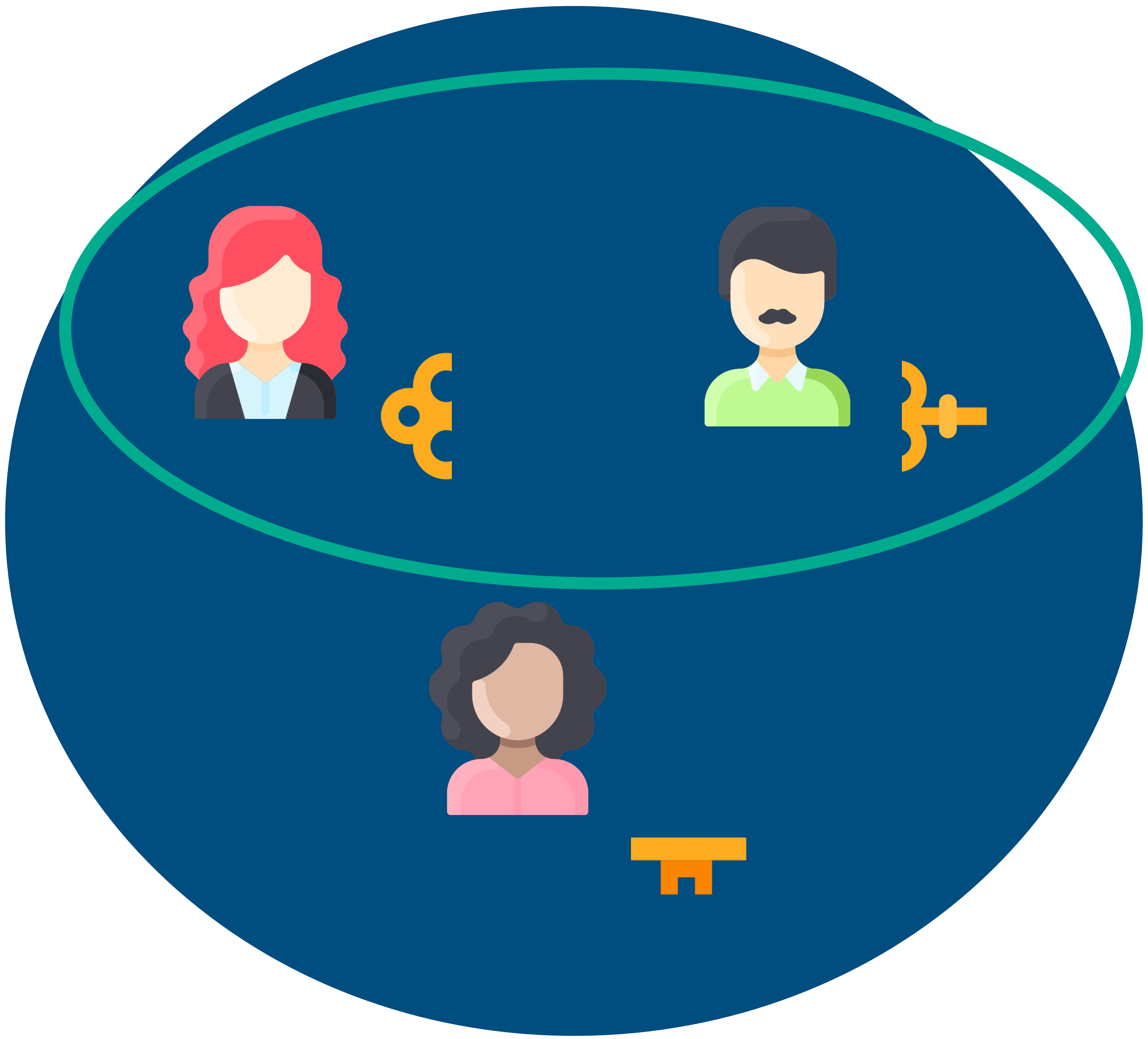
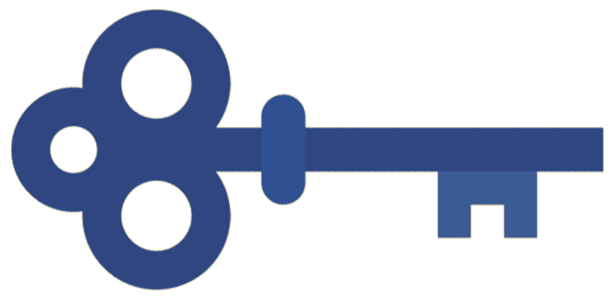
A Plausible Attack on the Adaptive Security of Threshold Schnorr Signatures

Elizabeth Crites & Alistair Stewart
Web3 Foundation

Threshold Signatures





Public Key PK

- $t+1$-out-of-n
- trusted key generation or DKG to produce PK

$(2,3)$ Example

NIST Threshold Standardization

NIST | COMPUTER SECURITY RESOURCE CENTER CSRC

NIST IR 8214C (2nd Public Draft)

NIST First Call for Multi-Party Threshold Schemes

Date Published: March 27, 2025

Comments Due: May 30, 2025 (public comment period is CLOSED)

Email Questions to: nistir-8214C-comments@nist.gov

(Single-Party) Schnorr Signatures

Signer:

$sk \leftarrow \mathbb{Z}_p, PK \leftarrow g^{sk}$

Icons by flaticon.com

(Single-Party) Schnorr Signatures

Signer:

$sk \leftarrow \mathbb{Z}_p,\ PK \leftarrow g^{sk}$

To sign a message m:

$r \leftarrow \mathbb{Z}_p,\ R \leftarrow g^r$

$c \leftarrow H(PK, m, R)$

$z \leftarrow r + c \cdot sk$

(Single-Party) Schnorr Signatures

$$sig = (R, z)$$

Signer:

$$sk \leftarrow \mathbb{Z}_p,\ PK \leftarrow g^{sk}$$

To sign a message m:

$$r \leftarrow \mathbb{Z}_p,\ R \leftarrow g^r$$

$$c \leftarrow H(PK, m, R)$$

$$z \leftarrow r + c \cdot sk$$

(Single-Party) Schnorr Signatures

Signer:

$sk \leftarrow \mathbb{Z}_p,\ PK \leftarrow g^{sk}$

To sign a message m:

$r \leftarrow \mathbb{Z}_p,\ R \leftarrow g^r$

$c \leftarrow H(PK, m, R)$

$z \leftarrow r + c \cdot sk$

$sig = (R, z)$ →

Verifier:

$c \leftarrow H(PK, m, R)$

$R \cdot PK^c = g^z$ ✓

Icons by flaticon.com

(Single-Party) Schnorr Signatures

$$sig = (R, z)$$

Signer:

$$sk \leftarrow \mathbb{Z}_p,\ PK \leftarrow g^{sk}$$

To sign a message m:

$$r \leftarrow \mathbb{Z}_p,\ R \leftarrow g^r$$

$$c \leftarrow H(PK, m, R)$$

$$z \leftarrow r + c \cdot sk$$

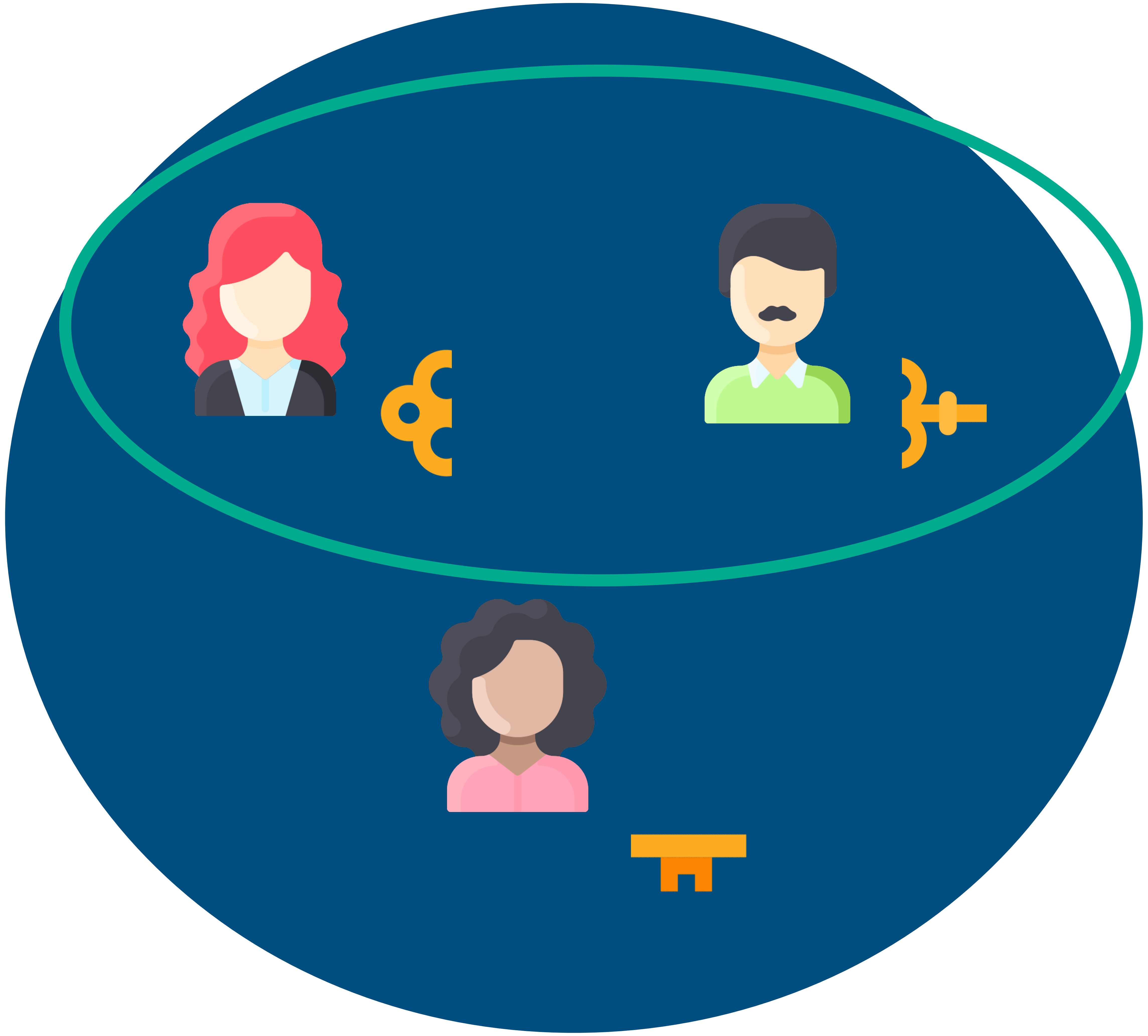
Verifier:

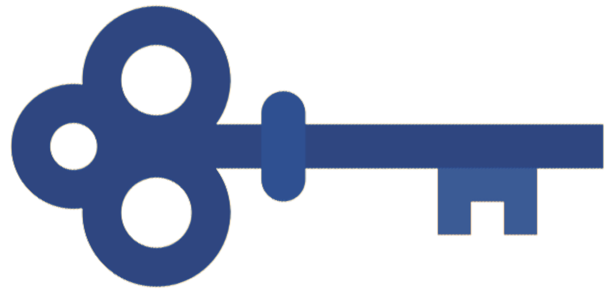
$$c \leftarrow H(PK, m, R)$$

$$R \cdot PK^c = g^z \ ✓$$

Unforgeable in the ROM under DL

Threshold Schnorr Signatures





Public Key PK

- final signature $sig = (R, z)$ verifies as Schnorr signature under PK

$(2,3)$ Example

Remember ROS attacks?

ROS Attacks

- ROS problem first stated in Schnorr's original paper
- many threshold, blind, and multi-signatures were shown insecure
- ROS attacks fundamentally rely on concurrency
- most recent showing a polynomial-time attack for greater than $0.725\log(p)$ (e.g., ≈180) concurrent sessions
- a birthday problem

Our Attack

- similar to ROS, we construct an attack where the forgery amounts to a linear combination of parties' public values
- uniquely, our attack allows a forgery *based on public key shares alone* - no partial signatures are required
- unlike ROS attacks, the attack works *even for a single signing session*

The Problem P

- we define a search problem P and show a concrete, efficient attack if P is easy to solve

The Problem P

- we define a search problem P and show a concrete, efficient attack if P is easy to solve

Definition 2. P *is the following search problem. Given* $\boldsymbol{w} \in \mathbb{Z}_p^{t+1}$ *and* $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \mathbb{Z}_p^{t+1}$, *find a set* $CS \subset \{1, \ldots, n\}$ *with* $|CS| = t_c$ *such that* $\boldsymbol{w} \in span(\{\boldsymbol{v}_i\}_{i \in CS})$ *if one exists.*

The Problem P

- we define a search problem P and show a concrete, efficient attack if P is easy to solve

Definition 2. P *is the following search problem. Given* $\boldsymbol{w} \in \mathbb{Z}_p^{t+1}$ *and* $\boldsymbol{v}_1, ..., \boldsymbol{v}_n \in \mathbb{Z}_p^{t+1}$, *find a set* $CS \subset \{1, \ldots, n\}$ *with* $|CS| = t_c$ *such that* $\boldsymbol{w} \in span(\{\boldsymbol{v}_i\}_{i \in CS})$ *if one exists.*

- similar to ROS, P does not rely on group elements or operations (field elements only)
- unlike ROS, P is not stated in terms of a random oracle

Adaptive Security

- our attack affects adaptive security only
- def: adversary cannot forge a signature, even if it can corrupt signers during signing
- NIST Call emphasizes a strong preference for schemes achieving provable adaptive security:

Adaptive Security

- our attack affects adaptive security only
- def: adversary cannot forge a signature, even if it can corrupt signers during signing
- NIST Call emphasizes a strong preference for schemes achieving provable adaptive security:

> "Given the possibility of adaptive corruptions in the real world, it is important to consider for any proposed threshold signature scheme whether the major safety properties of interest (such as unforgeability) are safeguarded against such an adversary."

Adaptive Security

- our attack affects adaptive security only
- def: adversary cannot forge a signature, even if it can corrupt signers during signing
- NIST Call emphasizes a strong preference for schemes achieving provable adaptive security:

> "Given the possibility of adaptive corruptions in the real world, it is important to consider for any proposed threshold signature scheme whether the major safety properties of interest (such as unforgeability) are safeguarded against such an adversary."

- *full* adaptive security is the analogue of static security ($t_c = t$ corruptions)

Conditions of Our Attack

Conditions of Our Attack

Our attack applies to any scheme with the following 3 properties:

Conditions of Our Attack

Our attack applies to any scheme with the following 3 properties:

1. Public key shares $PK_1, \ldots, PK_n$ are public

Conditions of Our Attack

Our attack applies to any scheme with the following 3 properties:

1. Public key shares $PK_1, \ldots, PK_n$ are public
2. Public keys are $PK = g^{f(0)}, PK_1 = g^{f(1)}, \ldots, PK_n = g^{f(n)}$, where f is a degree t polynomial with coefficients in $\mathbb{Z}_p$

Conditions of Our Attack

Our attack applies to any scheme with the following 3 properties:

1. Public key shares $PK_1, \ldots, PK_n$ are public
2. Public keys are $PK = g^{f(0)}, PK_1 = g^{f(1)}, \ldots, PK_n = g^{f(n)}$, where f is a degree t polynomial with coefficients in $\mathbb{Z}_p$

- e.g., Shamir secret sharing, DL-based DKGs like Pedersen, Gennaro et al.

