AFRL-AFOSR-VA-TR-2019-0091

Foundations and Applications of Program Obfuscation

Rafael Pass
CORNELL UNIVERSITY
373 PINE TREE RD
ITHACA, NY 14850-2820

04/14/2019
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/RTA2

Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
24-10-2018	Final Report	15-8-2015 - 14-8-2018

4. TITLE AND SUBTITLE
AFOSR: Foundations and Applications of Program Obfuscation

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-15-1-0262

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Rafael Pass
Department of Computer Science
Cornell NYC Tech
New York, NY 10044

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Cornell University
2 West Loop Rd, NY, NY 10044

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research
875 Randolph Street
Suite 325 Room 3112
Arlington , VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution A - Approved for Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The goal of program obfuscation is to “scramble” a computer program, hiding its implementation details (making it hard to “reverse-engineer”), wh ile preserving the func- tionality (i.e, input/output behavior) of the program. Program obfuscation is widely used in practice, enabling the distributio n of code for performing large-scale distributed computations (e.g., in applications such as Skype), while ensuring that users employ the service in t he intended way. But these usages largely rely on ad-hoc heuristics that often get broken. Our research has developed new techniques towards prov ably secure obfuscation. More generally, we have developed new techniques for proving the security of large-scale distributed services, including th e first provably secure constructions of large-scale MPC, proving the security of Nakamoto's blockchain protocol, and developing the first 2-round non-malleable commitment (a crucial component for round efficient MPC)

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(Include area code)*

Reset

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18
Adobe Professional 7.0

INSTRUCTIONS FOR COMPLETING SF 298

1. REPORT DATE. Full publication date, including day, month, if available. Must cite at least the year and be Year 2000 compliant, e.g. 30-06-1998; xx-06-1998; xx-xx-1998.

2. REPORT TYPE. State the type of report, such as final, technical, interim, memorandum, master's thesis, progress, quarterly, research, special, group study, etc.

3. DATES COVERED. Indicate the time during which the work was performed and the report was written, e.g., Jun 1997 - Jun 1998; 1-10 Jun 1996; May - Nov 1998; Nov 1998.

4. TITLE. Enter title and subtitle with volume number and part number, if applicable. On classified documents, enter the title classification in parentheses.

5a. CONTRACT NUMBER. Enter all contract numbers as they appear in the report, e.g. F33615-86-C-5169.

5b. GRANT NUMBER. Enter all grant numbers as they appear in the report, e.g. AFOSR-82-1234.

5c. PROGRAM ELEMENT NUMBER. Enter all program element numbers as they appear in the report, e.g. 61101A.

5d. PROJECT NUMBER. Enter all project numbers as they appear in the report, e.g. 1F665702D1257; ILIR.

5e. TASK NUMBER. Enter all task numbers as they appear in the report, e.g. 05; RF0330201; T4112.

5f. WORK UNIT NUMBER. Enter all work unit numbers as they appear in the report, e.g. 001; AFAPL30480105.

6. AUTHOR(S). Enter name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. The form of entry is the last name, first name, middle initial, and additional qualifiers separated by commas, e.g. Smith, Richard, J, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES). Self-explanatory.

8. PERFORMING ORGANIZATION REPORT NUMBER. Enter all unique alphanumeric report numbers assigned by the performing organization, e.g. BRL-1234; AFWL-TR-85-4017-Vol-21-PT-2.

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES). Enter the name and address of the organization(s) financially responsible for and monitoring the work.

10. SPONSOR/MONITOR'S ACRONYM(S). Enter, if available, e.g. BRL, ARDEC, NADC.

11. SPONSOR/MONITOR'S REPORT NUMBER(S). Enter report number as assigned by the sponsoring/ monitoring agency, if available, e.g. BRL-TR-829; -215.

12. DISTRIBUTION/AVAILABILITY STATEMENT. Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. If additional limitations/ restrictions or special markings are indicated, follow agency authorization procedures, e.g. RD/FRD, PROPIN, ITAR, etc. Include copyright information.

13. SUPPLEMENTARY NOTES. Enter information not included elsewhere such as: prepared in cooperation with; translation of; report supersedes; old edition number, etc.

