Round-Optimal, Fully Secure Distributed Key Generation

Jonathan Katz
Google and University of Maryland*

Work done while at Dfns Labs

*This work was not part of my UMD duties or responsibilities

Threshold cryptography

Goal: Share a secret key among n parties, such that:

- Any $t+1$ parties can jointly perform some cryptographic operation
- An adversary compromising up to t parties cannot

Threshold cryptography

Goal: Share a secret key among n parties, such that:

- Any $t+1$ parties can jointly perform some cryptographic operation
- An adversary compromising up to t parties cannot

Two components of a threshold cryptosystem:

1. Key distribution, either via a trusted dealer or a distributed key generation (DKG) protocol
2. Distributed protocol for signing, decrypting, etc.

Our results

Focus on fully secure DKG in the dlog setting

- Define security via an appropriate ideal functionality
  - Modular: secure DKG protocols can be composed with arbitrary (secure) threshold protocols
  - Cleaner; security guarantees more clear

Our results

Focus on fully secure DKG in the dlog setting

- Define security via an appropriate ideal functionality
  - Modular: secure DKG protocols can be composed with arbitrary (secure) threshold protocols
  - Cleaner; security guarantees more clear
- Study the round complexity of fully secure DKG in the honest-majority setting (assuming synchrony + broadcast)

Our results

Focus on fully secure DKG in the dlog setting

- Define security via an appropriate ideal functionality
  - Modular: secure DKG protocols can be composed with arbitrary (secure) threshold protocols
  - Cleaner; security guarantees more clear
- Study the round complexity of fully secure DKG in the honest-majority setting (assuming synchrony + broadcast)
  - Lower bound: No one-round protocols (regardless of setup)

Our results

Focus on fully secure DKG in the dlog setting

- Define security via an appropriate ideal functionality
  - Modular: secure DKG protocols can be composed with arbitrary (secure) threshold protocols
  - Cleaner; security guarantees more clear
- Study the round complexity of fully secure DKG in the honest-majority setting (assuming synchrony + broadcast)
  - Lower bound: No one-round protocols (regardless of setup)
  - Upper bound: Several round-optimal protocols with tradeoffs in terms of efficiency, setup, and assumptions

DKG in the dlog setting

DKG in the dlog setting

Notation

- n is the total number of parties
- t is an upper bound on the number of corrupted parties
- $\mathbb{G}$ is a cyclic group of prime order q, with generator g

DKG in the dlog setting

Notation

- n is the total number of parties
- t is an upper bound on the number of corrupted parties
- $\mathbb{G}$ is a cyclic group of prime order q, with generator g

Goal

Distributed protocol for n parties to generate

- Common public key $y = g^x$

DKG in the dlog setting

Notation

- n is the total number of parties
- t is an upper bound on the number of corrupted parties
- $\mathbb{G}$ is a cyclic group of prime order q, with generator g

Goal

Distributed protocol for n parties to generate

- Common public key $y = g^x$
- $(t+1)$-out-of-n secret sharing[a] $\{\sigma_i\}_{i=1}^n$ of the private key x

DKG in the dlog setting

Notation

- n is the total number of parties
- t is an upper bound on the number of corrupted parties
- $\mathbb{G}$ is a cyclic group of prime order q, with generator g

Goal

Distributed protocol for n parties to generate

- Common public key $y = g^x$
- $(t+1)$-out-of-n secret sharing[a] $\{\sigma_i\}_{i=1}^n$ of the private key x
- Common commitments $\{g^{\sigma_i}\}_{i=1}^n$ to the parties' shares

[a] Assume Shamir secret sharing, but it could also be n-out-of-n additive sharing

DKG in the dlog setting

Setup

Parties may have some (correlated) state before protocol execution, e.g.,

- CRS
- PKI
- ROM
- Correlated randomness

DKG in the dlog setting

Setup

Parties may have some (correlated) state before protocol execution, e.g.,

- CRS
- PKI
- ROM
- Correlated randomness

Ideally, state suffices for an unbounded (polynomial) number of executions

"Full security"

Desired security properties:

- Corrupted parties should not learn anything about x (beyond what is implied by y)
- Honest parties should hold a correct sharing of x (and commitments to other parties' shares)

"Full security"

Desired security properties:

- Corrupted parties should not learn anything about x (beyond what is implied by y)
- Honest parties should hold a correct sharing of x (and commitments to other parties' shares)
- Unbiasable: Corrupted parties should be unable to bias y

“Full security”

Desired security properties:

- Corrupted parties should not learn anything about x (beyond what is implied by y)
- Honest parties should hold a correct sharing of x (and commitments to other parties’ shares)
- Unbiasable: Corrupted parties should be unable to bias y
- Robustness (aka guaranteed output delivery): Corrupted parties should be unable to prevent generation of a key

"Full security"

Desired security properties:

- Corrupted parties should not learn anything about x (beyond what is implied by y)
- Honest parties should hold a correct sharing of x (and commitments to other parties' shares)
- Unbiasable: Corrupted parties should be unable to bias y
- Robustness (aka guaranteed output delivery): Corrupted parties should be unable to prevent generation of a key
- ...