Conditions of Our Attack

Our attack applies to any scheme with the following 3 properties:

1. Public key shares $PK_1, \ldots, PK_n$ are public
2. Public keys are $PK = g^{f(0)}, PK_1 = g^{f(1)}, \ldots, PK_n = g^{f(n)}$, where f is a degree t polynomial with coefficients in $\mathbb{Z}_p$
    - e.g., Shamir secret sharing, DL-based DKGs like Pedersen, Gennaro et al.
3. Final signature is compatible with Schnorr verification: $R \cdot PK^c = g^z$

Affected Schemes

- FROST, FROST2, FROST3
- SimpleTSig
- Sparkle, Sparkle+
- Lindell'22
- Classic S.
- GKMN'21 (deterministic)
- Arctic (deterministic)

<u>Robust (G.O.D.):</u>

- ROAST
- SPRINT
- HARTS
- GJKR'07
- Stinson-Strobl'01

Non-Affected Schemes

- Crackle & Snap
- FROST-Mask
- Abe-Fehr’04
- Zero S.
- Glacius
- Gargos
- non-threshold Schnorr schemes

Non-Affected Schemes

- Crackle & Snap
- FROST-Mask
- Abe-Fehr'04
- Zero S.
- Glacius
- Gargos
- non-threshold Schnorr schemes

"On the Adaptive Security of Key-Unique Threshold Signatures"

eprint 2025/943

Our Attack

The Attack

The Attack

- adversary sets $R^* = PK^{\alpha_0} PK_1^{\alpha_1} \cdots PK_n^{\alpha_n}$ for random $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{Z}_p$

The Attack

- adversary sets $R^* = PK^{\alpha_0} PK_1^{\alpha_1} \cdots PK_n^{\alpha_n}$ for random $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{Z}_p$
- valid forgery: $R^* PK^{c^*} = g^{z^*}$

The Attack

- adversary sets $R^* = PK^{\alpha_0} PK_1^{\alpha_1} \cdots PK_n^{\alpha_n}$ for random $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{Z}_p$
- valid forgery: $R^* PK^{c^*} = g^{z^*}$
- $R^* PK^{c^*} = PK^{c^* + \alpha_0} PK_1^{\alpha_1} \cdots PK_n^{\alpha_n}$

The Attack

- adversary sets $R^* = PK^{\alpha_0}PK_1^{\alpha_1}\cdots PK_n^{\alpha_n}$ for random $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{Z}_p$
- valid forgery: $R^*PK^{c^*} = g^{z^*}$
- $R^*PK^{c^*} = PK^{c^*+\alpha_0}PK_1^{\alpha_1}\cdots PK_n^{\alpha_n}$
- $sk_i = f(i) = a_0 + a_1 i + \ldots + a_t i^t$ where $\vec{v}_i = (1, i, i^2, \ldots, i^t)$ are Vandermonde vectors

The Attack

- adversary sets $R^* = PK^{\alpha_0} PK_1^{\alpha_1} \cdots PK_n^{\alpha_n}$ for random $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{Z}_p$
- valid forgery: $R^* PK^{c^*} = g^{z^*}$
- $R^* PK^{c^*} = PK^{c^* + \alpha_0} PK_1^{\alpha_1} \cdots PK_n^{\alpha_n}$
- $sk_i = f(i) = a_0 + a_1 i + \ldots + a_t i^t$ where $\vec{v}_i = (1, i, i^2, \ldots, i^t)$ are Vandermonde vectors
- compute $\overrightarrow{w} = c^* \vec{v}_0 + \sum_{i=0}^{n} \alpha_i \vec{v}_i$ where $c^* = H(PK, m^*, R^*)$

The Attack

The Attack

- uses oracle for solving problem P to obtain set $CS \subseteq \{1,\ldots,n\}$ with $|CS| = t_c$ such that $\overrightarrow{w} \in span(\{\vec{v}_{i \in CS}\})$

The Attack

- uses oracle for solving problem P to obtain set $CS \subseteq \{1,\ldots,n\}$ with $|CS| = t_c$ such that $\overrightarrow{w} \in span(\{\vec{v}_{i \in CS}\})$
- corrupts all parties in CS to obtain $\{sk_i\}_{i \in CS}$

The Attack

- uses oracle for solving problem P to obtain set $CS \subseteq \{1,\ldots,n\}$ with $|CS| = t_c$ such that $\overrightarrow{w} \in span(\{\vec{v}_{i \in CS}\})$
- corrupts all parties in CS to obtain $\{sk_i\}_{i \in CS}$
- computes (via linear algebra) $\{\beta_j\}_{j \in CS}$ such that:

The Attack

- uses oracle for solving problem P to obtain set $CS \subseteq \{1,\ldots,n\}$ with $|CS| = t_c$ such that $\vec{w} \in span(\{\vec{v}_{i \in CS}\})$
- corrupts all parties in CS to obtain $\{sk_i\}_{i \in CS}$
- computes (via linear algebra) $\{\beta_j\}_{j \in CS}$ such that:
- $\vec{w} = c^* \vec{v}_0 + \sum_{i=0}^{n} \alpha_i \vec{v}_i = \sum_{j \in CS} \beta_j \vec{v}_j$

The Attack

- uses oracle for solving problem P to obtain set $CS \subseteq \{1, \ldots, n\}$ with $|CS| = t_c$ such that $\overrightarrow{w} \in span(\{\vec{v}_{i \in CS}\})$
- corrupts all parties in CS to obtain $\{sk_i\}_{i \in CS}$
- computes (via linear algebra) $\{\beta_j\}_{j \in CS}$ such that:
- $\overrightarrow{w} = c^* \vec{v}_0 + \sum_{i=0}^{n} \alpha_i \vec{v}_i = \sum_{j \in CS} \beta_j \vec{v}_j$
- finally, computes $z^* = \sum_{j \in CS} \beta_j sk_j$

