Laconic Function Evaluation for Turing Machines

Nico Döttling [1] **Phillip Gajland** [2,3] Giulio Malavolta [2]

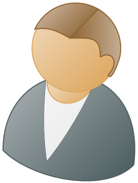


[1] CISPA Helmholtz Center for Information Security
[2] Max Planck Institute for Security and Privacy
[3] Ruhr-University Bochum

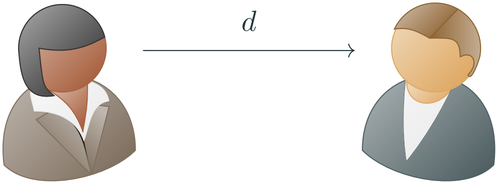
PKC 2023: 26th International Conference on Theory and Practice of Public Key Cryptography, Atlanta, USA

- Laconic function evaluation and applications
- Prior work and problem statement
- Our results and new applications

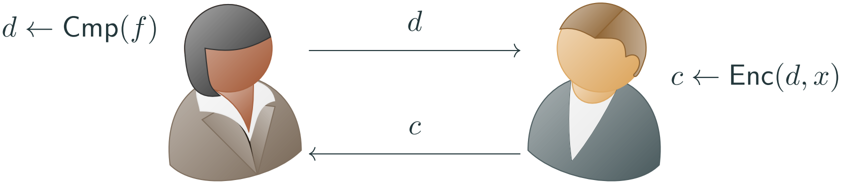
LACONIC FUNCTION EVALUATION

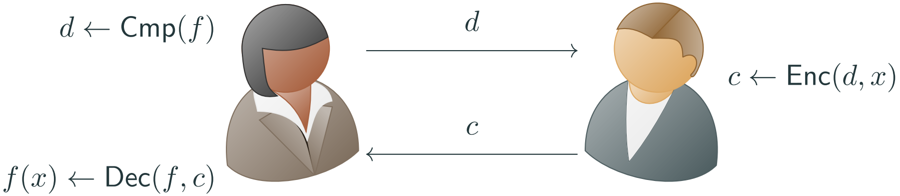


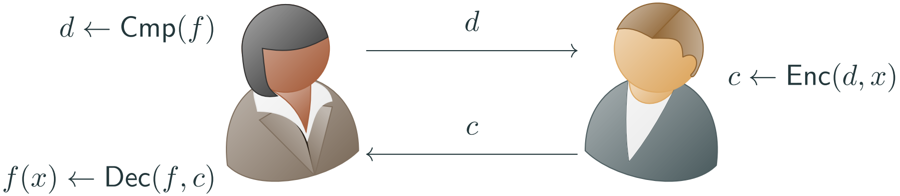
$d \leftarrow \mathsf{Cmp}(f)$



LACONIC FUNCTION EVALUATION [QWW18]

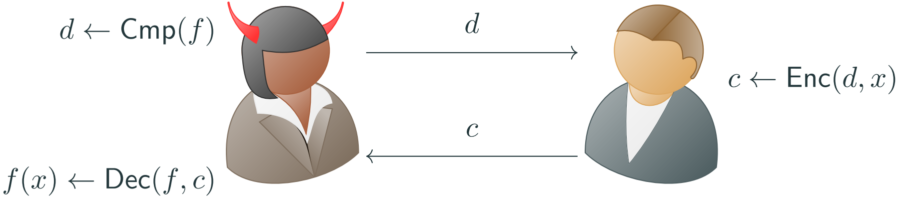



$d \leftarrow \mathsf{Cmp}(f)$
d
$c \leftarrow \mathsf{Enc}(d, x)$
c
$f(x) \leftarrow \mathsf{Dec}(f, c)$



Alice's digest:

- Depends on f
- Is short (Bob has to read it)

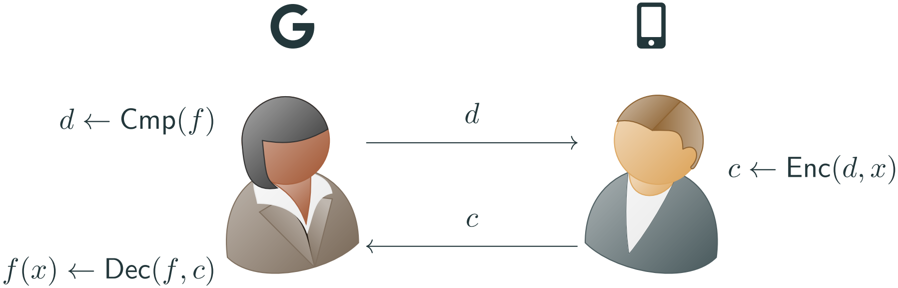


Alice's digest:

- Depends on f
- Is short (Bob has to read it)

Bob's ciphertext:

- Hides x (not $f(x)$)
- Efficient to compute

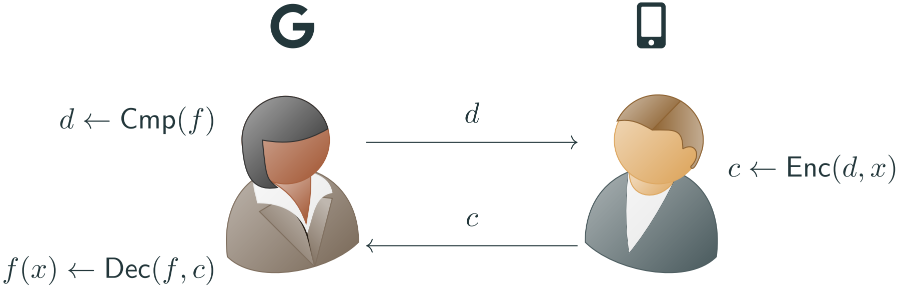


Alice's digest:

- Depends on f
- Is short (Bob has to read it)

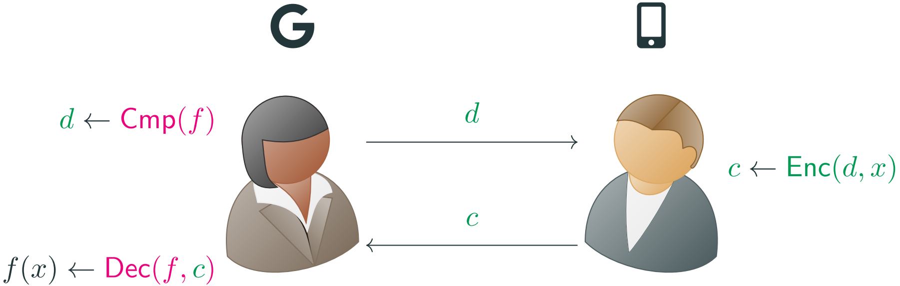
Bob's ciphertext:

- Hides x (not $f(x)$)
- Efficient to compute



Correctness:

- $\mathsf{Dec}(f, c) = f(x)$

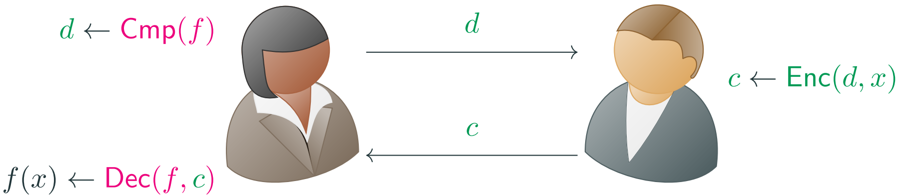


Correctness:

- $\mathsf{Dec}(f, c) = f(x)$

Efficiency:

- Bob's work is small



Correctness:

- $\mathsf{Dec}(f, c) = f(x)$

Efficiency:

- Bob's work is small

Security:

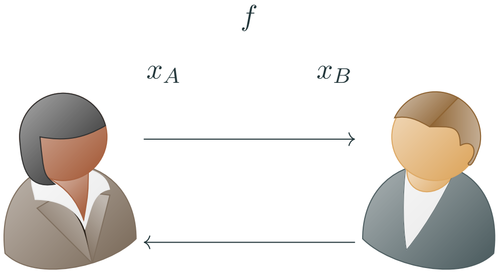
- c hides x - only reveals $f(x)$

APPLICATIONS

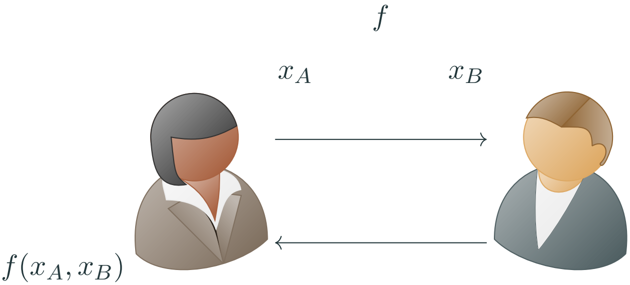
f

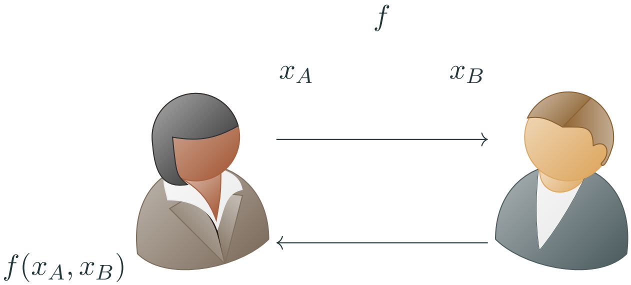
x_A x_B

APPLICATION: SUCCINCT 2-PARTY COMPUTATION



APPLICATION: SUCCINCT 2-PARTY COMPUTATION





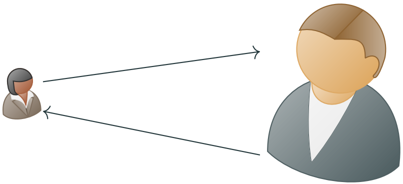
Security:

- Alice only learns $f(x_A, x_B)$ and Bob doesn't learn anything

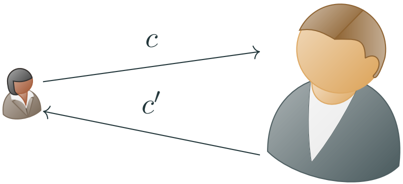
Efficiency:

- Communication + computation $\ll$ computing $f(x_A, x_B)$

SUCCINCT 2-PC: ALICE-OPTIMISED PROTOCOL VIA FHE



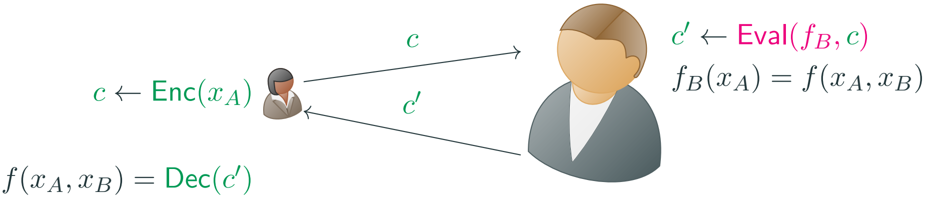
$c \leftarrow \mathsf{Enc}(x_A)$



$c' \leftarrow \mathsf{Eval}(f_B, c)$

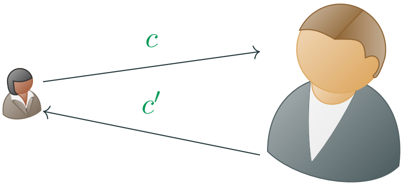
$f_B(x_A) = f(x_A, x_B)$

$f(x_A, x_B) = \mathsf{Dec}(c')$


$c \leftarrow \mathsf{Enc}(x_A)$
c
c'
$c' \leftarrow \mathsf{Eval}(f_B, c)$
$f_B(x_A) = f(x_A, x_B)$
$f(x_A, x_B) = \mathsf{Dec}(c')$

