Laconic Cryptography: New Paradigms, Constructions and Directions

Nico Döttling I 06.03.2024
Data-Driven Methods
Advanced Cryptography

Data-Driven Methods
Advanced Cryptography

Data-Driven Methods
Advanced Cryptography

Data-Driven Methods

Communication & Computational Overhead
Motivation: Federated Learning

DB₁

DB₂

DB₃
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
- Partially trained models leak information
Motivation: Federated Learning

- Jointly train models on datasets held by different owners
- Partially trained models leak information
- Conventional MPC leads to exorbitant communication and computation costs
Homomorphic Encryption
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Bob

Alice
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices
**Homomorphic Encryption**

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Bob

Alice

**Laconic Cryptography**

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Laconic Cryptography

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Laconic Cryptography

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Laconic Cryptography

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Laconic Cryptography

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Laconic Cryptography

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Laconic Cryptography

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
**Homomorphic Encryption**

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

**Laconic Cryptography**

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Laconic Cryptography

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Laconic Cryptography

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Homomorphic Encryption

[Gentry STOC’09]
Fully homomorphic encryption using ideal lattices

Laconic Cryptography

[CDGGMP CRYPTO’17]
Laconic Oblivious Transfer and Its Applications
Paradigms
RAM Delegation [CDGGMP17]

$H(DB_1)$

$H(DB_2)$

$H(DB_3)$
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGMP17]

$H(DB_1)$

$P_1$

Garble($P_1$)

$H(DB_2)$

$P_2$

$H(DB_3)$

$P_3$
RAM Delegation [CDGGMP17]

$H(DB_1)$

$H(DB_2)$

$H(DB_3)$

$Garble(P_2 \circ P_1)$

$Garble(P_1)$

$P_1$

$P_2$

$P_3$
RAM Delegation [CDGGMP17]

\[\text{Garble}(P_3 \circ P_2 \circ P_1)\]  

\[H(DB_3)\]

\[P_3\]

\[\text{Garble}(P_2 \circ P_1)\]

\[H(DB_2)\]

\[P_2\]

\[H(DB_1)\]

\[P_1\]
RAM Delegation [CDGGMP17]

\[ H(DB_1) \]
\[ P_1 \]

\[ H(DB_2) \]
\[ P_2 \]

\[ Garble(P_1) \]

\[ Garble(P_2 \circ P_1) \]

\[ Garble(P_3 \circ P_2 \circ P_1) \]

\[ H(DB_3) \]
\[ P_3 \]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGGMP17]
RAM Delegation [CDGGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGGMP17]
RAM Delegation [CDGGMP17]
RAM Delegation [CDGGMP17]
Nothing more than $res$ is revealed, even if parties collude.
Need a primitive to let garbled program access large database!

Nothing more than \(res\) is revealed, even if parties collude
Laconic OT

\[ i, m_0, m_1 \]

DB
Laconic OT

\[ i, m_0, m_1 \]
Laconic OT

\[ i, m_0, m_1 \]

\[ h \]

DB
Laconic OT

\[ i, m_0, m_1 \]

\[ c = Enc(h, i, m_0, m_1) \]
Laconic OT

\[ c = \text{Enc}(h, i, m_0, m_1) \]
Laconic OT

\[ c = Enc(h, i, m_0, m_1) \]

\[ m_{DB_i} = Dec(DB, c) \]
Laconic OT

\[ c = Enc(h, i, m_0, m_1) \]

\[ m_{DB_i} = Dec(DB, c) \]

Learns nothing about \( m_{1-DB_i} \)
Laconic OT

\[ c = Enc(h, i, m_0, m_1) \]