"Full security"

Desired security properties:

- Corrupted parties should not learn anything about x (beyond what is implied by y)
- Honest parties should hold a correct sharing of x (and commitments to other parties' shares)
- Unbiasable: Corrupted parties should be unable to bias y
- Robustness (aka guaranteed output delivery): Corrupted parties should be unable to prevent generation of a key
- . . .

Define security via an ideal functionality in a simulation-based framework

Ideal functionalities for (dlog-based) DKG

There are multiple ideal functionalities one could consider for DKG (see paper for examples and discussion)

Here: (one possible) ideal functionality for fully secure DKG

Ideal functionality for fully secure DKG (cf. [Wik04])

(For simplicity, assume $|\mathcal{C}| = t$)

$\mathcal{F}_{\mathsf{DKG}}^{t,n}$

1. Receive $\{\sigma_i\}_{i \in \mathcal{C}}$ from the adversary.
2. Choose $x \leftarrow \mathbb{Z}_q$ and set $y := g^x$.
3. Let f be the polynomial of degree at most t such that $f(0) = x$ and $f(i) = \sigma_i$ for $i \in \mathcal{C}'$. Set $\sigma_i := f(i)$ for $i \in [n] \setminus \mathcal{C}'$.
4. For $i \in [n]$, set $y_i := g^{\sigma_i}$. Let $Y := (y_1, \ldots, y_n)$.
5. For $i \in [n]$, send (y, σ_i, Y) to P_i.

Ideal functionality for fully secure DKG (cf. [Wik04])

(For simplicity, assume $|\mathcal{C}| = t$)

$$\mathcal{F}_{\mathsf{DKG}}^{t,n}$$

1. Receive $\{\sigma_i\}_{i \in \mathcal{C}}$ from the adversary.
2. Choose $x \leftarrow \mathbb{Z}_q$ and set $y := g^x$.
3. Let f be the polynomial of degree at most t such that $f(0) = x$ and $f(i) = \sigma_i$ for $i \in \mathcal{C}'$. Set $\sigma_i := f(i)$ for $i \in [n] \setminus \mathcal{C}'$.
4. For $i \in [n]$, set $y_i := g^{\sigma_i}$. Let $Y := (y_1, \ldots, y_n)$.
5. For $i \in [n]$, send (y, σ_i, Y) to P_i.

Impossible to t-securely realize unless $t < n/2$

Prior work

Lots of DKG protocols, but very few achieving full security

Most round-efficient (explicit) fully secure DKG protocol:

- 6 rounds [GJKR07]

Based on generic (honest-majority) MPC [GLS15, G+21, D+21]:

- 3 rounds with a CRS; 2 rounds with a CRS + PKI
    - complex / impractical / based on strong cryptographic assumptions

Prior work

Lots of DKG protocols, but very few achieving full security

Most round-efficient (explicit) fully secure DKG protocol:

- 6 rounds [GJKR07]

Based on generic (honest-majority) MPC [GLS15, G+21, D+21]:

- 3 rounds with a CRS; 2 rounds with a CRS + PKI
  - complex / impractical / based on strong cryptographic assumptions

Impossibility results for 1-round MPC with guaranteed output delivery do not apply here

Impossibility result

Fully secure DKG is impossible in one round, regardless of prior setup

- Even without robustness
- Even tolerating only a single corrupted party

Two-round protocols?

Two-round protocols?

Note we assume a rushing adversary . . .

Natural strategy

Protocol	Simulation
1. Parties commit to shares	1. Simulator extracts shares of corrupted parties
2. Parties decommit their shares	2. Corrupted parties open to extracted values; (simulated) honest parties force output to desired value

Two-round protocols?

Note we assume a rushing adversary . . .

Natural strategy

Protocol	Simulation
1. Parties commit to shares	1. Simulator extracts shares of corrupted parties
2. Parties decommit their shares	2. Corrupted parties open to extracted values; (simulated) honest parties force output to desired value

Problem: Some corrupted parties can abort in the second round. . .

Positive results

Intuitively, need protocols with the following property:

- Key is determined at the end of the first round (regardless of what corrupted parties do in the second round), but the adversary cannot compute it!