SUCCINCT 2-PC: BOB-OPTIMISED PROTOCOL VIA FUNCTION HIDING LFE

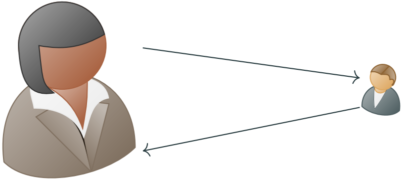
$c \leftarrow \mathsf{Enc}(x_A)$

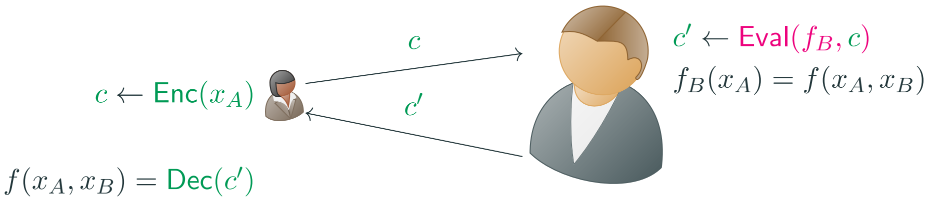


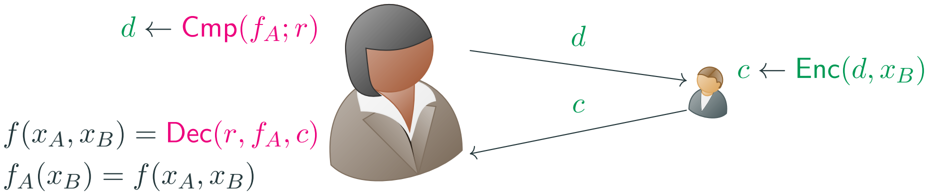
$c' \leftarrow \mathsf{Eval}(f_B, c)$

$f_B(x_A) = f(x_A, x_B)$

$f(x_A, x_B) = \mathsf{Dec}(c')$




$c \leftarrow \mathsf{Enc}(x_A)$
c
c'
$c' \leftarrow \mathsf{Eval}(f_B, c)$
$f_B(x_A) = f(x_A, x_B)$
$f(x_A, x_B) = \mathsf{Dec}(c')$


$d \leftarrow \mathsf{Cmp}(f_A; r)$
d
c
$c \leftarrow \mathsf{Enc}(d, x_B)$
$f(x_A, x_B) = \mathsf{Dec}(r, f_A, c)$
$f_A(x_B) = f(x_A, x_B)$

Direct applications:

- MPC with low online computation [QWW18]
- Adaptively secure MPC with sublinear communication complexity [CsW19]
- Compact NIZKs from various assumptions [KNYY19]

Techniques used for laconic cryptography have led to:

- Laconic Conditional Disclosure of Secrets [DGGM19]
- Single-server private-information retrieval from weaker assumptions [DGI+19]
- Identity-based encryption from weaker assumptions [DG17a, DG17b, BLSV18]
- Two-round MPC from minimal assumptions [GS17, GS18c, BL18]
- Adaptively secure garbled circuits from weaker assumptions [GS18a]
- Trapdoor functions from weaker assumptions [GH18]

Direct applications:

- MPC with low online computation [QWW18]
- Adaptively secure MPC with sublinear communication complexity [CsW19]
- Compact NIZKs from various assumptions [KNYY19]

Techniques used for laconic cryptography have led to:

- Laconic Conditional Disclosure of Secrets [DGGM19]
- Single-server private-information retrieval from weaker assumptions [DGI+19]
- Identity-based encryption from weaker assumptions [DG17a, DG17b, BLSV18]
- Two-round MPC from minimal assumptions [GS17, GS18c, BL18]
- Adaptively secure garbled circuits from weaker assumptions [GS18a]
- Trapdoor functions from weaker assumptions [GH18]

Direct applications:

- MPC with low online computation [QWW18]
- Adaptively secure MPC with sublinear communication complexity [CsW19]
- Compact NIZKs from various assumptions [KNYY19]

Techniques used for laconic cryptography have led to:

- Laconic Conditional Disclosure of Secrets [DGGM19]
- Single-server private-information retrieval from weaker assumptions [DGI+19]
- Identity-based encryption from weaker assumptions [DG17a, DG17b, BLSV18]
- Two-round MPC from minimal assumptions [GS17, GS18c, BL18]
- Adaptively secure garbled circuits from weaker assumptions [GS18a]
- Trapdoor functions from weaker assumptions [GH18]

Direct applications:

- MPC with low online computation [QWW18]
- Adaptively secure MPC with sublinear communication complexity [CsW19]
- Compact NIZKs from various assumptions [KNYY19]

Techniques used for laconic cryptography have led to:

- Laconic Conditional Disclosure of Secrets [DGGM19]
- Single-server private-information retrieval from weaker assumptions [DGI+19]
- Identity-based encryption from weaker assumptions [DG17a, DG17b, BLSV18]
- Two-round MPC from minimal assumptions [GS17, GS18c, BL18]
- Adaptively secure garbled circuits from weaker assumptions [GS18a]
- Trapdoor functions from weaker assumptions [GH18]

Direct applications:

- MPC with low online computation [QWW18]
- Adaptively secure MPC with sublinear communication complexity [CsW19]
- Compact NIZKs from various assumptions [KNYY19]

Techniques used for laconic cryptography have led to:

- Laconic Conditional Disclosure of Secrets [DGGM19]
- Single-server private-information retrieval from weaker assumptions [DGI$^+$19]
- Identity-based encryption from weaker assumptions [DG17a, DG17b, BLSV18]
- Two-round MPC from minimal assumptions [GS17, GS18c, BL18]
- Adaptively secure garbled circuits from weaker assumptions [GS18a]
- Trapdoor functions from weaker assumptions [GH18]

PRIOR WORK

[**QWW18**] First construction of LFE from LWE

[**PCFT20**] Generalisations and construction from iO

[**Ros22**] Stronger security for specific class of circuits

Problem: Ciphertext (and runtime of Enc) grow polynomially with depth of circuit

Think of: "Leveled" FHE vs. "pure" FHE

[**QWW18**] First construction of LFE from LWE

[**PCFT20**] Generalisations and construction from iO

[**Ros22**] Stronger security for specific class of circuits

Problem: Ciphertext (and runtime of Enc) grow polynomially with depth of circuit

Think of: "Leveled" FHE vs. "pure" FHE

[**QWW18**] First construction of LFE from LWE

[**PCFT20**] Generalisations and construction from iO

[**Ros22**] Stronger security for specific class of circuits

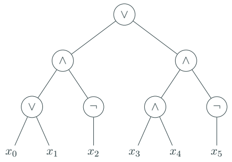
Problem: Ciphertext (and runtime of Enc) grow polynomially with depth of circuit

Think of: “Leveled” FHE vs. “pure” FHE

[**QWW18**] First construction of LFE from LWE

[**PCFT20**] Generalisations and construction from iO

[**Ros22**] Stronger security for specific class of circuits



Problem: Ciphertext (and runtime of Enc) grow polynomially with depth of circuit

Think of: “Leveled” FHE vs. “pure” FHE

"Is it possible to construct LFE where the size of the ciphertext and the runtime of Enc ***are independent of the circuit depth?"***

CONTRIBUTIONS

Theorem (Informal)

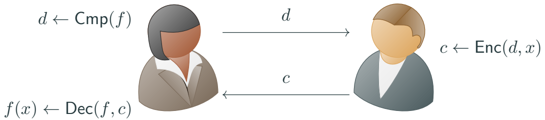
$$\boldsymbol{iO} + \boldsymbol{ULOT} \implies \boldsymbol{LFE}$$

- $|d| = \mathsf{poly}(\lambda)$
- $\mathsf{Enc} = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$
- $|c| = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$

Theorem (Informal)

$$\boldsymbol{i\mathcal{O}} + \boldsymbol{ULOT} \Longrightarrow \boldsymbol{LFE}$$

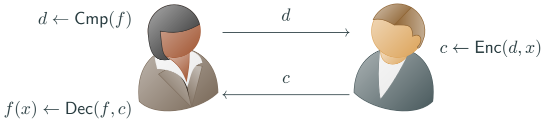
- $|d| = \mathsf{poly}(\lambda)$
- $\mathsf{Enc} = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$
- $|c| = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$



Theorem (Informal)

$$\boldsymbol{iO} + \boldsymbol{ULOT} \implies \boldsymbol{LFE}$$

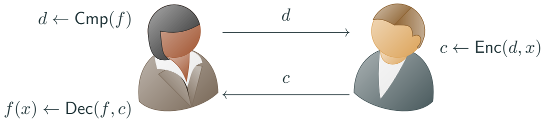
- $|d| = \mathsf{poly}(\lambda)$
- $\mathsf{Enc} = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$
- $|c| = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$



Theorem (Informal)

$$\boldsymbol{iO} + \boldsymbol{ULOT} \implies \boldsymbol{LFE}$$

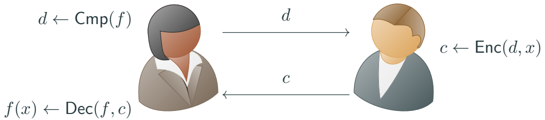
- $|d| = \mathsf{poly}(\lambda)$
- $\mathsf{Enc} = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$
- $|c| = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$



Theorem (Informal)

$$\boldsymbol{iO} + \boldsymbol{ULOT} \implies \boldsymbol{LFE}$$

- $|d| = \mathsf{poly}(\lambda)$
- $\mathsf{Enc} = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$
- $|c| = \mathcal{O}(|x|) \cdot \mathsf{poly}(\lambda)$