Learns nothing about \( m_{DB_i} \)
Constructing Laconic OT [CDGGMP17]
Hashing: Merkle Trees
Building Block: Hash Encryption

\[ h = H(x) \]
Building Block: Hash Encryption

\[ h = H(x) \]
Building Block: Hash Encryption

\[ h = H(x) \]

\[ i, b, m \]
Building Block: Hash Encryption

\[ h = H(x) \]

\[ i, b, m \]

\[ c = \text{Enc}(h, i, b, m) \]
Building Block: Hash Encryption

\[ i, b, m \]

\[ c = Enc(h, i, b, m) \]
Building Block: Hash Encryption

\[ h = H(x) \]

\[ c = Enc(h, i, b, m) \]

\[ m' = Dec(x, c) \]
Building Block: Hash Encryption

\[ h = H(x) \]

\[ c = Enc(h, i, b, m) \]

\[ m' = Dec(x, c) \]

if \( H(x) = h \) and \( x_i = b \)
Building Block: Hash Encryption

$h = H(x)$

$c = Enc(h, i, b, m)$

$m' = Dec(x, c)$

$m' = m$ if $H(x) = h$ and $x_i = b$
Hash Encryption from DDH (CDH)

$$k = \left( g, g_{1,0} = g^{\alpha_{1,0}}, \ldots, g_{i,0} = g^{\alpha_{i,0}}, \ldots, g_{n,0} = g^{\alpha_{n,0}}, \right)$$

$$g_{1,1} = g^{\alpha_{1,1}}, \ldots, g_{i,1} = g^{\alpha_{i,1}}, \ldots, g_{n,1} = g^{\alpha_{n,1}}$$

$$H(k, x; r) \leftarrow g^r \cdot \prod_{j} g_{j,x_j} = h$$

$$Enc(k, (h, i, b), m) :$$

$$c_1 \leftarrow h^s$$

$$e \leftarrow g_{i,b} \cdot m$$

$$c_0 \leftarrow g^s$$

$$\forall j \neq i : c_{j,0} \leftarrow g_{j,0}^s$$

$$c_{j,1} \leftarrow g_{j,1}^s$$

$$Dec(k, (x, r), c) :$$

$$m \leftarrow e \cdot \frac{c_0^r \cdot \prod_{j \neq i} c_{j,x_j}}{g_{i,x_i}^s \cdot c_0^r \cdot \prod_{j \neq i} c_{j,x_j}}$$

$$= m \cdot \frac{g_{i,b}^s}{g_{i,x_i}^s} = m$$
Building Block: Garbled Circuits
Building Block: Garbled Circuits

\((\tilde{C}, \text{lab}) \leftarrow \text{Garble}(C)\)
Building Block: Garbled Circuits

\((\tilde{C}, \text{lab}) \leftarrow \text{Garble}(C)\)
Building Block: Garbled Circuits

\[(\tilde{C}, \text{lab}) \leftarrow \text{Garble}(C)\]

\[\tilde{x} \leftarrow \text{GarbleInput1}(\text{lab}, x)\]

\[\tilde{y} \leftarrow \text{GarbleInput1}(\text{lab}, y)\]
Building Block: Garbled Circuits

\[(\tilde{C}, \text{lab}) \leftarrow \text{Garble}(C)\]

\[\tilde{x} \leftarrow \text{GarbleInput1}(\text{lab}, x) = \text{lab}_x\]

\[\tilde{y} \leftarrow \text{GarbleInput1}(\text{lab}, y) = \text{lab}_y\]
Building Block: Garbled Circuits

\[ (\tilde{C}, \text{lab}) \leftarrow \text{Garble}(C) \]

\[ \tilde{x} \leftarrow \text{GarbleInput1}(\text{lab}, x) = \text{lab}_x \]

\[ \tilde{y} \leftarrow \text{GarbleInput1}(\text{lab}, y) = \text{lab}_y \]

\[ \text{Eval}(\tilde{C}, \tilde{x}, \tilde{y}) \]
Building Block: Garbled Circuits

\[
(\tilde{C}, \text{lab}) \leftarrow \text{Garble}(C)
\]

\[
\tilde{x} \leftarrow \text{GarbleInput1}(\text{lab}, x) = \text{lab}_x
\]

