Signatures

- ordinary Schnorr public key
- obtained via interactive Distributed Key Generation (DKG) protocol

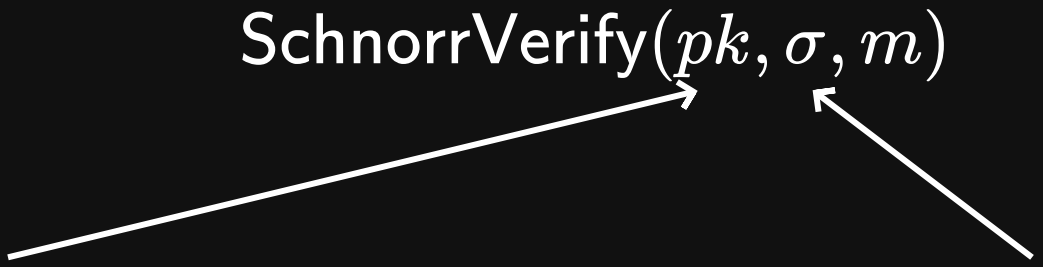
- ordinary Schnorr signature

Schnorr-compatible Threshold Signatures

- ordinary Schnorr public key
- obtained via interactive Distributed Key Generation (DKG) protocol

- ordinary Schnorr signature
- obtained via interactive signing protocol

Schnorr-compatible Threshold Signatures



- ordinary Schnorr public key
- obtained via interactive Distributed Key Generation (DKG) protocol

- ordinary Schnorr signature
- obtained via interactive signing protocol

Schnorr-compatible threshold signatures are indistinguishable from ordinary signatures on the Bitcoin blockchain.

State-of-the-art Scheme: FROST

State-of-the-art Scheme: FROST

- FROST: Flexible Round-Optimized Schnorr Threshold Signatures (Komlo and Goldberg 2020)

State-of-the-art Scheme: FROST

- FROST: Flexible Round-Optimized Schnorr Threshold Signatures (Komlo and Goldberg 2020)
- Signing is two rounds of communication

State-of-the-art Scheme: FROST

- FROST: Flexible Round-Optimized Schnorr Threshold Signatures (Komlo and Goldberg 2020)
- Signing is two rounds of communication
- FROST does not require the majority of signers to be honest.

State-of-the-art Scheme: FROST

- FROST: Flexible Round-Optimized Schnorr Threshold Signatures (Komlo and Goldberg 2020)
- Signing is two rounds of communication
- FROST does not require the majority of signers to be honest.
  - Honest majority would exclude 3-of-4 setup.

State-of-the-art Scheme: FROST

- FROST: Flexible Round-Optimized Schnorr Threshold Signatures (Komlo and Goldberg 2020)
- Signing is two rounds of communication
- FROST does not require the majority of signers to be honest.
  - Honest majority would exclude 3-of-4 setup.
    - Assumes up to $t - 1 = 2$ dishonest signers for unforgeability.

State-of-the-art Scheme: FROST

- FROST: Flexible Round-Optimized Schnorr Threshold Signatures (Komlo and Goldberg 2020)
- Signing is two rounds of communication
- FROST does not require the majority of signers to be honest.
  - Honest majority would exclude 3-of-4 setup.
    - Assumes up to $t - 1 = 2$ dishonest signers for unforgeability.
    - So, 2 signers assumed to be honest, which is not the majority.

There are (almost) no real-world deployments. Why?

Real-World Cryptography Stack

Layer
Deployments
Implementations
Standards and Specifications
Papers and Proofs

Real-World Cryptography Stack

Layer	**FROST signing**
Deployments	
Implementations	
Standards and Specifications	
Papers and Proofs	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)