CONSTRUCTION

Indistinguishability Obfuscation

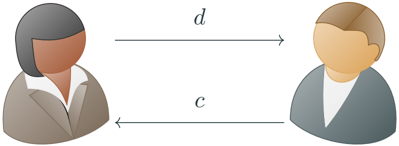
$$C_0(x) \quad = \quad C_1(x) \quad \Longrightarrow \quad i\mathcal{O}(C_0)(x) \quad \approx \quad i\mathcal{O}(C_1)(x)$$

Indistinguishability Obfuscation

$$C_0(x) = C_1(x) \implies i\mathcal{O}(C_0)(x) \approx i\mathcal{O}(C_1)(x)$$

Updatable Laconic Oblivious Transfer

$d \leftarrow \mathsf{ULOT.Hash}(D)$

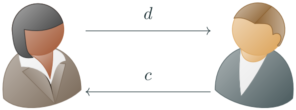


$c \leftarrow \mathsf{ULOT.Send}\,(d, L, \{m_0, m_1\})$

$m_{D[L]} \leftarrow \mathsf{ULOT.Receive}(L, c)$

$d \leftarrow \mathsf{Cmp}(f)$

- $d \leftarrow \mathsf{ULOT.Hash}(f)$



$f(x) \leftarrow \mathsf{Dec}(f, c)$

- **while** $i \neq T$
- Run obfuscated Step Circuit at i
- $\mathsf{ULOT.Receive}$ returns the inputs for $i\mathcal{O}(\mathsf{SC}_{i+1})$
- $f(x) \leftarrow$ last output from Step Circuit

$c \leftarrow \mathsf{Enc}(d, x)$

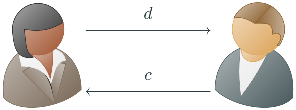
- Obfuscate Step Circuit SC as

  $i\mathcal{O}(\mathsf{SC}) \leftarrow$ Step Circuit SC
  - Perform 1 step of the computation
  - Write outputs to DB using $\mathsf{ULOT.Send}$
- Encrypt input x
- $c \leftarrow (i\mathcal{O}(\mathsf{SC}), \mathsf{Sym.Enc}(x))$

- The proof follows from a pebbling strategy similar to [GS18b]

$d \leftarrow \mathsf{Cmp}(f)$

1 : $d \leftarrow \mathsf{ULOT.Hash}(f)$



$f(x) \leftarrow \mathsf{Dec}(f, c)$

1 : **while** $i \neq T$

2 : Run obfuscated Step Circuit at i

3 : $\mathsf{ULOT.Receive}$ returns the inputs for $i\mathcal{O}(\mathsf{SC}_{i+1})$

4 : $f(x) \leftarrow$ last output from Step Circuit

$c \leftarrow \mathsf{Enc}(d, x)$

1 : Obfuscate Step Circuit SC as

$i\mathcal{O}(\mathsf{SC}) \leftarrow$ Step Circuit SC

1 : Perform 1 step of the computation

2 : Write outputs to DB using ULOT.Send

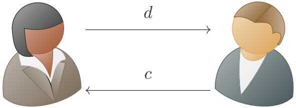
2 : Encrypt input x

3 : $c \leftarrow (i\mathcal{O}(\mathsf{SC}), \mathsf{Sym.Enc}(x))$

- The proof follows from a pebbling strategy similar to [GS18b]

$d \leftarrow \mathsf{Cmp}(f)$

1 : $d \leftarrow \mathsf{ULOT.Hash}(f)$



$f(x) \leftarrow \mathsf{Dec}(f, c)$

1 : **while** $i \neq T$
2 : Run obfuscated Step Circuit at i
3 : $\mathsf{ULOT.Receive}$ returns the inputs for $i\mathcal{O}(\mathsf{SC}_{i+1})$
4 : $f(x) \leftarrow$ last output from Step Circuit

$c \leftarrow \mathsf{Enc}(d, x)$

1 : Obfuscate Step Circuit SC as

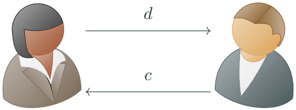
$i\mathcal{O}(\mathsf{SC}) \leftarrow$ Step Circuit SC
- 1 : Perform 1 step of the computation
- 2 : Write outputs to DB using $\mathsf{ULOT.Send}$

