Final Project Discussion

CS 598 DH

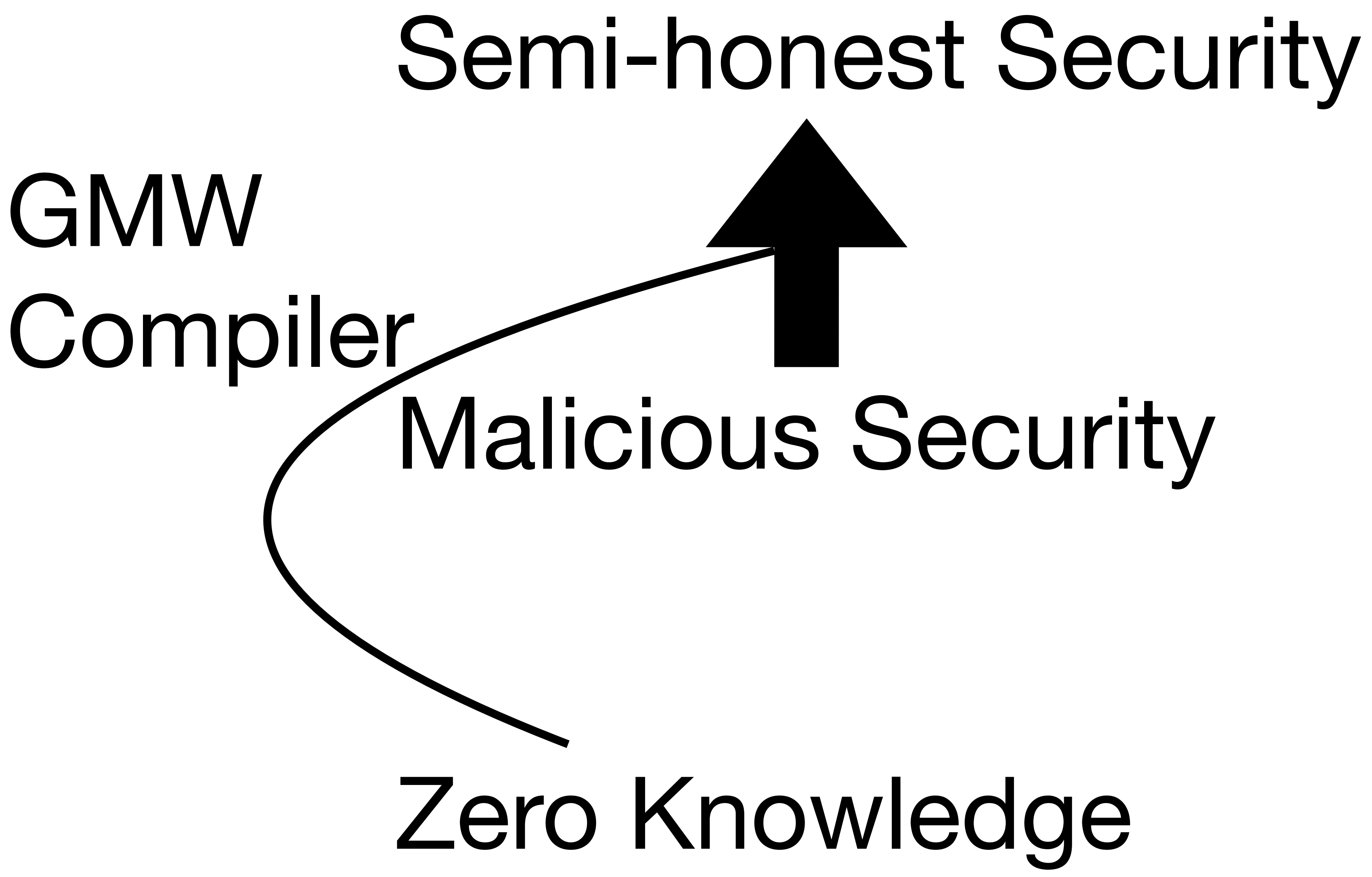
Today’s objectives

Survey possible topics for your project

Form a team

Start looking through the literature

Setting



Semi-honest Security

GMW
Compiler

Malicious Security

Zero Knowledge

General-Purpose Tools

GMW Protocol

- Multi-party
- Multi-round

Garbled Circuit

- Constant Round
- Two Party

Primitives

Oblivious Transfer

Pseudorandom functions/encryption

Commitments

“Are there other security models?”