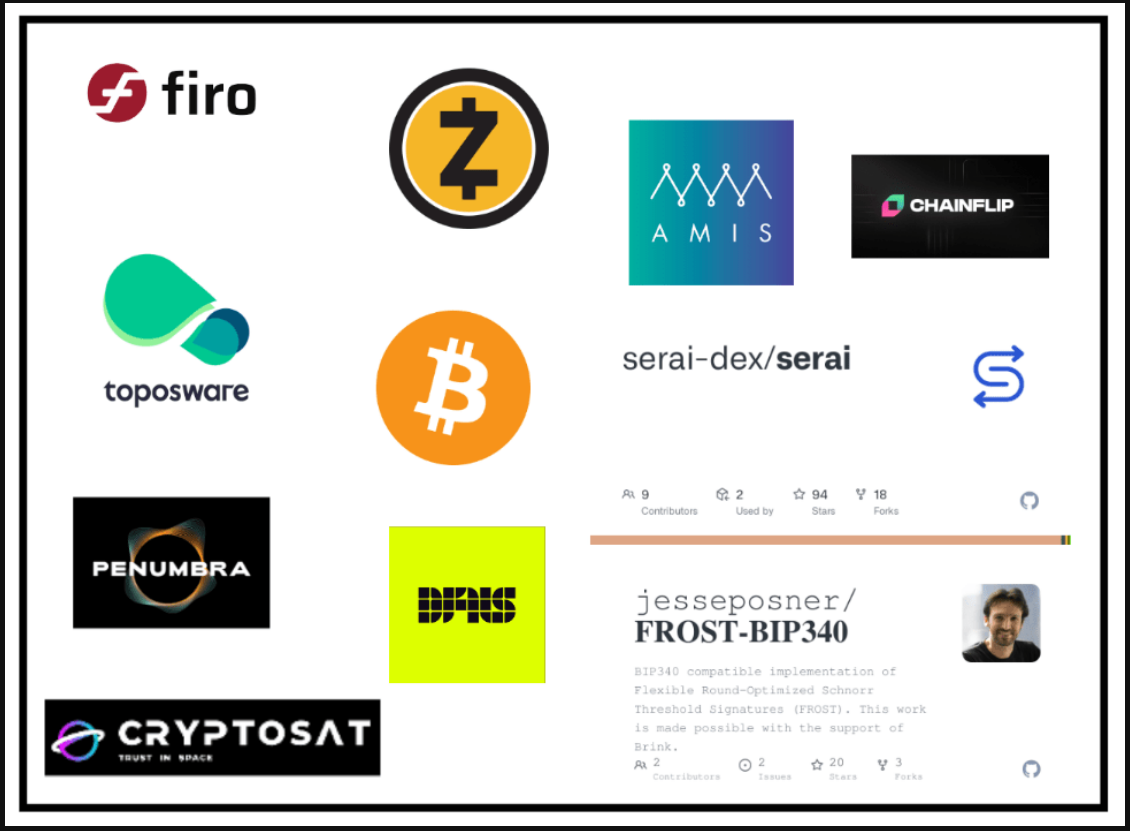
RFC 9591 (IRTF/CFRG)

Internet Research Task Force (IRTF)
Request for Comments: 9591
Category: Informational
Published: June 2024
ISSN: 2070-1721

D. Connolly
Zcash Foundation
C. Komlo
University of Waterloo,
Zcash Foundation
I. Goldberg
University of Waterloo
C. A. Wood
Cloudflare

The Flexible Round-Optimized Schnorr Threshold (FROST) Protocol for Two-Round Schnorr Signatures

Implementations



*From: Lessons Learned from Multi-Party Schnorr Signatures at RWC'23 (Crites, Komlo, **Ruffing**)*

Real-World Cryptography Stack

Layer	**FROST signing**
Deployments	
Implementations	Many
Standards and Specifications	RFC 9591 (IRTF/CFRG)
Papers and Proofs	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)

Real-World Cryptography Stack

Layer	**FROST signing**
Deployments	*Very few*
Implementations	Many
Standards and Specifications	RFC 9591 (IRTF/CFRG)
Papers and Proofs	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)

Let's increase adoption by implementing FROST in our Bitcoin crypto library.

Let's increase adoption by implementing FROST in our Bitcoin crypto library.

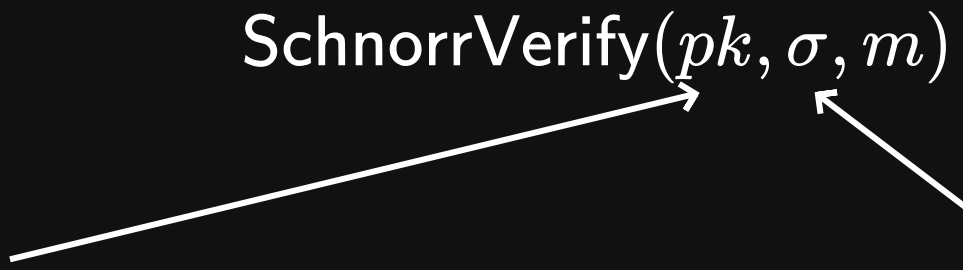
secp256k1-zkp Public

A fork of libsecp256k1 with support for advanced and experimental features such as Confidential Assets and MuSig2