Attack Success

$(n, t+1)$	$t_c = t$	$t_c = t-1$	$t_c = t-2$	$t_c = t-3$
(64,43)	195.84	446.97	698.2	949.52
(128,86)	137.87	388.92	640.02	891.17
(196,131)	75.41	326.45	577.53	828.64
(512,342)	0.0	37.25	288.28	539.32
(768,513)	0.0	0.0	53.8	304.82
(1024,683)	0.0	0.0	0.0	69.29

Table 2. The probability that our attack succeeds is 2^{-x} for x given in the table, with $p \approx 2^{252}$, where x is computed as in Theorem 2. Here, n is the total number of potential signers, $t+1$ is the threshold, and t_c is the corruption threshold.

Attack Success

$(n, t+1)$	$t_c = t$	$t_c = t-1$	$t_c = t-2$	$t_c = t-3$
(64,43)	195.84	446.97	698.2	949.52
(128,86)	137.87	388.92	640.02	891.17
(196,131)	75.41	326.45	577.53	828.64
(512,342)	0.0	37.25	288.28	539.32
(768,513)	0.0	0.0	53.8	304.82
(1024,683)	0.0	0.0	0.0	69.29

Table 2. The probability that our attack succeeds is 2^{-x} for x given in the table, with $p \approx 2^{252}$, where x is computed as in Theorem 2. Here, n is the total number of potential signers, $t+1$ is the threshold, and t_c is the corruption threshold.

Insecure

Attack Success

$(n, t+1)$	$t_c = t$	$t_c = t-1$	$t_c = t-2$	$t_c = t-3$
(64,43)	195.84	446.97	698.2	949.52
(128,86)	137.87	388.92	640.02	891.17
(196,131)	75.41	326.45	577.53	828.64
(512,342)	0.0	37.25	288.28	539.32
(768,513)	0.0	0.0	53.8	304.82
(1024,683)	0.0	0.0	0.0	69.29

Table 2. The probability that our attack succeeds is 2^{-x} for x given in the table, with $p \approx 2^{252}$, where x is computed as in Theorem 2. Here, n is the total number of potential signers, $t+1$ is the threshold, and t_c is the corruption threshold.

Insecure

Implications of Our Results

Our results have two striking implications:

1. If P is easy to solve, all schemes meeting Conditions 1-3 are statically secure but not adaptively secure

   Would be first such separation for any natural protocol, solving a long-standing open problem in MPC

   Moreover, would apply to a large class of schemes and would hold even in the strongest idealized models: the AGM and the GGM

Implications of Our Results

2. The full adaptive security of these schemes cannot be proven without an assumption that implies the hardness of some instances of P

   Such an assumption would likely go beyond assumptions about the group and ROs since P is not defined in terms of them

   Moreover, this extends to corruption thresholds below $t_c = t$

Call to Action

- attack is “plausible” because we do not know if the problem P is easy to solve or not
- some preliminary analysis, but further investigation needed

On the Adaptive Security of FROST

Elizabeth Crites
Web3 Foundation

Jonathan Katz
Google

Chelsea Komlo
University of Waterloo
NEAR One

Stefano Tessaro
University of Washington

Chenzhi Zhu
University of Washington

FROST

- Flexible Round-Optimized Schnorr Threshold signatures
- 2 rounds
  - 1 offline pre-processing round, 1 online signing round
  - static security in the ROM under AOMDL
  - OMDL: Given $(X_0, X_1, \ldots, X_t)$ and t queries to a DL solution oracle $\mathcal{O}^{DL}(X)$, output $t+1$ discrete logs $(x_0, x_1, \ldots, x_t)$
  - AOMDL: falsifiable variant of OMDL

Threshold Schnorr Signatures

How to share sk ?

How to share r ?

$$z \leftarrow r + c \cdot sk$$

$$sig = (R, z)$$

FROST

- To sign a message m, party P_i
  - Round 1: samples $r_i, s_i \xleftarrow{\$} \mathbb{Z}_p$, sets $R_i \leftarrow g^{r_i}$, $S_i \leftarrow g^{s_i}$, and outputs R_i, S_i

$m, \mathcal{S} \rightarrow$

  - Round 2: computes
    - $a_i \leftarrow H'(i, PK, m, \{R_i, S_i\}_{i \in \mathcal{S}})$
    - $R = \Pi_{i \in \mathcal{S}} R_i \cdot S_i^{a_i}$
    - $c \leftarrow H(PK, m, R)$
    - $z_i \leftarrow r_i + a_i \cdot s_i + c \cdot \lambda_i^{\mathcal{S}} \cdot sk_i$
    - outputs z_i

$z = \Sigma_{i \in \mathcal{S}} z_i$

$sig = (R, z)$

$R \cdot PK^c \stackrel{?}{=} g^z$

Optimizations FROST2 / FROST3

- FROST2 computational optimization of FROST
- FROST3 improves communication complexity of FROST2
- we prove adaptive security of all 3 variants

IRTF FROST Standardization

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

D. Connolly
Zcash Foundation
C. Komlo
University of Waterloo, Zcash Foundation
I. Goldberg
University of Waterloo
C. A. Wood
Cloudflare
June 2024