Positive results

Setup	Rounds	Assumptions
CRS + PKI	2	NIZK + PKE
CRS	2	NIZK + MP-NIKE
ROM + 1-round preprocessing	2	(none)

Positive results

Setup	Rounds	Assumptions
CRS + PKI	2	NIZK + PKE
CRS	2	NIZK + MP-NIKE
ROM + 1-round preprocessing	2	—
CRS + 2-round preprocessing	1	NIZK + OWF

(See also concurrent work [BHL24])

Positive results

Setup	Rounds	Assumptions
CRS + PKI	2	NIZK + PKE
CRS	2	NIZK + MP-NIKE
ROM + 1-round preprocessing	2	—
CRS + 2-round preprocessing	1	NIZK + OWF

(See also concurrent work [BHL24])

Fully secure* DKG is impossible in one round (regardless of prior setup)

* Impossibility only holds for statistically unbiased protocols

Positive results

Setup	Rounds	Assumptions
CRS + PKI	2	NIZK + PKE
CRS	2	NIZK + MP-NIKE
ROM + 1-round preprocessing	2	—
CRS + 2-round preprocessing	1	NIZK + OWF

Based on hash functions alone

Very efficient for moderate t, n

Background: Pseudorandom secret sharing [CDI05]

Notation

Let $\mathbb{S}_{n-t,n}$ be the collection of all subsets of $[n]$ of size $n-t$

For $S \in \mathbb{S}_{n-t,n}$, let $Z_S \in \mathbb{Z}_q[X]$ be the degree-t polynomial with $Z_S(0) = 1$ and $Z_S(i) = 0$ for $i \in [n] \setminus S$

Let $F : \{0,1\}^\kappa \times \{0,1\}^\ell \to \mathbb{Z}_q$ be a pseudorandom function

Background: Pseudorandom secret sharing [CDI05]

Notation

Let $\mathbb{S}_{n-t,n}$ be the collection of all subsets of $[n]$ of size $n-t$

For $S \in \mathbb{S}_{n-t,n}$, let $Z_S \in \mathbb{Z}_q[X]$ be the degree-t polynomial with $Z_S(0) = 1$ and $Z_S(i) = 0$ for $i \in [n] \setminus S$

Let $F : \{0,1\}^\kappa \times \{0,1\}^\ell \to \mathbb{Z}_q$ be a pseudorandom function

Assume for all $S \in \mathbb{S}_{n-t,n}$ and all $i \in S$, party P_i holds $k_S \in \{0,1\}^\kappa$

Background: Pseudorandom secret sharing [CDI05]

Notation

Let $\mathbb{S}_{n-t,n}$ be the collection of all subsets of $[n]$ of size $n-t$

For $S \in \mathbb{S}_{n-t,n}$, let $Z_S \in \mathbb{Z}_q[X]$ be the degree-t polynomial with $Z_S(0) = 1$ and $Z_S(i) = 0$ for $i \in [n] \setminus S$

Let $F : \{0,1\}^\kappa \times \{0,1\}^\ell \to \mathbb{Z}_q$ be a pseudorandom function

Assume for all $S \in \mathbb{S}_{n-t,n}$ and all $i \in S$, party P_i holds $k_S \in \{0,1\}^\kappa$

Given a nonce $N \in \{0,1\}^\ell$, each party P_i can compute the share

$$\sigma_i := \sum_{S \in \mathbb{S}_{n-t,n} \,:\, i \in S} F_{k_S}(N) \cdot Z_S(i)$$

Background: Pseudorandom secret sharing [CDI05]

Notation

Let $\mathbb{S}_{n-t,n}$ be the collection of all subsets of $[n]$ of size $n-t$

For $S \in \mathbb{S}_{n-t,n}$, let $Z_S \in \mathbb{Z}_q[X]$ be the degree-t polynomial with $Z_S(0) = 1$ and $Z_S(i) = 0$ for $i \in [n] \setminus S$

Let $F : \{0,1\}^\kappa \times \{0,1\}^\ell \to \mathbb{Z}_q$ be a pseudorandom function

Assume for all $S \in \mathbb{S}_{n-t,n}$ and all $i \in S$, party P_i holds $k_S \in \{0,1\}^\kappa$

Given a nonce $N \in \{0,1\}^\ell$, each party P_i can compute the share

$$\sigma_i := \sum_{S \in \mathbb{S}_{n-t,n} : i \in S} F_{k_S}(N) \cdot Z_S(i)$$

This is a $(t+1)$-out-of-n Shamir secret sharing of

$$x_N = \sum_{S \in \mathbb{S}_{n-t,n}} F_{k_S}(N) \cdot Z_S(0) = \sum_{S \in \mathbb{S}_{n-t,n}} F_{k_S}(N)$$

DKG from PRSS

PRSS implies a one-round (semi-honest) DKG protocol:

- For each set $S \in \mathbb{S}_{n-t,n}$, a designated party broadcasts $\hat{y}_S := g^{F_{k_S}(N)}$
- Parties compute public key $y = g^{x_N}$ from the $\{\hat{y}_S\}$

DKG from PRSS

PRSS implies a one-round (semi-honest) DKG protocol:

- For each set $S \in \mathbb{S}_{n-t,n}$, a designated party broadcasts $\hat{y}_S := g^{F_{k_S}(N)}$
- Parties compute public key $y = g^{x_N}$ from the $\{\hat{y}_S\}$

Problems:

- Corrupted party may broadcast incorrect $\hat{y}_S$
  - Even if multiple parties in S broadcast $\hat{y}_S$, other parties don't know which value is correct

DKG from PRSS

PRSS implies a one-round (semi-honest) DKG protocol:

- For each set $S \in \mathbb{S}_{n-t,n}$, a designated party broadcasts $\hat{y}_S := g^{F_{k_S}(N)}$
- Parties compute public key $y = g^{x_N}$ from the $\{\hat{y}_S\}$

Problems:

- Corrupted party may broadcast incorrect $\hat{y}_S$
  - Even if multiple parties in S broadcast $\hat{y}_S$, other parties don't know which value is correct
- PRSS assumes a trusted dealer, which we want to avoid

A fully secure DKG protocol

A fully secure protocol (high-level):

- Round 1: All parties in S broadcast a "deterministic commitment" to $\hat{y}_S$ (i.e., $H(\hat{y}_S)$)
  - If there is disagreement, ignore S (equivalent to treating $F_{k_S}(N) = 0$, $\hat{y}_S = 1$)

A fully secure DKG protocol

A fully secure protocol (high-level):

- Round 1: All parties in S broadcast a "deterministic commitment" to $\hat{y}_S$ (i.e., $H(\hat{y}_S)$)
  - If there is disagreement, ignore S (equivalent to treating $F_{k_S}(N) = 0$, $\hat{y}_S = 1$)
- Round 2: Parties reveal $\hat{y}_S$
  - Incorrect preimages of $H(\hat{y}_S)$ ignored
- Parties compute public key $y = g^{x_N}$ from the $\{\hat{y}_S\}$

A fully secure DKG protocol

A fully secure protocol (high-level):

- Round 1: All parties in S broadcast a "deterministic commitment" to $\hat{y}_S$ (i.e., $H(\hat{y}_S)$)
  - If there is disagreement, ignore S (equivalent to treating $F_{k_S}(N) = 0$, $\hat{y}_S = 1$)
- Round 2: Parties reveal $\hat{y}_S$
  - Incorrect preimages of $H(\hat{y}_S)$ ignored
- Parties compute public key $y = g^{x_N}$ from the $\{\hat{y}_S\}$

No longer any need for a trusted dealer – a designated party in each set S can simply distribute k_S in a preprocessing phase!

- Note: we do not assume correct behavior during preprocessing

A fully secure DKG protocol

Theorem

Let F be a pseudorandom function, and model H as a random oracle. Then for $t < n/2$ this protocol t-securely realizes $\mathcal{F}^{t,n}_{\mathsf{DKG}}$.

A small modification to the protocol achieves adaptive security (assuming secure erasure)

Proof intuition

Useful observations:

- Every $S \in \mathbb{S}_{n-t,n}$ contains at least one honest party
- There exists a set $S_{\mathcal{H}} \in \mathbb{S}_{n-t,n}$ containing only honest parties

Proof intuition

Useful observations:

- Every $S \in \mathbb{S}_{n-t,n}$ contains at least one honest party
- There exists a set $S_{\mathcal{H}} \in \mathbb{S}_{n-t,n}$ containing only honest parties

Robustness/no bias: Fix some $S \in \mathbb{S}_{n-t,n}$.

- If there is disagreement among the $\{h_{i,S}\}_{i \in S}$, then S is excluded
- Otherwise, a preimage $\hat{y}_S$ for the common value h_S will be sent (since S contains an honest party)
- Moreover, at most one preimage will be sent (by collision resistance)

Proof intuition

Useful observations:

- Every $S \in \mathbb{S}_{n-t,n}$ contains at least one honest party
- There exists a set $S_{\mathcal{H}} \in \mathbb{S}_{n-t,n}$ containing only honest parties

Robustness/no bias: Fix some $S \in \mathbb{S}_{n-t,n}$.

- If there is disagreement among the $\{h_{i,S}\}_{i \in S}$, then S is excluded
- Otherwise, a preimage $\hat{y}_S$ for the common value h_S will be sent (since S contains an honest party)
- Moreover, at most one preimage will be sent (by collision resistance)

Secrecy: $S_{\mathcal{H}}$ is never excluded, so the pseudorandom contribution $k_{S_{\mathcal{H}}}$ is always included in the effective private key

Open questions

Open questions

- Some of our protocols have complexity $O(\binom{n}{t})$ – can this be improved?
- Some of our protocols rely on preprocessing – can this be avoided?
- Is 2-round fully secure DKG in the plain model possible?

Thank you!

Paper available at `https://eprint.iacr.org/2023/1094`