C ☆ 383 215

github.com/BlockstreamResearch/

Schnorr-compatible Threshold Signatures



- ordinary Schnorr public key
- obtained via interactive Distributed Key Generation (DKG) protocol

- ordinary Schnorr signature
- obtained via interactive signing protocol

RFC 9591 (IRTF/CFRG)

RFC 9591 (IRTF/CFRG)

Key generation for FROST signing is out of scope for this document.

Real-World Cryptography Stack

Layer	**FROST signing**
Deployments	*Very few*
Implementations	Many
Standards and Specifications	RFC 9591 (IRTF/CFRG)
Papers and Proofs	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)

CG**R**S (CRYPTO'23)

Real-World Cryptography Stack

Layer	**FROST signing**	**FROST DKG**
Deployments	*Very few*	
Implementations	Many	
Standards and Specifications	RFC 9591 (IRTF/CFRG)	-
Papers and Proofs	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)

Just implement DKG!

DKG for FROST

DKG for FROST

- For compatibility with FROST,
  DKG should not require honest majority.

DKG for FROST

- For compatibility with FROST,
  DKG should not require honest majority.
- DKG should work in the asynchronous setting.

DKG for FROST

- For compatibility with FROST,
  DKG should not require honest majority.
- DKG should work in the asynchronous setting.
- **PedPop** [KG20]: Pedersen DKG [Ped91] with
  Proofs of Possession

DKG for FROST

- For compatibility with FROST,
  DKG should not require honest majority.
- DKG should work in the asynchronous setting.
- **PedPop** [KG20]: Pedersen DKG [Ped91] with Proofs of Possession
  - 2 rounds

DKG for FROST

- For compatibility with FROST,
  DKG should not require honest majority.
- DKG should work in the asynchronous setting.
- **PedPop** [KG20]: Pedersen DKG [Ped91] with Proofs of Possession
  - 2 rounds
  - Not a general-purpose DKG, but proven secure in combination with FROST

rk-based DKG scenario, the equality check can be instant
n signer transmit their local value of the common parame
reof) using a reliable broadcast protocol, e.g., echo broa
ly, the recipients can compare their local value with the
heck for equality among all participants

rk-based DKG scenario, the equality check can be instant
n signer transmit their local value of the common parame
reof) using a reliable broadcast protocol, e.g., echo broa
ly, the recipients can compare their local value with the
heck for equality among all participants

What does that mean exactly?

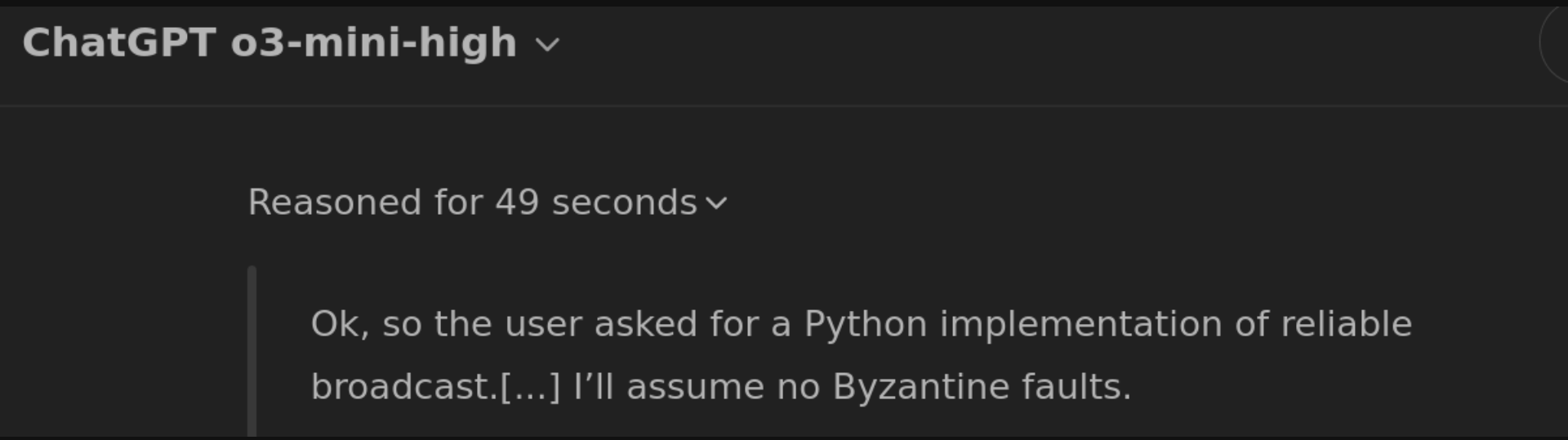
- I know that reliable broadcast is more than regular network broadcast.