\[
\tilde{y} \leftarrow \text{GarbleInput1}(\text{lab}, y) = \text{lab}_y
\]

\[
\text{Eval}(\tilde{C}, \tilde{x}, \tilde{y}) = C(x, y)
\]
Building Block: Garbled Circuits

\[(\tilde{C}, \text{lab}) \leftarrow \text{Garble}(C)\]

\[\tilde{x} \leftarrow \text{GarbleInput}(\text{lab}, x)\]

\[\tilde{y} \leftarrow \text{GarbleInput1}(\text{lab}, y) = \text{lab}_y\]

\[(\tilde{C}, \tilde{x}, \tilde{y}) \approx_c \text{GCsim}(C, C(x, y))\]

\[\text{Eval}(\tilde{C}, \tilde{x}, \tilde{y}) = C(x, y)\]
Bootstrapping/Recryption
\[ c = \text{Enc}(h, \text{lab}) \]
Bootstrapping/Recryption

$c = \text{Enc}(h, lab)$

$\text{GarbleInput}(lab, h_1)$
Bootstrapping/Recryption

c = Enc(h, lab)

GarbleInput(lab, h_1)
\[ c = \text{Enc}(h, lab) \]

\[ c' = \text{Enc}(h_1, m) \]
Laconic OT: Encryption

\( i, m_0, m_1 \)
Laconic OT: Encryption
Laconic OT: Encryption
Laconic OT: Encryption

$i, m_0, m_1$
Laconic OT: Encryption

\[ i, m_0, m_1 \]
Laconi OT: Encryption
Laconic OT: Encryption

$i \quad m_0, m_1$

$h \quad C_1$
Laconic OT: Encryption

Outputs $m_0$ or $m_1$ depending on $DB_i$
Core Paradigm: Delegate Work “into the Future”

Outputs $m_0$ or $m_1$ depending on $DB_i$
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption
Laconic OT: Decryption

- Not entire database is needed to decrypt
Laconic OT: Decryption

- Not entire database is needed to decrypt
- Partial witness can be sufficient
Advanced Constructions
Identity-Based Encryption [Sha84]

\[ (mpk, msk) \leftarrow \text{Setup}(1^\lambda) \]
Identity-Based Encryption [Sha84]

\((mpk, msk) \leftarrow \text{Setup}(1^\lambda)\)

mpk

“bob@nist.gov”
Identity-Based Encryption [Sha84]

\[(mpk, msk) \leftarrow \text{Setup}(1^\lambda)\]
\[sk_{id} \leftarrow \text{KeyGen}(msk, id)\]
Identity-Based Encryption [Sha84]

\[(mpk, msk) \leftarrow Setup(1^\lambda)\]
\[sk_{id} \leftarrow KeyGen(msk, id)\]
Identity-Based Encryption [Sha84]

$(mpk, msk) \leftarrow \text{Setup}(1^\lambda)$

$sk_{id} \leftarrow \text{KeyGen}(msk, id)$

$c \leftarrow \text{Encrypt}(mpk, id, m)$

$sk_{bob@nist.gov}$

“bob@nist.gov”
Identity-Based Encryption [Sha84]

$(mpk, msk) \leftarrow \text{Setup}(1^\lambda)$

$sk_{id} \leftarrow \text{KeyGen}(msk, id)$

$c \leftarrow \text{Encrypt}(mpk, id, m)$
Identity-Based Encryption [Sha84]

\[(mpk, msk) \leftarrow Setup(1^\lambda)\]
\[sk_{id} \leftarrow KeyGen(msk, id)\]

\[c \leftarrow Encrypt(mpk, id, m)\]

\[m \leftarrow Decrypt(sk_{id}, c)\]
Identity-Based Encryption [DG17]: Setup

\[ \text{pk}_0 \quad \text{pk}_1 \quad \text{pk}_2 \quad \text{pk}_3 \quad \text{pk}_4 \quad \text{pk}_5 \quad \text{pk}_6 \quad \text{pk}_7 \]
Identity-Based Encryption [DG17]: Setup