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

Our Main Results

Our Main Results

1. FROST/2/3 secure up to $t/2$ adaptive corruptions in the ROM under AOMDL
    - same as the original assumptions for FROST static security

Our Main Results

1. FROST/2/3 secure up to $t/2$ adaptive corruptions in the ROM under AOMDL
    - same as the original assumptions for FROST static security
2. FROST/2/3 secure up to t (i.e., full) adaptive corruptions in the AGM+ROM under AOMDL+LDVR (our new assumption)

Our Main Results

1. FROST/2/3 secure up to $t/2$ adaptive corruptions in the ROM under AOMDL
    - same as the original assumptions for FROST static security
2. FROST/2/3 secure up to t (i.e., full) adaptive corruptions in the AGM+ROM under AOMDL+LDVR (our new assumption)
3. Unconditional hardness of LDVR for interesting values above $t/2$

The LDVR Problem

MAIN $\mathsf{Expt}_{\mathcal{A}}^{(t_c,t,n)\text{-ldvr}}(\kappa)$

$\mathsf{ctr} := 0$

$(p, st) \leftarrow_\$ \mathcal{A}(\kappa)$

// $2^\kappa < p < 2^{\kappa+1}$, p prime

for $j \in \{0, \ldots, n\}$ **do**

  $\boldsymbol{v}_j := (1, j, \ldots, j^t) \in \mathbb{Z}_p^{t+1}$

$(\mathrm{CS}, i^*) \leftarrow_\$ \mathcal{A}^{\mathcal{O}}(st)$

// $\mathrm{CS} \subseteq \{1, \ldots, n\}, |\mathrm{CS}| \le t_c, i^* \in [\mathsf{ctr}]$

$$\boldsymbol{w} := c_{i^*}\boldsymbol{v}_0 + \sum_{j=0}^{n} \boldsymbol{\alpha}_{i^*}[j] \cdot \boldsymbol{v}_j$$

if $\boldsymbol{w} \in \mathsf{span}(\{\boldsymbol{v}_i\}_{i \in \mathrm{CS}})$

  return 1

return 0

$\mathcal{O}(\boldsymbol{\alpha})$

// $\boldsymbol{\alpha} \in \mathbb{Z}_p^{n+1}$

$\mathsf{ctr} := \mathsf{ctr} + 1$

$\boldsymbol{\alpha}_{\mathsf{ctr}} := \boldsymbol{\alpha}$

$c_{\mathsf{ctr}} \leftarrow_\$ \mathbb{Z}_p$

return c_{ctr}

Fig. 6. The LDVR experiment with parameters $t_c \le t < n$.

The LDVR Problem

Definition 2. P *is the following search problem. Given* $\boldsymbol{w} \in \mathbb{Z}_p^{t+1}$ *and* $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \mathbb{Z}_p^{t+1}$*, find a set* $CS \subset \{1, \ldots, n\}$ *with* $|CS| = t_c$ *such that* $\boldsymbol{w} \in span(\{\boldsymbol{v}_i\}_{i \in CS})$ *if one exists.*

MAIN $\mathsf{Expt}_{\mathcal{A}}^{(t_c,t,n)\text{-ldvr}}(\kappa)$

$\mathsf{ctr} := 0$

$(p, st) \leftarrow\$ \mathcal{A}(\kappa)$

// $2^\kappa < p < 2^{\kappa+1}$, p prime

for $j \in \{0, \ldots, n\}$ **do**

  $\boldsymbol{v}_j := (1, j, \ldots, j^t) \in \mathbb{Z}_p^{t+1}$

$(\mathrm{CS}, i^*) \leftarrow\$ \mathcal{A}^{\mathcal{O}}(st)$

// $\mathrm{CS} \subseteq \{1, \ldots, n\}, |\mathrm{CS}| \leq t_c, i^* \in [\mathsf{ctr}]$

$$\boldsymbol{w} := c_{i^*} \boldsymbol{v}_0 + \sum_{j=0}^{n} \boldsymbol{\alpha}_{i^*}[j] \cdot \boldsymbol{v}_j$$