- I know that reliable broadcast is more than regular network broadcast.
- But there's no obvious off-the-shelf implementation.

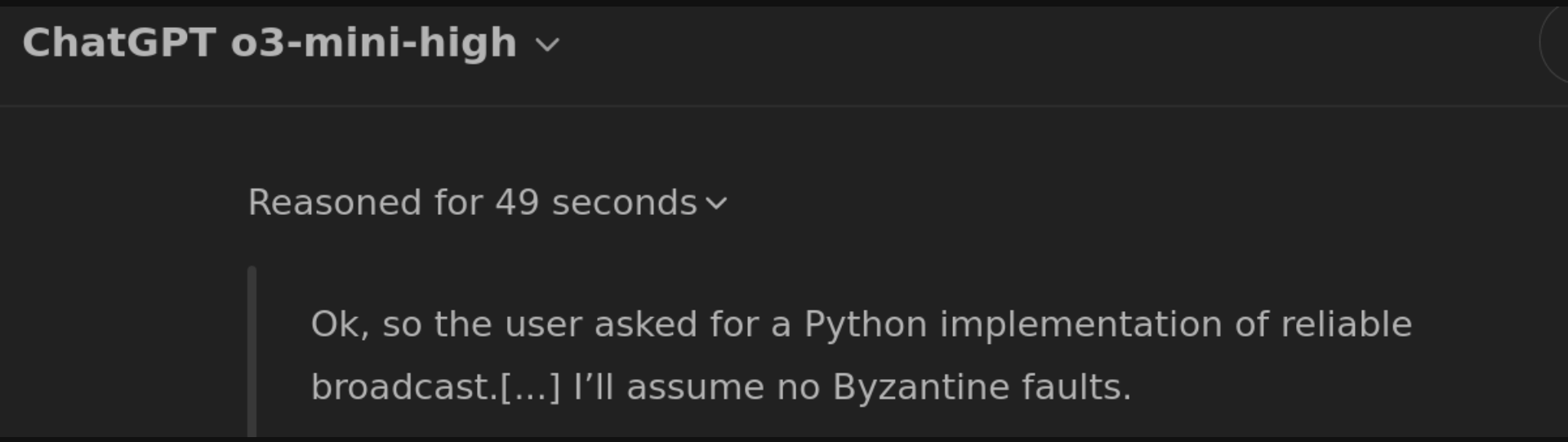
- I know that reliable broadcast is more than regular network broadcast.
- But there's no obvious off-the-shelf implementation.
- So just implement it.

Approach 1: "Vibe Coding"

Approach 1: "Vibe Coding"



Approach 1: "Vibe Coding"



ChatGPT models interpret "reliable" simply as
guaranteed message delivery,
overlooking malicious senders entirely.

Approach 2: RTFM

Approach 2: RTFM

Documentation of a FROST signing implementation instructing users to provide a suitable broadcast channel themselves:

Approach 2: RTFM

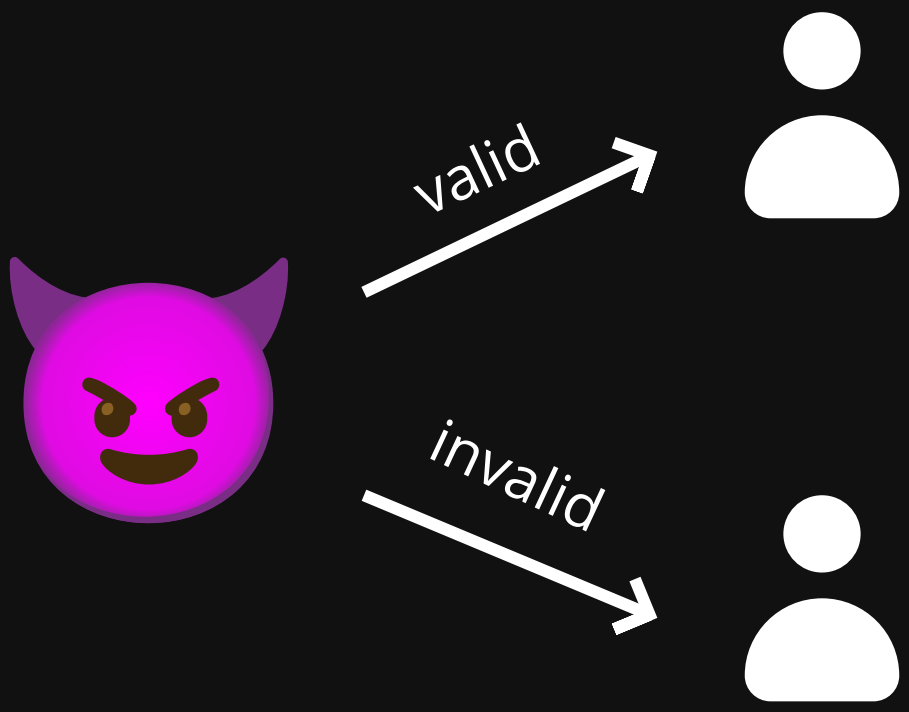
Documentation of a FROST signing implementation instructing users to provide a suitable broadcast channel themselves:

A secure broadcast channel in the context of multi-party computation protocols such as FROST has the following properties:

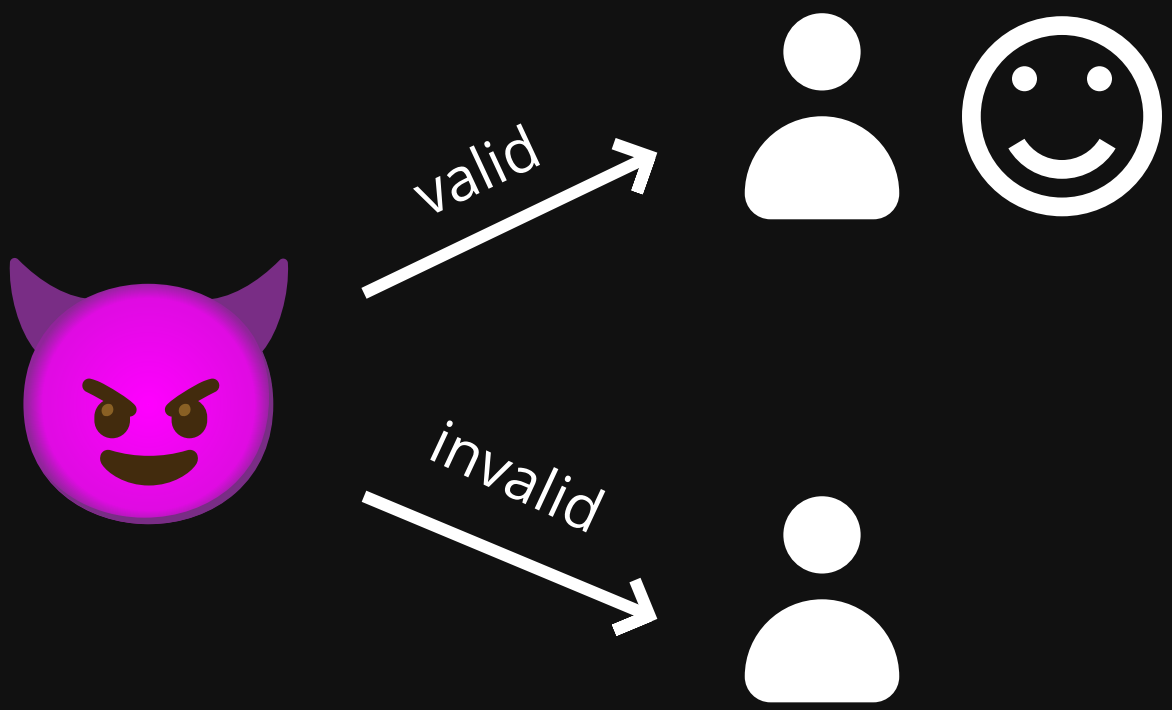
1. Consistent. Each participant has the same view of the message sent over the channel.
2. Authenticated. Players know that the message was in fact sent by the claimed sender. In practice, this requirement is often fulfilled by a PKI.
3. Reliable Delivery. Player i knows that the message it sent was in fact received by the intended participants.
4. Unordered. The channel does not guarantee ordering of messages.