2 : Encrypt input x
3 : $c \leftarrow (i\mathcal{O}(\mathsf{SC}), \mathsf{Sym.Enc}(x))$

- The proof follows from a pebbling strategy similar to [GS18b]

$d \leftarrow \mathsf{Cmp}(f)$

1 : $d \leftarrow \mathsf{ULOT.Hash}(f)$



$f(x) \leftarrow \mathsf{Dec}(f, c)$

1 : **while** $i \neq T$

2 : Run obfuscated Step Circuit at i

3 : $\mathsf{ULOT.Receive}$ returns the inputs for $i\mathcal{O}(\mathsf{SC}_{i+1})$

4 : $f(x) \leftarrow$ last output from Step Circuit

$c \leftarrow \mathsf{Enc}(d, x)$

1 : Obfuscate Step Circuit SC as

$i\mathcal{O}(\mathsf{SC}) \leftarrow$ Step Circuit SC

  1 : Perform 1 step of the computation

  2 : Write outputs to DB using $\mathsf{ULOT.Send}$

2 : Encrypt input x

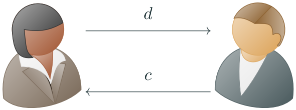
3 : $c \leftarrow (i\mathcal{O}(\mathsf{SC}), \mathsf{Sym.Enc}(x))$

- The proof follows from a pebbling strategy similar to [GS18b]

CONSTRUCTION: HIGH-LEVEL OVERVIEW

$d \leftarrow \mathsf{Cmp}(f)$

$d \leftarrow \mathsf{ULOT.Hash}(f)$



$f(x) \leftarrow \mathsf{Dec}(f, c)$

---

while $i \neq T$
  Run obfuscated Step Circuit at i
  $\mathsf{ULOT.Receive}$ returns the inputs for $i\mathcal{O}(\mathsf{SC}_{i+1})$
$f(x) \leftarrow$ last output from Step Circuit

$c \leftarrow \mathsf{Enc}(d, x)$

Obfuscate Step Circuit SC as
$i\mathcal{O}(\mathsf{SC}) \leftarrow$ Step Circuit SC

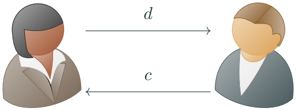
> Perform 1 step of the computation
> Write outputs to DB using $\mathsf{ULOT.Send}$

Encrypt input x
$c \leftarrow (i\mathcal{O}(\mathsf{SC}), \mathsf{Sym.Enc}(x))$

- The proof follows from a pebbling strategy similar to [GS18b]

$d \leftarrow \mathsf{Cmp}(f)$

- $d \leftarrow \mathsf{ULOT.Hash}(f)$



$f(x) \leftarrow \mathsf{Dec}(f, c)$

- **while** $i \neq T$
  - Run obfuscated Step Circuit at i
  - $\mathsf{ULOT.Receive}$ returns the inputs for $i\mathcal{O}(\mathsf{SC}_{i+1})$
- $f(x) \leftarrow$ last output from Step Circuit

$c \leftarrow \mathsf{Enc}(d, x)$

- Obfuscate Step Circuit SC as
  $i\mathcal{O}(\mathsf{SC}) \leftarrow$ Step Circuit SC
  - Perform 1 step of the computation
  - Write outputs to DB using $\mathsf{ULOT.Send}$
- Encrypt input x
- $c \leftarrow (i\mathcal{O}(\mathsf{SC}), \mathsf{Sym.Enc}(x))$

- The proof follows from a pebbling strategy similar to [GS18b]

- Using the work of [KNYY19] we get NIZKs with optimal prover complexity
- Witness encryption [GGSW13] where $|c|$ depends only on $|w|$ and λ
- First ABE for Turing machines [GKP+13] from falsifiable assumptions

- Using the work of [KNYY19] we get NIZKs with optimal prover complexity
- Witness encryption [GGSW13] where $|c|$ depends only on $|w|$ and λ
- First ABE for Turing machines [GKP+13] from falsifiable assumptions

- Using the work of [KNYY19] we get NIZKs with optimal prover complexity
- Witness encryption [GGSW13] where $|c|$ depends only on $|w|$ and λ
- First ABE for Turing machines [GKP+13] from falsifiable assumptions

CONCLUSION

Contributions:

- Asymptotically optimal Laconic Function Evaluation for Turing machines
- New applications:
  - NIZK with optimal prover complexity
  - WE and ABE for Turing machines from falsifiable assumptions

ia.cr/2023/502

phillip.gajland@{mpi-sp.org,rub.de}
p4i11ip

PKC 2023

Contributions:

- Asymptotically optimal Laconic Function Evaluation for Turing machines
- New applications:
  - NIZK with optimal prover complexity
  - WE and ABE for Turing machines from falsifiable assumptions

ia.cr/2023/502

phillip.gajland@{mpi-sp.org,rub.de}
p4i11ip

PKC 2023

[BL18] Fabrice Benhamouda and Huijia Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part II*, volume 10821 of *Lecture Notes in Computer Science*, pages 500–532, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[BLSV18] Zvika Brakerski, Alex Lombardi, Gil Segev, and Vinod Vaikuntanathan. Anonymous IBE, leakage resilience and circular security from new assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 535–564, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[CsW19] Ran Cohen, abhi shelat, and Daniel Wichs. Adaptively secure MPC with sublinear communication complexity. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 30–60, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[DG17a] Nico Döttling and Sanjam Garg. From selective IBE to full IBE and selective HIBE. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017: 15th Theory of Cryptography Conference, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 372–408, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany.

[DG17b] Nico Döttling and Sanjam Garg. Identity-based encryption from the Diffie-Hellman assumption. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 537–569, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[DGGM19] Nico Döttling, Sanjam Garg, Vipul Goyal, and Giulio Malavolta. Laconic conditional disclosure of secrets and applications. In David Zuckerman, editor, *60th Annual Symposium on Foundations of Computer Science*, pages 661–685, Baltimore, MD, USA, November 9–12, 2019. IEEE Computer Society Press.

[DGI+19] Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 3–32, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 467–476, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[GH18] Sanjam Garg and Mohammad Hajiabadi. Trapdoor functions from the computational Diffie-Hellman assumption. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 362–391, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany.

[GKP+13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. How to run turing machines on encrypted data. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 536–553, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[GS17] Sanjam Garg and Akshayaram Srinivasan. Garbled protocols and two-round MPC from bilinear maps. In Chris Umans, editor, *58th Annual Symposium on Foundations of Computer Science*, pages 588–599, Berkeley, CA, USA, October 15–17, 2017. IEEE Computer Society Press.

[GS18a] Sanjam Garg and Akshayaram Srinivasan. Adaptively secure garbling with near optimal online complexity. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part II*, volume 10821 of *Lecture Notes in Computer Science*, pages 535–565, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[GS18b] Sanjam Garg and Akshayaram Srinivasan. A simple construction of iO for turing machines. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018: 16th Theory of Cryptography Conference, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 425–454, Panaji, India, November 11–14, 2018. Springer, Heidelberg, Germany.

[GS18c] Sanjam Garg and Akshayaram Srinivasan. Two-round multiparty secure computation from minimal assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part II*, volume 10821 of *Lecture Notes in Computer Science*, pages 468–499, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[KNYY19] Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Exploring constructions of compact NIZKs from various assumptions. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 639–669, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[PCFT20] Bo Pang, Long Chen, Xiong Fan, and Qiang Tang. Multi-input laconic function evaluation. In Joseph K. Liu and Hui Cui, editors, *ACISP 20: 25th Australasian Conference on Information Security and Privacy*, volume 12248 of *Lecture Notes in Computer Science*, pages 369–388, Perth, WA, Australia, November 30 – December 2, 2020. Springer, Heidelberg, Germany.

[QWW18] Willy Quach, Hoeteck Wee, and Daniel Wichs. Laconic function evaluation and applications. In Mikkel Thorup, editor, *59th Annual Symposium on Foundations of Computer Science*, pages 859–870, Paris, France, October 7–9, 2018. IEEE Computer Society Press.

[QWW20] Willy Quach, Hoeteck Wee, and Daniel Wichs. Laconic function evaluation. https://www.youtube.com/watch?v=sZ7US6yKD-I, 2020.

[Ros22] Razvan Rosie. Adaptively secure laconic function evaluation for NC^1. In Steven D. Galbraith, editor, *Topics in Cryptology – CT-RSA 2022*, volume 13161 of *Lecture Notes in Computer Science*, pages 427–450, Virtual Event, March 1–2, 2022. Springer, Heidelberg, Germany.