- Simpler Setting: Key generator has “pre-generated” a polynomial number of keys and hands them out “on demand”
Identity-Based Encryption [DG17]: Setup

- Simpler Setting: Key generator has “pre-generated” a polynomial number of keys and hands them out “on demand”
- Can be turned into full-blown IBE (with exponentially many identities) by using pseudorandomness and trapdoors (Chameleon encryption)
Identity-Based Encryption [DG17]: KeyGen
Identity-Based Encryption [DG17]: KeyGen
Identity-Based Encryption [DG17]: Encryption
Reverse Delegation as in Laconic OT
Identity-Based Encryption [DG17]: Encryption
Reverse Delegation as in Laconic OT
Identity-Based Encryption [DG17]: Encryption
Reverse Delegation as in Laconic OT

Outputs $Enc(pk_{id}, m)$
Identity-Based Encryption [DG17]: Encryption

Reverse Delegation as in Laconic OT

Outputs $Enc(pk_{id}, m)$
Registration-based Encryption [GHMR18]

\[(pk, sk) \leftarrow \text{KeyGen}()\]
Registration-based Encryption [GHMR18]

- Master Secret Key $msk$ is single point of failure in IBE

$(pk, sk) \leftarrow KeyGen()$
Registration-based Encryption [GHMR18]

- Master Secret Key $msk$ is single point of failure in IBE
- Idea: Replace Key-Authority with Key-Curator

$(pk, sk) \leftarrow \text{KeyGen()}$
Registration-based Encryption [GHMR18]

- Master Secret Key $msk$ is single point of failure in IBE
- Idea: Replace Key-Authority with Key-Curator

$\text{(pk, sk)} \leftarrow \text{KeyGen()}$
Registration-based Encryption [GHMR18]

• Master Secret Key $msk$ is single point of failure in IBE

• Idea: Replace Key-Authority with Key-Curator

$(pk, sk) \leftarrow \text{KeyGen()}$
Registration-based Encryption [GHMR18]

- Master Secret Key $msk$ is single point of failure in IBE
- Idea: Replace Key-Authority with Key-Curator

$(pk, sk) \leftarrow \text{KeyGen}()$
Registration-based Encryption [GHMR18]

- Master Secret Key $msk$ is single point of failure in IBE
- Idea: Replace Key-Authority with Key-Curator

$(pk, sk) \leftarrow \text{KeyGen}()$
Registration-based Encryption [GHMR18]

- Master Secret Key $msk$ is single point of failure in IBE
- Idea: Replace Key-Authority with Key-Curator

\[
\begin{align*}
 pk \leftarrow & \text{KeyGen()} \\
 h, st' \leftarrow & \text{Encrypt}(h, id, m) \\
 id \leftarrow & pk
\end{align*}
\]
Registration-based Encryption [GHMR18]

- Master Secret Key $msk$ is single point of failure in IBE
- Idea: Replace Key-Authority with Key-Curator

$$h, st' \leftarrow \text{KeyGen}()$$

$$h, st \leftarrow \text{KeyGen}()$$

$$c \leftarrow \text{Encrypt}(h, id, m)$$

$$(pk, sk) \leftarrow \text{KeyGen}()$$
Registration-based Encryption [GHMR18]

- Master Secret Key \( msk \) is single point of failure in IBE

- Idea: Replace Key-Authority with Key-Curator

\[
\begin{align*}
&h, st' \leftarrow KeyGen() \\
&c \leftarrow Encrypt(h, id, m) \\
&m \leftarrow Dec(w_{id}, sk, m)
\end{align*}
\]