Resulting DKG Is Still Broken

2-of-3 Example

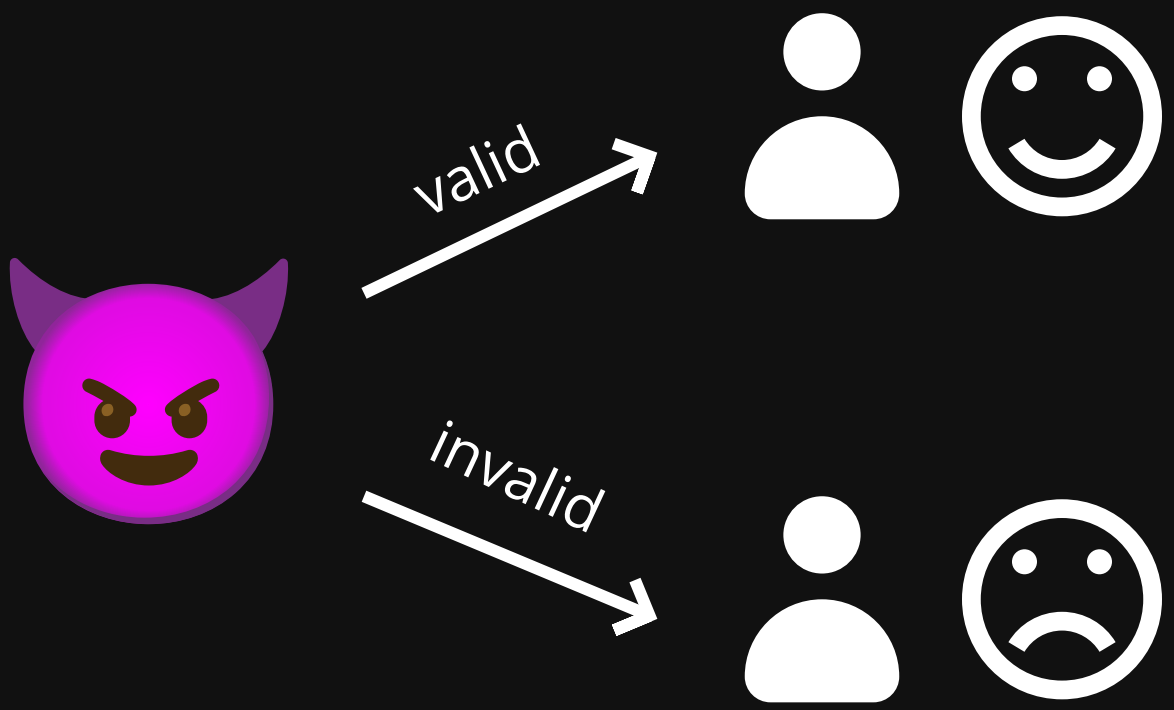


2-of-3 Example



DKG succeeded, let's send money to resulting pk!

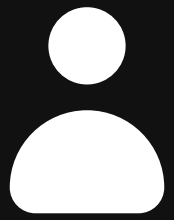
2-of-3 Example

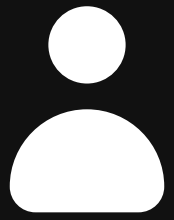


DKG succeeded, let's send money to resulting pk!

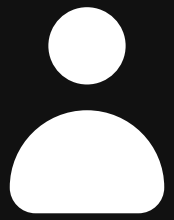
DKG not succeeded

Just 1 usable signer left, but we need 2!
Money is lost forever.





Just 1 usable signer left, but we need 2!
Money is lost forever.



