# Threshold Signatures

## Part 3: Quantum Resistant Schemes

*Bar-Ilan University Winter School on Cryptography*

Rosario Gennaro

Protocol Labs Research

D.Boneh, G, S.Goldfeder, A.Jain, S.Kim, P.M.R. Rasmussen, A.Sahai: Threshold Cryptosystems from Threshold Fully Homomorphic Encryption. CRYPTO (1) 2018: 565-596

# How to thresholdize any scheme

We are going to show how to use Threshold Fully Homomorphic Encryption (TFHE) to build a universal thresholdizer: a compiler that takes any cryptographic scheme and builds a non-interactive threshold version of it.

C.Gentry, A.Sahai, and B.Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. CRYPTO 2013.

# Let's recall the GSW13 FHE Scheme

- The secret key is a vector $sk \in Z_q^{\,l}$
- A ciphertext is a matrix $ct \in Z_q^{\,l x m}$
- To decrypt we take the inner product of a column $ct^k$ of $ct$ with $sk$
  - If $d = \langle ct^k, sk \rangle$ is small then the plaintext bit is 0 otherwise is 1

- A $n$-out-of-n scheme follows:
  - Split $sk = sk_1 + \dots + sk_n$
  - Party $i$ outputs $d_i = \langle ct^k, sk_i \rangle + noise$
    - The noise is needed to hide the secret share from reconstruction
    - $d \sim d_1 + \dots + d_n$

# The problem with threshold

- If we split $sk$ with Shamir
- Let $[sk_1 \dots sk_n]$ be the shares
- If Party $i$ outputs $d_i = \langle ct^k, sk_i \rangle + noise$
    - When we interpolate with the Lagrangians $\sum_{i \in S} \lambda_{i,S}\, d_i$
    - The noise is the combination is not guaranteed to be small anymore
    - $d$ is very far from $\sum_{i \in S} \lambda_{i,S}\, d_i$

## First solution

# Use Linear Secret Sharing with binary coefficients

- We split ***sk*** with a secret sharing scheme
  - Which is linear (so that we can still easily compute the inner product)
  - And reconstruction involves only 1/0 coefficients
- Let $[sk_1 \dots sk_n]$ be the shares
- Party ***i*** outputs $d_i = \langle ct^k, sk_i \rangle + noise$
  - We then reconstruct $\sum_{i \in S} \beta_{i,S}\, d_i$
  - $d \sim \sum_{i \in S} \beta_{i,S}\, d_i$
  - Since the combined noise is small (because $\beta_{i,S}$ is binary)

First solution

# How expressive are {0,1}-LSSS

- It turns out that they are quite expressive
  - They include threshold access structures
- The drawback is that they are not very efficient
  - For $n$ players the shares grow as $n^4$

Second Solution

# Grow the parameters to accommodate the noise

- Split ***sk*** with Shamir
- Let $[sk_1 \dots sk_n]$ be the shares
- Party ***i*** outputs $d_i = \langle ct^k, sk_i \rangle + noise$
    - Remove the denominators to make the Lagrangian integers
        - $\sum_{i \in S} \lambda_{i,S}\, n! d_i$
- Choose LWE parameters large enough to accommodate the noise growth

- The issue now is that the parameters of the FHE are dependent on ***n***

# A universal thresholdizer

- ***Setup***: Given a secret $k$ it outputs shares $[k_1 \ldots k_n]$ and a verification key $VK$
- ***Eval***: on input a circuit $C(.,.)$, input $x$ and share $k_i$
  - It outputs a partial evaluation $y_i$
- ***Verify***: On input $C(.,.),x,VK,i,y_i$ it accepts or rejects
- ***Reconstruct***: from $t+1$ accepted partial evaluations $y_i$ it computes $y=C(k,x)$

# Combine TFHE with NIZKs

- ***Setup***:
  - The share of each party is defined as
    - $sk_i$ the share of the TFHE
  - On input the secret $k$ the verification key $VK$ is defined as
    - $FHE(k), COM(sk_i)$
- ***Eval***: on input a circuit $C(.,.)$, input $x, VK$ and share $sk_i$
  - Each party evaluates FHE(C(k,x)) using the homomorphism of FHE
  - Then it produces $y_i$ as
    - the partial decryption under $sk_i$ for the TFHE +
    - a NIZK of correctness wrt $VK,C$
- ***Verify***: checks the NIZK
- ***Reconstruct***: uses the reconstruction procedure of the TFHE

# Applications

If ***k*** is the secret key for a cryptographic scheme and ***C*** is the circuit expressing the cryptographic computation, we obtain 1-round threshold version of any scheme

One interesting application is the “compression” of the non-succinct Shamir-based TFHE we showed earlier

- Our Shamir-based TFHE scheme had parameters growing with n
- We can build a non-succinct universal thresholdizer using this non-succinct TFHE scheme
- But then this UT can be used to thresholdize a succinct FHE
  - Reminds me of the boosting step for FHE