\( (pk, sk) \leftarrow KeyGen() \)
Laconic Function Evaluation [QWW18]

\[ C \]

\[ x \]
Laconic Function Evaluation [QWW18]
Laconic Function Evaluation [QWW18]
Laconic Function Evaluation [QWW18]

c = Enc(h, x)
Laconic Function Evaluation [QWW18]

\[ c = Enc(h, x) \]
Laconic Function Evaluation [QWW18]

\[ c = Enc(h, x) \]

\[ y = Dec(C, c) = C(x) \]
Laconic Function Evaluation [QWW18]

\[ c = \text{Enc}(h, x) \]

\[ y = \text{Dec}(C, c) = C(x) \]

Learns nothing about \( x \) except \( C(x) \)
Laconic Function Evaluation [QWW18]

\[ c = Enc(h, x) \]

\[ c \]

\[ y = Dec(C, c) = C(x) \]

\textbf{Learns nothing about} \( x \) \textbf{except} \( C(x) \)
Laconic Function Evaluation [QWW18]

\[ c = Enc(h, x) \]

\[ y = Dec(C, c) = C(x) \]

Learns nothing about \( x \) except \( C(x) \)
Laconic Function Evaluation [QWW18]

- Receiver commits to large function instead of database

\[ c = \text{Enc}(h, x) \]

\[ y = \text{Dec}(C, c) = C(x) \]

Learns nothing about \(x\) except \(C(x)\)
Laconic Function Evaluation [QWW18]

- Receiver commits to large function instead of database
- Laconic OT is a special case of LFE: Hashed function is selection function

\[ c = Enc(h, x) \]

\[ y = Dec(C, c) = C(x) \]

Learns nothing about \( x \) except \( C(x) \)
Laconic Function Evaluation [QWW18]

- Receiver commits to large function instead of database
- Laconic OT is a special case of LFE: Hashed function is selection function
- [QWW18] construct LFE from LWE
  
  $c = \text{Enc}(h, x)$
  
  $y = \text{Dec}(C, c) = C(x)$

Learns nothing about $x$ except $C(x)$
Laconic Function Evaluation [QWW18]

- Receiver commits to large function instead of database
- Laconic OT is a special case of LFE: Hashed function is selection function
- [QWW18] construct LFE from LWE
- Size of ciphertext $c$ depends on depth of circuit $C$, but not on size

$c = Enc(h, x)$

$y = Dec(C, c) = C(x)$

Learns nothing about $x$ except $C(x)$
Laconic Function Evaluation [QWW18]

- Receiver commits to large function instead of database
- Laconic OT is a special case of LFE: Hashed function is selection function
- [QWW18] construct LFE from LWE
- Size of ciphertext $c$ depends on depth of circuit $C$, but not on size

$c = Enc(h, x)$

$Learns$ $nothing$ $about$ $x$ $except$ $C(x)$
Laconic Private Set Intersection (LPSI) [ABDGHP21]
Laconic Private Set Intersection (LPSI) [ABDGHP21]
Laconic Private Set Intersection (LPSI) [ABDGHP21]
Laconic Private Set Intersection (LPSI) [ABDGHP21]

\[ c = \text{Enc}(h, x) \]
Laconic Private Set Intersection (LPSI) [ABDGHP21]

\[ c = \text{Enc}(h, x) \]
Laconic Private Set Intersection (LPSI) [ABDGHP21]

\[ c = \text{Enc}(h, x) \]

\[ \text{Dec}(C, c) = x \text{ if } x \in S \text{ otherwise } \bot \]
Laconic Private Set Intersection (LPSI) [ABDGHP21]

\[ c = Enc(h, x) \]

\[ Dec(C, c) = x \text{ if } x \in S \text{ otherwise } \perp \]

Learns nothing about \( x \) if \( x \notin S \)
Laconic Private Set Intersection (LPSI) [ABDGHP21]

\[ c = Enc(h, x) \]

\[ Dec(C, c) = x \text{ if } x \in S \text{ otherwise } \bot \]

