Obfuscating Finite Automata

Steven D. Galbraith and Lukas Zobernig

The University of Auckland

SAC 2020

Introduction

- **General purpose** program obfuscation is hard.
- Generic VBB obfuscation is **impossible**.
- Indistinguishability obfuscation seems **infeasible** (at least for now).
- Should we just give up and stop caring about obfuscation altogether?
- Consider **special purpose obfuscation**: Bite-sized problems which we can solve.
- Buzzwords: **Point functions, hyperplane membership**, **conjunctions**, **pattern matching with wildcards**, **fuzzy Hamming distance matching**, **compute and compare programs**, etc.

A Few Open Problems

We know how to obfuscate:

- Point functions,
- which are generalised by conjunctions.
- Fuzzy Hamming distance matching, yielding secure sketches, fuzzy extractors.

Open Obfuscation Problems

- **Finite automata**,
- **regular expressions**,
- **substring matching**.

A Few Open Problems

- Genise et al. [2] gave an **interactive** solution for finite automata,
- they mention *antivirus signatures* as an application.
- The idea is to use *fully homomorphic encryption* (FHE) to evaluate a secret automaton on a public input.
- This produces an encrypted *state vector*, which a server can decrypt and then answer about a virus infection.
- Can learn an automaton from accept/reject behaviour (**we will fix this**).
- Desmoulins et al. [1] describe a flip-side scheme, matching public automata on encrypted inputs.

A Common Theme

All of the aforementioned problems were **evasive**!

Definition (Evasive Program Collection)

Let $\mathcal{P} = \{\mathcal{P}_n\}_{n \in \mathbb{N}}$ be a collection of polynomial-size programs such that every $P \in \mathcal{P}_n$ is a program $P : \{0,1\}^n \to \{0,1\}$. The collection $\mathcal{P}$ is called **evasive** if there exists a negligible function ϵ such that for every $n \in \mathbb{N}$ and for every $y \in \{0,1\}^n$:

$$\Pr_{P \leftarrow \mathcal{P}_n}[P(y) = 1] \leq \epsilon(n).$$

Focus on evasive problems for now, as many of those have special purpose obfuscators.

Evasive Finite Automata

In the same spirit, we shall consider **evasive finite automata**.

Definition (Evasive Finite Automata Collection)

Let $\{\mathcal{M}_r\}_{r\in\mathbb{N}}$ be a collection of finite automata such that every automaton in $\mathcal{M}_r$ has r states. The collection is called *evasive* if there exists a negligible function ϵ such that for every $r \in \mathbb{N}$ and for every polynomial-size input $y \in \Sigma^*$:

$$\Pr_{M\leftarrow\mathcal{M}_r}[M(y) = 1] \leq \epsilon(r).$$

Observations

- Limit to polynomial size inputs $y \in \Sigma^*$ or else y could be a string that contains all possible substrings of a certain length.
- Can possibly learn structure of non-evasive finite automata from input/accept/reject behaviour.

The Key Idea(s)

- Represent a **deterministic** finite automaton (DFA) by **transition matrices**.
- This ensures that states are represented by **canonical basis vectors**.
- Use a **matrix graded encoding scheme** to encrypt the transition matrices.
- This allows us to evaluate the hidden DFA on plaintext input by multiplying encoded matrices.
- **But how do we get a plaintext answer?**

Limited Zero Testing

The matrix encoding scheme needs to support limited zero testing: In our case, decide whether the **last coordinate** of an encrypted vector is 0.

Transition Matrices

- Every DFA with $r \in \mathbb{N}$ states on input symbols $\sigma \in \Sigma$ can be represented by $|\Sigma|$-many *transition* matrices $M_\sigma \in \{0,1\}^{r \times r}$, acting on a state vector $v \in \{0,1\}^r$.
- We can choose the matrices such that they have the following form:

$$M_{\sigma_1} = \begin{pmatrix} * & * \\ 0 & 0 \end{pmatrix}, \dots, M_{\sigma_{m-1}} = \begin{pmatrix} * & * \\ 0 & 0 \end{pmatrix}, M_{\sigma_m} = \begin{pmatrix} * & \cdots & * & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ * & \cdots & * & 0 & 0 \\ 0 & \cdots & 0 & 1 & 1 \end{pmatrix},$$

  where $\Sigma = \{\sigma_1, \dots, \sigma_m\}$.
- We identify the r-th canonical basis vector e_r with the *accepting state* r.
- The limited zero-test can detect this vector.

HAO15 With Limited Zero Testing

We use the HAO15 matrix FHE scheme by Hiromasa et al. [3] over the ring $\mathbb{Z}/q\mathbb{Z}$.