14. ABSTRACT. A brief (approximately 200 words) factual summary of the most significant information.

15. SUBJECT TERMS. Key words or phrases identifying major concepts in the report.

16. SECURITY CLASSIFICATION. Enter security classification in accordance with security classification regulations, e.g. U, C, S, etc. If this form contains classified information, stamp classification level on the top and bottom of this page.

17. LIMITATION OF ABSTRACT. This block must be completed to assign a distribution limitation to the abstract. Enter UU (Unclassified Unlimited) or SAR (Same as Report). An entry in this block is necessary if the abstract is to be limited.

Foundations and Applications of Program Obfuscation

Rafael Pass
Department of Computer Science
Cornell University

Program Manager: Tristan Nguyen
Final report: 8/15/2015 - 8/14/2018
Mathematics, Information and Life Sciences Directorate
Information Operations and Security

Contents

1 Introduction

The goal of *program obfuscation* is to "scramble" a computer program, hiding its implementation details (making it hard to "reverse-engineer"), while preserving the functionality (i.e, input/output behavior) of the program: an obfuscator $\mathcal{O}$ is a compiler which takes a program C and compiles it to a program $C' = \mathcal{O}(C)$ that has exactly the same functionality as C (i.e., they provide the same output on every input), yet the code of C' is "unintelligible". Program obfuscation is widely used in practice—for example, in applications such as Skype or Instagram, it permits distributing code to a *huge number of users*, enabling them to communicate and perform large-scale distributed computations, while ensuring the users only employ the service in the intended way. However, these usages largely rely on *ad-hoc heuristics* that often get broken (for instance, a google search reveals multiple practical methods for reverse-engineering Instagram, exposing all secret keys used by the obfuscated Instagram clients).

Rather, similar to the modern study of Cryptography, it would be desirable to put the study of program obfuscation under a *rigorous mathematical treatment* by:

- precisely defining its security properties,
- precisely defining some computational intractrability assumptions (e.g., the hardness of factoring products of large primes), and
- *proving* that any attack on the obfuscation must violate the computational assumptions.

The principal goals of our original proposal was a) to provide such a treatment, and b) more generally, studying techniques for ensuring that large-scale distributed services can only be employed in their intended way. During this reporting period we have made progress on both of these directions, but we here mostly focus on a).

2 Foundations of iO

An in-depth study of program obfuscation was initiated in the work of Barak, Goldreich, Impagliazzo, Rudich, Sahai, Vadhan, and Yang [BGI+01]; their central result shows that a natural "simulation-based" definition of program obfuscation—called *virtual black-box (VBB)* obfuscation—which roughly speaking requires that every bit that can be learnt from the description of the code can be "simulated" using just black-box access to the functionality—is impossible to acheive if we are after "general-purpose" obfuscators that can obfuscate any program. Consequently, Barak *et al.* [BGI+01] suggested a weaker notion of obfuscation—the notion of *indistinguishability obfuscation (**iO**)*. Roughly speaking, this notion requires that 'obfuscations $\mathcal{O}(C_1)$ and $\mathcal{O}(C_2)$ of any two *equivalent* programs C_1 and C_2 (i.e., whose outputs agree on all inputs) are computationally indistinguishable. In the last couple of years, several surprising

cryptographic applications of such relaxed notions have been demonstrated in the literature (some by the PI) [GGH+13, SW14, BCP14, BCPR13, BP13], and following the groundbreaking work of Garg, Gentry, Halevi, Raykova and Sahai [GGH+13] several *candidate* constructions of such obfuscators have been proposed (some by the PI) [GGH+13, BR14, BGK+13, BCP14]. But the central open question in the field is whether constructions of obfuscators satisfying indistinguishability obfuscation can be *reduced* (using a mathematical proof) to some natural computational hardness assumption (that isn't broken!).

iO and Function Compression Let us note that for all known application of **iO**, it is important that the obfuscator is *efficient*—namely, polynomial-time. Indeed, as already observed by [BGI+01], it is "trivial" to provide an *inefficient* **iO** with running time $poly(|C|, \lambda) \cdot 2^n$, where C is the circuit to be obfuscated, λ is the security parameter, and n is the input length of C, exists *unconditionally*: simply output the function table of C (i.e., the output of C on all possible inputs). Recall that, in contrast, for "standard" (efficient) **iO**, the running time and size of the obfuscator is required to be $poly(|C|, \lambda)$—namely, *polylogarithmic* in the size of the truth table of C).