Recap

Recap

- Goal: implement FROST with Distributed Key Generation

Recap

- Goal: implement FROST with Distributed Key Generation
- Even if you carefully

Recap

- Goal: implement FROST with Distributed Key Generation
- Even if you carefully
  - study the FROST papers,

Recap

- Goal: implement FROST with Distributed Key Generation
- Even if you carefully
  - study the FROST papers,
  - obtain the right definitions for the flavor of broadcast needed,

Recap

- Goal: implement FROST with Distributed Key Generation
- Even if you carefully
  - study the FROST papers,
  - obtain the right definitions for the flavor of broadcast needed,
  - interpret them accurately,

Recap

- Goal: implement FROST with Distributed Key Generation
- Even if you carefully
  - study the FROST papers,
  - obtain the right definitions for the flavor of broadcast needed,
  - interpret them accurately,
  - correctly implement broadcast, secure channels, and the DKG protocol itself,

Recap

- Goal: implement FROST with Distributed Key Generation
- Even if you carefully
  - study the FROST papers,
  - obtain the right definitions for the flavor of broadcast needed,
  - interpret them accurately,
  - correctly implement broadcast, secure channels, and the DKG protocol itself,

...the result is still broken.

Our Solution: ChillDKG

Real-World Cryptography Stack

Layer	**FROST signing**	**FROST DKG**
Deployments	Very few	
Implementations	Many	
Standards and Specifications	RFC 9591 (IRTF/CFRG)	ChillDKG
Papers and Proofs	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)

Real-World Cryptography Stack

Layer	**FROST signing**	**FROST DKG**
Deployments	Very few	
Implementations	Many	
Standards and Specifications	RFC 9591 (IRTF/CFRG)	ChillDKG
Papers and Proofs	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)	KG (SAC'20), BCKMTZ (CRYPTO'22), CG**R**S (CRYPTO'23)

Based on SimplPedPop

Based on SimplPedPop

SimplPedPop [CG**R**S23] replaces broadcast abstraction with interactive equality check protocol:

$$\mathsf{Eq}(input) \to \{\mathsf{succeed}, \mathsf{fail}\}$$

Based on SimplPedPop

SimplPedPop [CG**R**S23] replaces broadcast abstraction with interactive equality check protocol:

$$\mathsf{Eq}(input) \to \{\textsf{succeed}, \textsf{fail}\}$$

- **Integrity:** If some honest signer succeeds, then the $input$ values of all honest signers are equal.

Based on SimplPedPop

SimplPedPop [CG**R**S23] replaces broadcast abstraction with interactive equality check protocol:

$$\mathsf{Eq}(input) \to \{\textsf{succeed}, \textsf{fail}\}$$

- **Integrity:** If some honest signer succeeds, then the $input$ values of all honest signers are equal.
- **Agreement:** If some honest signer succeeds, then eventually all honest signers will succeed.

ChillDKG's Eq Protocol

ChillDKG's E_q Protocol

A signed variant of echo-broadcast
(Goldwasser-Lindell 2005):

ChillDKG's Eq Protocol

A signed variant of echo-broadcast (Goldwasser-Lindell 2005):

1. Sign $input$, send signature to everyone.

ChillDKG's Eq Protocol

A signed variant of echo-broadcast (Goldwasser-Lindell 2005):

1. Sign $input$, send signature to everyone.
2. Upon receiving valid signatures from all n signers, succeed.

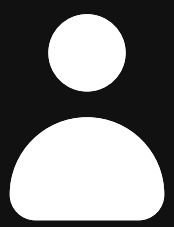
ChillDKG's Eq Protocol

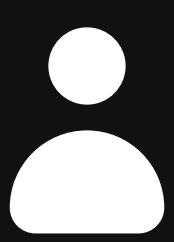
A signed variant of echo-broadcast (Goldwasser-Lindell 2005):

1. Sign $input$, send signature to everyone.
2. Upon receiving valid signatures from all n signers, succeed.

Integrity:

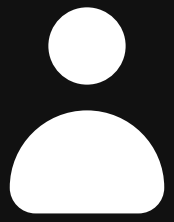
Certificates Ensure Agreement



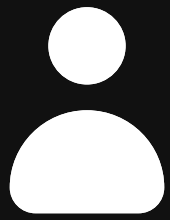


Certificates Ensure Agreement

succeed!