if $\boldsymbol{w} \in \mathsf{span}(\{\boldsymbol{v}_i\}_{i \in \mathrm{CS}})$

  return 1

return 0

$\mathcal{O}(\boldsymbol{\alpha})$

// $\boldsymbol{\alpha} \in \mathbb{Z}_p^{n+1}$

$\mathsf{ctr} := \mathsf{ctr} + 1$

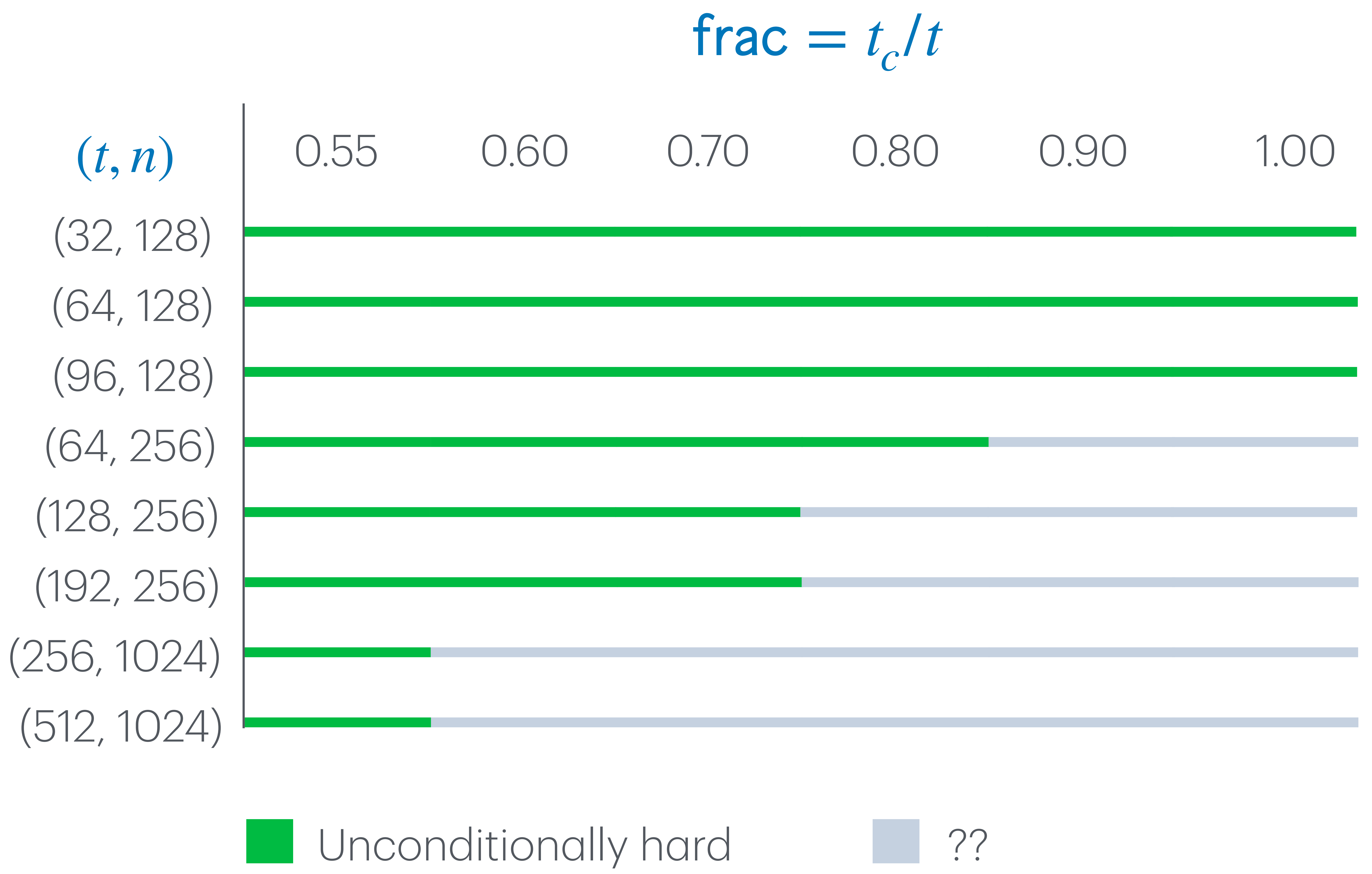
$\boldsymbol{\alpha}_{\mathsf{ctr}} := \boldsymbol{\alpha}$

$c_{\mathsf{ctr}} \leftarrow\$ \mathbb{Z}_p$

return c_{ctr}

Fig. 6. The LDVR experiment with parameters $t_c \leq t < n$.

Unconditional Hardness of LDVR



Half Adaptive Security Proof

- FROST/2/3 for up to $t/2$ adaptive corruptions in the ROM under AOMDL
  - same assumptions as static FROST
  - similar structure to static FROST proof

Full Adaptive Security Proof

Full Adaptive Security Proof

- FROST/2/3 for up to t adaptive corruptions in the AGM+ROM under AOMDL+LDVR

Full Adaptive Security Proof

- FROST/2/3 for up to t adaptive corruptions in the AGM+ROM under AOMDL+LDVR
- when adversary queries $c^* = H(PK, m^*, R^*)$, it must output representation:

Full Adaptive Security Proof

- FROST/2/3 for up to t adaptive corruptions in the AGM+ROM under AOMDL+LDVR
- when adversary queries $c^* = H(PK, m^*, R^*)$, it must output representation:

$$R^* = g^{\delta} \cdot \mathsf{PK}^{\beta} \cdot \prod_{k=1}^{n} \mathsf{PK}_k^{\beta_k} \prod_{i=1}^{q_s} R_{i,1}^{\gamma_{i,1}} S_{i,1}^{\gamma'_{i,1}} \cdots R_{i,n}^{\gamma_{i,n}} S_{i,n}^{\gamma'_{i,n}}$$

Full Adaptive Security Proof

- FROST/2/3 for up to t adaptive corruptions in the AGM+ROM under AOMDL+LDVR
- when adversary queries $c^* = H(PK, m^*, R^*)$, it must output representation:

$$R^* = g^{\delta} \cdot \mathsf{PK}^{\beta} \cdot \prod_{k=1}^{n} \mathsf{PK}_k^{\beta_k} \prod_{i=1}^{q_s} \underbrace{R_{i,1}^{\gamma_{i,1}} S_{i,1}^{\gamma'_{i,1}} \cdots R_{i,n}^{\gamma_{i,n}} S_{i,n}^{\gamma'_{i,n}}}$$

Can replace with g and $PK_i's$?