Learns nothing about \( x \) if \( x \notin S \)
Laconic Private Set Intersection (LPSI) [ABDGHP21]

\[ c = Enc(h, x) \]

\[ Dec(C, c) = x \text{ if } x \in S \text{ otherwise } \bot \]

Learns nothing about \( x \) if \( x \notin S \)
Laconic Private Set Intersection (LPSI) [ABDGHP21]

- [ABDGHP21] provides an efficient black-box construction of LPSI from number-theoretic assumptions.

\[
s = \begin{cases} \text{Dec}(C, c) & \text{if } x \in S \\ \bot & \text{otherwise} \end{cases}
\]

\[c = \text{Enc}(h, x)\]

Learns nothing about \(x\) if \(x \notin S\).
Application of LPSI: Self-Revealing Encryption

\[ S \]

\( pk, sk \)
Application of LPSI: Self-Revealing Encryption

$h \rightarrow S$
Application of LPSI: Self-Revealing Encryption

\[ c = Enc(h, pk, m) \]
Application of LPSI: Self-Revealing Encryption

\[ c = Enc(h, pk, m) \]
Application of LPSI: Self-Revealing Encryption

\[ c = Enc(h, pk, m) \]

\[ m = Dec(sk, c) \]
Application of LPSI: Self-Revealing Encryption

\[ c = \text{Enc}(h, pk, m) \]

\[ m = \text{Dec}(sk, c) \]
Application of LPSI: Self-Revealing Encryption

$c = Enc(h, pk, m)$

$m = Dec(sk, c)$

$h$

$S$

$pk, sk$

- Can decrypt $m$ if $m \in S$
Application of LPSI: Self-Revealing Encryption

\[ c = Enc(h, pk, m) \]

\[ m = Dec(sk, c) \]

- Can decrypt \( m \) if \( m \in S \)
- Otherwise learns nothing about \( m \)
New Directions
New Directions

• [DKLLMR’22]: First Laconic Crypto Schemes \textit{without} bootstrapping
• Key Insight: Lattice-based re-encryption gadget without intermediate decryption
New Directions

• [DKLLMR’22]: First Laconic Crypto Schemes **without** bootstrapping

• Key Insight: Lattice-based re-encryption gadget without intermediate decryption

• Practically efficient: Prototype Implementation with Single Digit Millisecond runtimes
New Directions

• [DKLLMR’22]: First Laconic Crypto Schemes **without** bootstrapping
• Key Insight: Lattice-based re-encryption gadget without intermediate decryption
• Practically efficient: Prototype Implementation with Single Digit Millisecond runtimes
• Applications: Registration-based Encryption, Laconic Oblivious Transfer, Private Set Intersection
Bootstrapping/Recryption

\[ c = \text{Enc}(h, \text{lab}) \]

\[ \text{GarbleInput(}\text{lab, } h_1) \]
Bootstrapping/Recryption

\[ c = \text{Enc}(h, lab) \]

\[ h_0 \quad h_1 \]

\[ \text{GarbleInput}(lab, h_1) \]

\[ c' = \text{Enc}(h_1, m) \]
Bootstrapping/Recryption

\[ c = \text{Enc}(h, \text{lab}) \]

\[ c' = \text{Enc}(h_1, m) \]
Bootstrapping/Recryption

$c = \text{Enc}(h, lab)$

$GC\left(\text{GarbleInput}(lab, h_1)\right)$

$c' = \text{Enc}(h_1, m)$
New Recryption Algorithm [DKLLMR’22]:
New Recryption Algorithm [DKLLMR’22]:

\[ H(\text{pk}_0, \text{pk}_2) = h = A \cdot \begin{pmatrix} G^{-1}(h_0) \\ G^{-1}(h_1) \end{pmatrix} \]
New Recryption Algorithm [DKLLMR’22]:

\[
H(\text{pk}_0, \text{pk}_2) = h = A \cdot \begin{pmatrix} G^{-1}(h_0) \\ G^{-1}(h_1) \end{pmatrix}
\]