Matrix & Vector Encoding

Given matrix $M \in \{0,1\}^{r \times r}$ or vector $v \in \{0,1\}^r$, HAO15 encodings $C \in (\mathbb{Z}/q\mathbb{Z})^{N \times N}$ or $c \in (\mathbb{Z}/q\mathbb{Z})^N$, respectively, satisfy:

$$SC = MSG + E,$$
$$Sc = \beta v + e,$$

for *gadget matrix* G, *secret* matrix S, noise E,e, and scaling constant β.

Homomorphisms

Multiply encoded matrices C_1, C_2 via $C_1 \odot C_2 := C_1 G^{-1}(C_2)$ and apply encoded matrices to encoded vectors via $C \odot c := CG^{-1}(c)$.

HAO15 With Limited Zero Testing

Limited Zero Testing

Let s_r by the last row of the secret S. Then the last entry v_r of v is equal to

$$v_r = \left\lceil \frac{s_r \cdot c \mod q}{\beta} \right\rfloor .$$

Maximal Grading

- Every multiplication of encoded objects **accumulates noise**.
- We have a **maximal grading** κ (number of possible multiplications):

$$\kappa \leq \frac{q}{4\sqrt{n}(n+r)\lceil \log(q) \rceil}.$$

Correctness

Given an input word $w \in \Sigma^*$, we compute

$$c_w = \left(\bigodot_{i=|w|}^{1} C_{w_i} \right) G^{-1}(c).$$

This corresponds to the plaintext computation

$$t = \left(\prod_{i=|w|}^{1} M_{w_i} \right) e_1.$$

The automaton accepts the input if $t = e_r$. We see that c_w is an encoding of t such that $Sc_w = \beta t + e$ for some noise vector e. Given only s_r, we have

$$\begin{pmatrix} 0_{(r-1)\times(n+r)} \\ \hline s_r \end{pmatrix} c_w = \beta \begin{pmatrix} 0_{r-1} \\ \hline t_r \end{pmatrix} + \begin{pmatrix} 0_{r-1} \\ \hline e' \end{pmatrix}.$$

Security

DFA Security for HAO15

We assume that given encodings of two matrices $M, M' \in \{0,1\}^{r\times r}$ which differ by at most one entry in some row but not the last row

$$\begin{aligned} SC &= MSG + E, \\ SC' &= M'SG + E', \end{aligned}$$

the following two distributions are computationally indistinguishable:

$$(s_r, (C_\sigma)_{\sigma\in\Sigma}, \alpha) \overset{c}{\approx} (s_r, (C'_\sigma)_{\sigma\in\Sigma}, \alpha),$$

where s_r is the last row of the secret key S, and α is auxiliary information.

Security

Assuming HAO15 is DFA secure, we show that our obfuscator for evasive DFAs is a **virtual black box** (VBB) obfuscator.

Theorem

Let $\mathcal{D} = \{D_\lambda\}_{\lambda \in \mathbb{N}}$ be an efficiently samplable DFA evasive distribution with auxiliary information. Assume that for every $\lambda \in \mathbb{N}$ it holds that HAO15 with security parameter λ is DFA secure for D_λ. Then the obfuscator $\mathcal{O}$ is a VBB obfuscator for $\mathcal{D}$.

Conclusion

- We started from the HAO15 matrix FHE scheme,
- which we extended by a **limited zero-testing** primitive.
- We represent finite automata by transition matrices, these are encoded using the HAO15 scheme.
- We can evaluate the hidden automaton on **plaintext** input by multiplying encoded matrices.
- We needed to restrict to evasive DFAs, otherwise black-box access suffices to learn the DFA structure.
- Finally, we obtain a **VBB obfuscator** for **evasive DFAs**.
- This solves the problem of **obfuscated substring matching**.

References

Nicolas Desmoulins, Pierre-Alain Fouque, Cristina Onete, and Olivier Sanders. "Pattern Matching on Encrypted Streams". In: *ASIACRYPT 2018*. Springer, 2018, pp. 121–148.

Nicholas Genise, Craig Gentry, Shai Halevi, Baiyu Li, and Daniele Micciancio. "Homomorphic Encryption for Finite Automata". In: *ASIACRYPT 2019*. Springer, 2019, pp. 473–502.

Ryo Hiromasa, Masayuki Abe, and Tatsuaki Okamoto. "Packing Messages and Optimizing Bootstrapping in GSW-FHE". In: *PKC 2015*. Springer, 2015, pp. 699–715.