L.De Feo, M.Meyer:Threshold Schemes from Isogeny Assumptions. Public Key Cryptography (2) 2020: 187-212

# Hard Homogenous Spaces

- A set $\mathcal{E}$ endowed with a group action $G$
  - If $g \in G$ and $E \in \mathcal{E}$ there is an operation $g*E=E' \in \mathcal{E}$
  - Hard problems:
    - Given $E,E'$ find $g$ such that $g*E=E'$ (discrete log)
    - Given $E,E'=g*E,F$ find $F'=g*F$ (CDH)
  - The main difference with cyclic groups and discrete log based schemes is that there is no “structure” on the set $\mathcal{E}$
    - Which is the source of the conjecture quantum hardness
  - In isogeny-based instantiations
    - $\mathcal{E}$ is a set of elliptic curves
    - The operation $*$ brings you from one curve to another

# A signature scheme based on HHS

- A rift on Schnorr's. Let E be a "base" curve and assume $G=(Z_q,+)$
- Alice knows $g \in G$ such that $F=g*E$
- To prove this in ZK she runs the following protocol:
  - She chooses $a \in G$ at random and sends $F'=a*E$
  - The verifier sends a bit $b$
  - If $b=0$
    - Alice answers with $c=a$
    - The verifier checks that $c*E=F'$
  - If $b=1$
    - Alice answers with $c=ag^{-1}$
    - The verifier checks that $c*F=F'$
- This proof can be turned into a signature scheme via the Fiat-Shamir heuristic

Let’s talk about isogenies

# A threshold signature scheme based on HHS

- Alice knows $g \in G$: $F=g*E$
  - $\$a \in G$ sends $F'=a*E$
  - The verifier sends a bit $b$
  - If $b=0$
    - Alice answers $c=a$
    - Verifier checks $c*E=F'$
  - If $b=1$
    - Alice answers $c=ag^{-1}$
    - Verifier checks $c*F=F'$

- Assume a dealer has shared $g$ via Shamir among n parties with threshold $t$
- When $t+1$ parties want to sign they map their shares to additive ones $g = g_1 + \dots + g_{t+1}$
- Each party selects a random value $a_i$
  - The computation of $F'$ is performed sequentially
    - The first party computes $F_1=a_1*E$
    - Each next party i computes $F_i=a_i*F_{i-1}$
  - $F'=F_{t+1}$
  - Compute the challenge $b$ via hashing
  - Each party outputs $c_i=a_i-g_i$
  - And $c = c_1 + \dots + c_{t+1}$

***Note the sequential computation***
***You cannot combine two separate isogeny computations***

*Daniele Cozzo, Nigel P. Smart:*
*Sashimi: Cutting up CSI-FiSh Secret Keys to Produce an Actively Secure Distributed Signing Protocol. PQCrypto 2020: 169-186*

# A DKG for isogenies

- Assume a dealer has shared $g$ via Shamir among n parties with threshold $t$
- When $t+1$ parties want to sign they map their shares to additive ones $g = g_1 + \dots + g_{t+1}$
- Each party selects a random value $a_i$
  - The computation of $F'$ is performed sequentially
    - The first party computes $F_1=a_1*E$
    - Each next party i computes $F_i=a_i*F_{i-1}$
  - $F'=F_{t+1}$
  - Compute the challenge $b$ via hashing
  - Each party outputs $c_i=a_i-g_i$
  - And $c = c_1 + \dots + c_{t+1}$

- The generation of the nonce can be used as a DKG
- As in FROST
  - Use the same ZK proof to prove knowledge of the contribution
  - Malicious security with abort

*Ward Beullens, Lucas Disson, Robi Pedersen, Frederik Vercauteren: CSI-RAShi: Distributed Key Generation for CSIDH. PQCrypto 2021: 257-276*

# A Robust DKG for isogenies

- What if we want robustness (guaranteed termination)
  - With honest majority
- Note that in the setting of isogenies there is no equivalent of a Pedersen’s VSS
  - Since it require combining two separate isogeny computations
- It is possible however for each party to do a non-malleable VSS via ZK proofs
  - Providing the non-malleable and recoverable properties of the commitment that we need to make the joint-VSS work
- The combination of the secret keys into a unique public key however remains sequential

# A non-exhaustive list of open problems

- DKG: truly scalable, without quadratic communication
  - Can we use recent advances in SNARKs?
- Better proofs:
  - We have UC proofs for Threshold DSA
  - FROST has a proof for concurrent security but not a full UC proof
- How inefficient is the FHE based UT?
  - FHE has been making great strides. At what point it pays off to build threshold schemes just by calling (a tailored version of) UT?
  - A similar question can be made for MPC
- Can we have threshold isogeny-based schemes without having to pay sequential rounds?