\[
c \approx s \cdot (A + (G||0))
\]

\[
c_1 \approx s \cdot h + \frac{q}{2} m
\]
New Recryption Algorithm [DKLLMR’22]:

\[ H(pk_0, pk_2) = h = A \cdot \begin{pmatrix} G^{-1}(h_0) \\ G^{-1}(h_1) \end{pmatrix} \]

\[ c \approx s \cdot (A + (G||0)) \]

\[ c_1 \approx s \cdot h + \frac{q}{2} m \]
New Recryption Algorithm [DKLLMR’22]:

\[ H(pk_0, pk_2) = h = A \cdot \begin{pmatrix} G^{-1}(h_0) \\ G^{-1}(h_1) \end{pmatrix} \]

\[ c \approx s \cdot (A + (G \parallel 0)) \]

\[ c_1 \approx s \cdot h + \frac{q}{2} m \]

\[ c_1' = c_1 - c \begin{pmatrix} G^{-1}(h_0) \\ G^{-1}(h_1) \end{pmatrix} \approx s \cdot h_0 + \frac{q}{2} m \]
Efficient Private Laconic OT
Efficient Private Laconic OT

• Generally: Laconic OT ciphertext $c$ either reveals query index $i$ or decryption has linear complexity
Efficient Private Laconic OT

• Generally: Laconic OT ciphertext $c$ either reveals query index $i$ or decryption has linear complexity

• [DHMW24]: Private laconic OT (hiding query index $i$) and polylogarithmic decryption complexity
Efficient Private Laconic OT

• Generally: Laconic OT ciphertext $c$ either reveals query index $i$ or decryption has linear complexity

• [DHMW24]: Private laconic OT (hiding query index $i$) and polylogarithmic decryption complexity

• Leverages recent breakthrough on doubly efficient private information retrieval [LMW23]
Private Laconic OT with Preprocessing
• Preprocessing model: Sender and receiver compute and store a "correlations" before e.g. sender gets his input
Private Laconic OT with Preprocessing

- Preprocessing model: Sender and receiver compute and store a “correlations” before e.g. sender gets his input.
- Emerging line of research in sublinear PIR with preprocessing following [CK20].
• Preprocessing model: Sender and receiver compute and store a “correlations” before e.g. sender gets his input

• Emerging line of research in sublinear PIR with preprocessing following [CK20].

• Very efficient, online phase uses only symmetric key crypto
Private Laconic OT with Preprocessing

- Preprocessing model: Sender and receiver compute and store a “correlations” before e.g. sender gets his input
- Emerging line of research in sublinear PIR with preprocessing following [CK20].
- Very efficient, online phase uses only symmetric key crypto
- [BDHL24]: private laconic OT with preprocessing. Also only using symmetric key crypto in online phase
Conclusion
Conclusion

• Laconic Cryptography: Secure computation on **LARGE** data with *small* communication in 2 messages
Conclusion

- Laconic Cryptography: Secure computation on LARGE data with small communication in 2 messages
- Beyond succinct communication: sublinear computation
Conclusion

• Laconic Cryptography: Secure computation on LARGE data with small communication in 2 messages

• Beyond succinct communication: sublinear computation

• Unexpected Applications: IBE, RBE, Self-Revealing Encryption
Conclusion

• Laconic Cryptography: Secure computation on LARGE data with small communication in 2 messages

• Beyond succinct communication: sublinear computation

• Unexpected Applications: IBE, RBE, Self-Revealing Encryption

• Until recently: Mostly theoretical progress
Conclusion

• Laconic Cryptography: Secure computation on **LARGE** data with *small* communication in 2 messages

• Beyond succinct communication: sublinear computation

• Unexpected Applications: IBE, RBE, Self-Revealing Encryption

• Until recently: Mostly theoretical progress

• Now: Breaking the wall to practical usefulness, new ideas such as preprocessing