Covert Security with Public Verifiability: Faster, Leaner, and Simpler

Cheng Hong, Alibaba Group, vince.hc@alibaba-inc.com
Jonathan Katz, University of Maryland, jkatz@cs.umd.edu
Vladimir Kolesnikov, Georgia Tech, kolesnikov@gatech.edu
Wen-jie Lu, University of Tsukuba, riku@mdl.cs.tsukuba.ac.jp
Xiao Wang, MIT and Boston University, wangxiao@northwestern.edu

November 14, 2018

Abstract

Guaranteed Output in $O(\sqrt{n})$ Rounds for Round-Robin Sampling Protocols*

Ran Cohen, cohenran@idc.ac.il, Reichman University
Jack Doerner, j@ckdoerner.net, Northe
Yashvanth Kondi

ember 22, 202

Abstract

round-robin secu
ture, such as the
omial commitme
structure, protoc
t rounds, where
ile them generica
rounds. Our com
ishonest majority.
re $\Omega(n)$ sequentia
. Our compiled
the output, as a
to Cleve's impos
of the aforementio

this work appeare

Complete Fairness in Secure Two-Party Computation

S. Dov Gordon*, Carmit Hazay†, Jonathan Katz‡, Yehuda Lindell§

Abstract

In the setting of secure two-party computation, two mutually distrusting parties wish to compute some function of their inputs while preserving, to the extent possible, various security properties such as privacy, correctness, and more. One desirable property is *fairness* which guarantees, informally, that if one party receives its output, then the other party does too. Cleve (STOC 1986) showed that complete fairness cannot be achieved *in general* without an

Efficient Secure Multiparty Computation with Identifiable Abort

Carsten Baum[1]*, Emmanuela Orsini[2] †, and Peter Scholl[2] ‡

[1] Department of Computer Science, Aarhus University
cbaum@cs.au.dk
[2] Department of Computer Science, University of Bristol
{Emmanuela.Orsini, Peter.Scholl}@bristol.ac.uk

Abstract. We study secure multiparty computation (MPC) in the dishonest majority setting providing security with identifiable abort, where if the protocol aborts, the honest parties can agree upon the identity of a corrupt party. All known constructions that achieve this notion require expensive zero-knowledge techniques to obtain active security, so are not practical.
In this work, we present the first efficient MPC protocol with identifiable abort. Our protocol has an information-theoretic online phase with message complexity $O(n^2)$ for each secure multiplication (where n is the number of parties), similar to the BDOZ protocol (Bendlin et al., Eurocrypt 2011), and a factor in the security parameter lower than the identifiable abort protocol of Ishai et al. (Crypto 2014). A key component of our protocol is a linearly homomorphic information-theoretic signature scheme, for which we provide the first definitions and construction based on a previous non-homomorphic scheme. We then show how to implement the preprocessing for our protocol using somewhat homomorphic encryption, similarly to the SPDZ protocol (Damgård et al., Crypto 2012) and other recent works with applicable efficiency improvements.

Guaranteed Output Delivery Comes Free in Honest Majority MPC

Vipul Goyal[1], Yifan Song[1](✉), and Chenzhi Zhu[2]

Abstract. We study th
output delivery over poi
of a public broadcast ch
s.t. the communication c
number of works have fo
to the above question ha
complexity of evaluating
We resolve the above que
$O(Cn\phi)$ bits (ignoring fi
element in the field, C is
construction where the
protocol. This represents
plexity of $O(C(n\phi-\kappa)+$
depth of the circuit. Fur
5.5 field elements per pa
corrupted parties have be
which requires 5.5 field e

1 Introduction

YOSO: You Only Speak Once
Secure MPC with Stateless Ephemeral Roles

Craig Gentry[1](✉), Shai Halevi[1], Hugo Krawczyk[1], Bernardo Magri[2], Jesper Buus Nielsen[2], Tal Rabin[1,3], and Sophia Yakoubov[4]