In a work appearing in PKC'16, we show that **iO** with such a "trivial" exponential running-time may actually be interesting, as long as the obfuscator manages to output just a *slight compression* of the trivial function table. More precisely, we introduce a notion called **XiO** where the the running-time of the obfuscator may still be "trivial" (namely, $poly(|C|, \lambda) \cdot 2^n$) but we now require that the obfuscated code is of size $poly(|C|, \lambda) \cdot 2^{n(1-\epsilon)}$, where $\epsilon > 0$). Perhaps surprisingly, we show that under standard cryptographic hardness assumptions, **XiO** (with subexponential security) implies the standard notion of **iO**. Intringuinly, we also show that this notion is closely related to the complexity-theoretic notion of "function compression"—in fact, **XiO** may be view as a computational analog of this notion.

This result, and results obtained in our TCC'16 paper (which related **iO** to a compressing form of a different well-studied cryptographic primitive called randomized encodings) shows that the core difficutly in acheiving **iO** boils down to questions about function compression.

(We mention that this new insight, as well as our techniques, have now lead to several new breakthrough results—for instance, (our previously graduated Ph.D student, now a faculty at UC SB) Huijia Lin, showed how to obtain **iO** from constant-degree multilinear maps—all previous contructions required multilinear maps with polynomial degree.)

In a very recent CRYPTO'18 paper, we intiate a more general study of "weakly compressing obfuscation", showing among other things, interesting connections between statistically secure **XiO** and results in learning theory, that **XiO** and one-way function is likely to be a weak primitive—in that it does not imply even public-key encryption in a black-box way (whereas **iO** plus one-way functions does). We also present methods for amplifying the correctess of **XiO**.

- Huijia Lin, Rafael Pass, Karn Seth, Sidharth Telang:Indistinguishability Obfuscation with

Non-trivial Efficiency. Public Key Cryptography (2) 2016: 447-462

- Huijia Lin, Rafael Pass, Karn Seth, Sidharth Telang: Output-Compressing Randomized Encodings and Applications. TCC (A1) 2016: 96-124
- Nir Bitansky, Ran Canetti, Sanjam Garg, Justin Holmgren, Abhishek Jain, Huijia Lin, Rafael Pass, Sidharth Telang, Vinod Vaikuntanathan: Indistinguishability Obfuscation for RAM Programs and Succinct Randomized Encodings. SIAM J. Comput. 47(3): 1123-1210 (2018)
- Gilad Asharov, Naomi Ephraim, Ilan Komargodski, Rafael Pass: On the Complexity of Compressing Obfuscation. CRYPTO (3) 2018: 753-783

Barriers to Acheiving Obfuscation All known candidate construction for acheiving **iO** rely on a new mathematical construct called a multilinear map. In two works appearing in TCC'16, we present the first barriers to acheiving strong forms of general-purpose obfuscation using such multilinear maps.

- Rafael Pass, Abhi Shelat: Impossibility of VBB Obfuscation with Ideal Constant-Degree Graded Encodings. TCC (A1) 2016: 3-17
- Mohammad Mahmoody, Ameer Mohammed, Soheil Nematihaji, Rafael Pass, Abhi Shelat: Lower Bounds on Assumptions Behind Indistinguishability Obfuscation. TCC (A1) 2016: 49-66
- Elette Boyle, Rafael Pass: Limits of Extractability Assumptions with Distributional Auxiliary Input. ASIACRYPT (2) 2015: 236-261

3 Large-Scale MPC

We initiated the study of Cryptography for *Parallel* RAM programs. If we expect cryptography to be used in large-scale settings, it is crucial to leverage the parallel nature of computation that is ubiquitous on the internet today (through e.g., Map Reduce). In contrast, essentially all cryptographic techniques for securely computing (RAM) programs require sequentializing the program before securely computing it. In papers appearing in CRYPTO 2015 and TCC 2016, we show how to overcome this barriers and achieve secure computations which maintain the same level of parallelism as the original parallel program.

We mention that an intruiging open problem in the area of security for Parallel RAM programs is that the computational overhead, although only polylogarithmic, still is a lot higher than the computational overhead needed for securely computing just RAM program. During the reporting period (as well as the next one), we have been working on overcoming this gap.