Full Adaptive Security Proof

- FROST/2/3 for up to t adaptive corruptions in the AGM+ROM under AOMDL+LDVR

- when adversary queries $c^* = H(PK, m^*, R^*)$, it must output representation:

$$R^* = g^{\delta} \cdot \mathsf{PK}^{\beta} \cdot \prod_{k=1}^{n} \mathsf{PK}_k^{\beta_k} \prod_{i=1}^{q_s} \underbrace{R_{i,1}^{\gamma_{i,1}} S_{i,1}^{\gamma'_{i,1}} \cdots R_{i,n}^{\gamma_{i,n}} S_{i,n}^{\gamma'_{i,n}}}$$

Can replace with g and $PK_i's$?

- "Want" this in order to break LDVR

Full Adaptive Security Proof

- FROST/2/3 for up to t adaptive corruptions in the AGM+ROM under AOMDL+LDVR

- when adversary queries $c^* = H(PK, m^*, R^*)$, it must output representation:

$$R^* = g^{\delta} \cdot \mathsf{PK}^{\beta} \cdot \prod_{k=1}^{n} \mathsf{PK}_k^{\beta_k} \prod_{i=1}^{q_s} \underbrace{R_{i,1}^{\gamma_{i,1}} S_{i,1}^{\gamma'_{i,1}} \cdots R_{i,n}^{\gamma_{i,n}} S_{i,n}^{\gamma'_{i,n}}}$$

Can replace with g and $PK_i's$?

- "Want" this in order to break LDVR

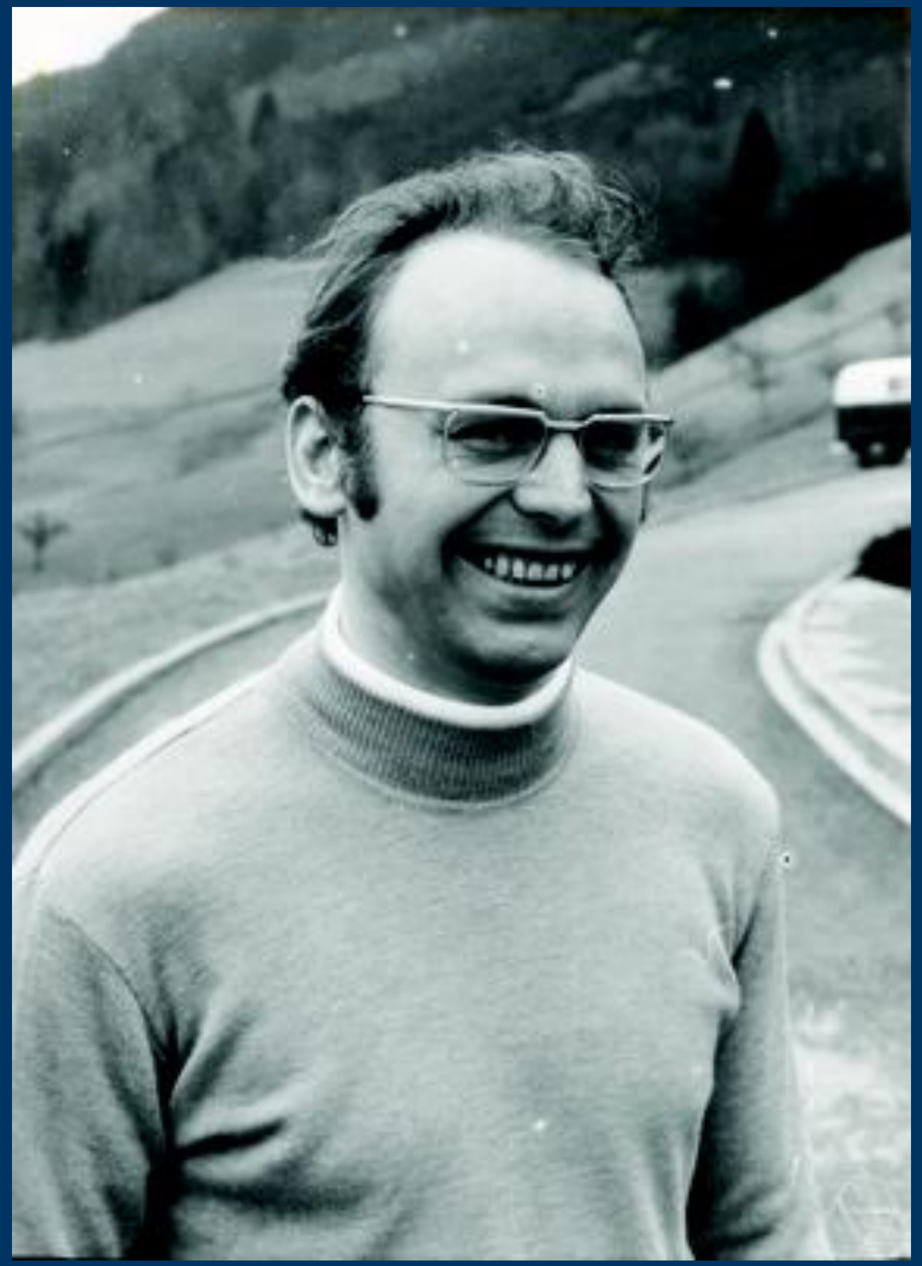
- If no, can break AOMDL instead

Call to Action

- we do not know if P or LDVR is easy or hard (beyond the unconditional bound)
- other schemes may be proven under variants of these assumptions

Plausible Attack

Adaptive FROST



Claus-Peter Schnorr

1943-2025

From the Oberwolfach Photo Collection