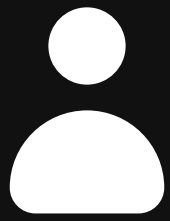


Certificates Ensure Agreement



succeed!

certificate
= list of signatures



Certificates Ensure Agreement

Agreement in the DKG

Agreement in the DKG

In SimplPedPop:

Agreement in $\mathbf{Eq} \Rightarrow$ Agreement in the DKG

Agreement in the DKG

In SimplPedPop:

Agreement in $\mathbf{Eq} \Rightarrow$ Agreement in the DKG

So, if one participant succeeds in the DKG, then they can convince every honest participant to succeed.

Agreement in the DKG

In SimplPedPop:

Agreement in $\mathbf{Eq}$ $\Rightarrow$ Agreement in the DKG

So, if one participant succeeds in the DKG, then they can convince every honest participant to succeed.

Agreement in the DKG

In SimplPedPop:

Agreement in $\mathbf{Eq}$ $\Rightarrow$ Agreement in the DKG

So, if one participant succeeds in the DKG, then they can convince every honest participant to succeed.

(does not require honest majority)

ChillDKG Bonus

ChillDKG Bonus

- Extend **Eq input** with (essential parts of) transcript of DKG protocol.

ChillDKG Bonus

- Extend **Eq input** with (essential parts of) transcript of DKG protocol.
- Difficulty: Keep **input** size $O(n)$
  - homomorphic encryption to the rescue

ChillDKG Bonus

- Extend **Eq input** with (essential parts of) transcript of DKG protocol.
- Difficulty: Keep **input** size $O(n)$
  - homomorphic encryption to the rescue
- Then, **input** plus certificate allows any signer to recover their DKG outputs from single per-device seed (e.g., when device is broken).

ChillDKG Bonus

- Extend **Eq input** with (essential parts of) transcript of DKG protocol.
- Difficulty: Keep **input** size $O(n)$
  - homomorphic encryption to the rescue
- Then, **input** plus certificate allows any signer to recover their DKG outputs from single per-device seed (e.g., when device is broken).
  - Per DKG backup is just public data (**input** and certificate) instead of secret data.

Takeaways

Takeaways

- DKG is integral to FROST, yet securely implementing it was notoriously challenging.

Takeaways

- DKG is integral to FROST, yet securely implementing it was notoriously challenging.
  - Unclear requirements cause subtle vulnerabilities.

Takeaways

- DKG is integral to FROST, yet securely implementing it was notoriously challenging.
  - Unclear requirements cause subtle vulnerabilities.
- ChillDKG provides a ready-to-implement solution addressing these issues.

Takeaways

- DKG is integral to FROST, yet securely implementing it was notoriously challenging.
  - Unclear requirements cause subtle vulnerabilities.
- ChillDKG provides a ready-to-implement solution addressing these issues.
  - Offers more practical features we didn't have time to cover.

Takeaways

- DKG is integral to FROST, yet securely implementing it was notoriously challenging.
  - Unclear requirements cause subtle vulnerabilities.
- ChillDKG provides a ready-to-implement solution addressing these issues.
  - Offers more practical features we didn't have time to cover.
  - github.com/BlockstreamResearch/bip-frost-dkg

Takeaways

- DKG is integral to FROST, yet securely implementing it was notoriously challenging.
  - Unclear requirements cause subtle vulnerabilities.
- ChillDKG provides a ready-to-implement solution addressing these issues.
  - Offers more practical features we didn't have time to cover.
  - github.com/BlockstreamResearch/bip-frost-dkg

Feedback welcome!

me@real-or-random.org
jonasd.nick@gmail.com
x.com/blksresearch

Blockstream RESEARCH

You've unlocked the backup slides

But it doesn't terminate if someone refuses to sign!

But it doesn't terminate if someone refuses to sign!

True! We were hoping for this question.

But:

But it doesn't terminate if someone refuses to sign!

True! We were hoping for this question.

But:

1. If there's an attacker, you might **want** to abort DKG!

But it doesn't terminate if someone refuses to sign!

True! We were hoping for this question.

But:

1. If there's an attacker, you might **want** to abort DKG!
2. We want to support dishonest majority, where we can't guarantee termination anyway!

But it doesn't terminate if someone refuses to sign!

True! We were hoping for this question.

But:

1. If there's an attacker, you might **want** to abort DKG!
2. We want to support dishonest majority, where we can't guarantee termination anyway!
3. ChillDKG supports blaming misbehaving signers, which requires help from the DKG coordinator.