- Elette Boyle, Kai-Min Chung, Rafael Pass: Oblivious Parallel RAM and Applications. TCC (A2) 2016: 175-204
- Elette Boyle, Kai-Min Chung, Rafael Pass: Large-Scale Secure Computation: Multi-party Computation for (Parallel) RAM Programs. CRYPTO (2) 2015: 742-762

2-round non-malleable commitments Non-malleable commitments are one of the most fundamental cryptographic building blocks. Notably, they are a central component for the design of round-efficient secure multi-party computation protocols. A major open problem in the literature was coming up with non-malleable commitments with only 2 communication rounds; in fact, one of my earlier results from a few year ago shows that using standard proof techniques (so-called black-box techniques), we cannot base 2-round non-malleable commitments on standard polynomial hardness. In our FOCS17 paper, joint with Huijia Lin and Pratik Soni, we show how to overcome this barrier, and construct 2-round non-malleable commitments based on sub-exponential hardness assumptions. (The paper was invited to the special issue for selected paper from FOCS17)

- Huijia Lin, Rafael Pass, Pratik Soni: Two-Round and Non-Interactive Concurrent Non-Malleable Commitments from Time-Lock Puzzles. FOCS 2017: 576-587

Theoretical foundations of blockchains While blockchains have been extensively featured in the media and there is a significant interest in industry, up until recently, there was not even a definition of what a blockchain actually is. In a EuroCrypt 2017 paper, jointly with my Ph.D student (Seeman) and visitor (Shelat), we provided the first formal definition of a blockchain, and next demonstrated that Nakamotos ingenious blockchain protocol (which forms the basis of the Bitcoin crypto currency) indeed provably satisfies this abstraction, and in particular satisfies "consistency" and "liveness", as long as the parameters are appropriately set as a function of the delay in the network.

Subsequently, joint with Elaine Shi, I have been working on developing new blockchain protocols that overcome some of the limitations of Nakamotos blockchains: Our FruitChain protocol (PODC17), is the first incentive-compatible blockchains (overcoming in a provably-secure way selfish mining attacks). Our Hybrid Consensus (DISC17) shows how to (in theory) overcome the scalability issues with Nakamotos blockchains. Our Sleepy Consensus (AsiaCrypt17) shows how to remove the computationally wasteful proof of work from Nakamotos blockchain.

- Rafael Pass, Lior Seeman, Abhi Shelat: Analysis of the Blockchain Protocol in Asynchronous Networks. EUROCRYPT (2) 2017: 643-673
- Rafael Pass, Elaine Shi: FruitChains: A Fair Blockchain. PODC 2017: 315-324

- Rafael Pass, Elaine Shi: Hybrid Consensus: Efficient Consensus in the Permissionless Model. DISC 2017: 39:1-39:16
- Ben Fisch, Rafael Pass, Abhi Shelat: Socially Optimal Mining Pools. WINE 2017: 205-218
- Joseph Y. Halpern, Rafael Pass: A Knowledge-Based Analysis of the Blockchain Protocol. TARK 2017: 324-335
- Rafael Pass, Elaine Shi: The Sleepy Model of Consensus. ASIACRYPT (2) 2017: 380-409
- Rafael Pass, Elaine Shi: Rethinking Large-Scale Consensus. CSF 2017: 115-129

References

[BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In *TCC*, pages 52–73, 2014.

[BCPR13] Nir Bitansky, Ran Canetti, Omer Paneth, and Alon Rosen. Indistinguishability obfuscation vs. auxiliary-input extractable functions: One must fall. *IACR Cryptology ePrint Archive*, 2013:641, 2013.

[BGI+01] Boaz Barak, Oded Goldreich, Rusell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *Advances in Cryptology CRYPTO 2001*, pages 1–18. Springer, 2001.

[BGK+13] Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In *EuroCrypt'14*, 2013.

[BP13] Elette Boyle and Rafael Pass. Limits of extractability assumptions with distributional auxiliary input. 2013.

[BR14] Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In *TCC*, pages 1–25, 2014.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *Proc. of FOCS 2013*, 2013.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. *Proc. of STOC 2014*, 2014.