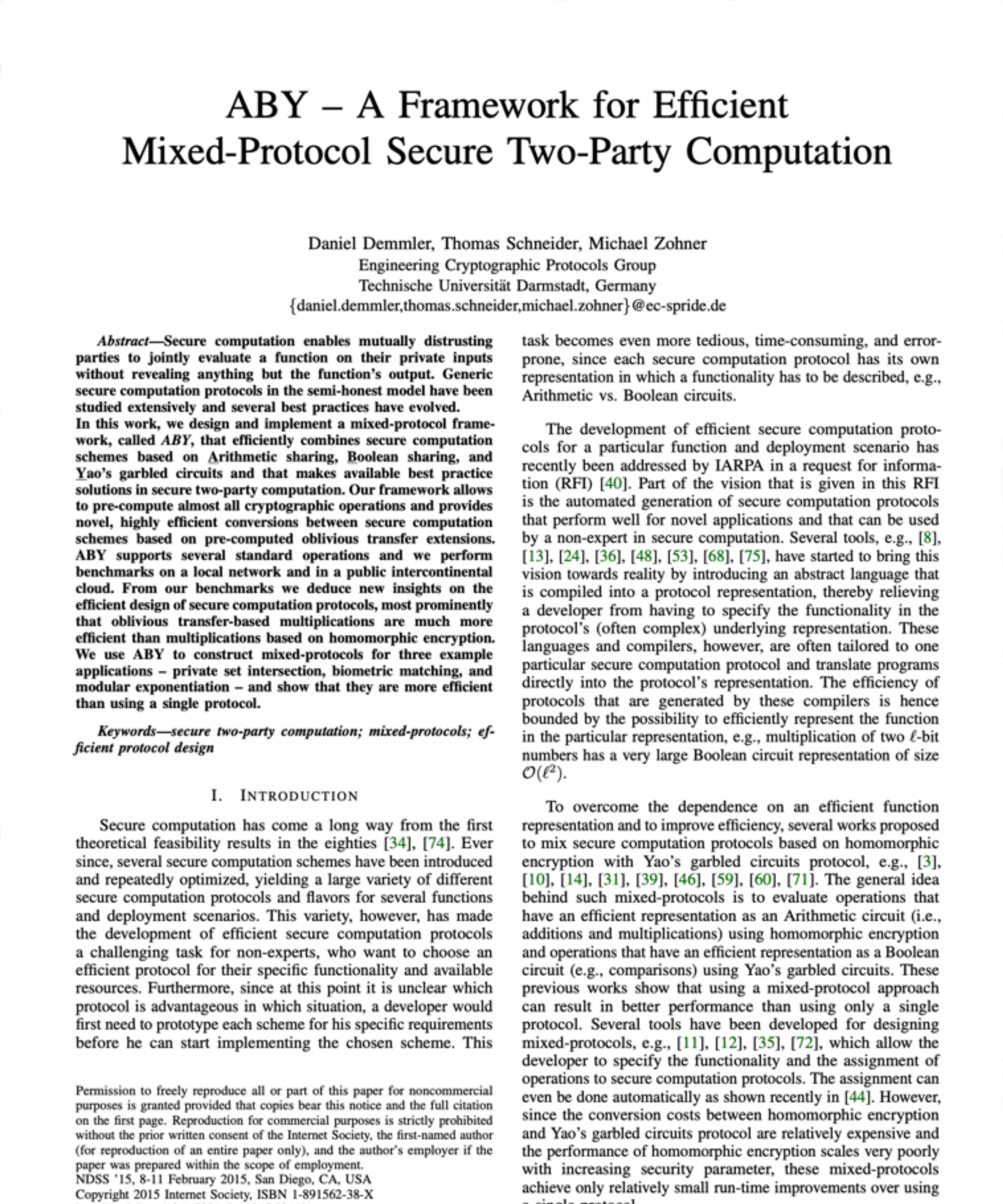
[1] Algorand Foundation, New York, USA
hugo@ee.technion.ac.il
[2] Concordium Blockchain Research Center, Aarhus University, Aarhus, Denm
[3] UPenn, Philadelphia, USA
[4] Aarhus University, Aarhus, Denmark

Abstract. The inherent difficulty of maintaining stateful environments

“Are there other security properties/models?”

- Fairness
- Identifiable Abort
- Guaranteed Output Delivery
- Adaptive Security
- Covert Security
- Asynchronous Networks
- Publicly Verifiable Covert Security
- …

“combine GMW and GC?”



ABY – A Framework for Efficient Mixed-Protocol Secure Two-Party Computation

Daniel Demmler, Thomas Schneider, Michael Zohner
Engineering Cryptographic Protocols Group
Technische Universität Darmstadt, Germany
{daniel.demmler,thomas.schneider,michael.zohner}@ec-spride.de

***Abstract*—Secure computation enables mutually distrusting parties to jointly evaluate a function on their private inputs without revealing anything but the function's output. Generic secure computation protocols in the semi-honest model have been studied extensively and several best practices have evolved.**

In this work, we design and implement a mixed-protocol framework, called *ABY*, that efficiently combines secure computation schemes based on Arithmetic sharing, Boolean sharing, and Yao's garbled circuits and that makes available best practice solutions in secure two-party computation. Our framework allows to pre-compute almost all cryptographic operations and provides novel, highly efficient conversions between secure computation schemes based on pre-computed oblivious transfer extensions. ABY supports several standard operations and we perform benchmarks on a local network and in a public intercontinental cloud. From our benchmarks we deduce new insights on the efficient design of secure computation protocols, most prominently that oblivious transfer-based multiplications are much more efficient than multiplications based on homomorphic encryption. We use ABY to construct mixed-protocols for three example applications – private set intersection, biometric matching, and modular exponentiation – and show that they are more efficient than using a single protocol.

Keywords*—secure two-party computation; mixed-protocols; efficient protocol design

I. Introduction

Secure computation has come a long way from the first theoretical feasibility results in the eighties [34], [74]. Ever since, several secure computation schemes have been introduced and repeatedly optimized, yielding a large variety of different secure computation protocols and flavors for several functions and deployment scenarios. This variety, however, has made the development of efficient secure computation protocols a challenging task for non-experts, who want to choose an efficient protocol for their specific functionality and available resources. Furthermore, since at this point it is unclear which protocol is advantageous in which situation, a developer would first need to prototype each scheme for his specific requirements before he can start implementing the chosen scheme. This task becomes even more tedious, time-consuming, and error-prone, since each secure computation protocol has its own representation in which a functionality has to be described, e.g., Arithmetic vs. Boolean circuits.

The development of efficient secure computation protocols for a particular function and deployment scenario has recently been addressed by IARPA in a request for information (RFI) [40]. Part of the vision that is given in this RFI is the automated generation of secure computation protocols that perform well for novel applications and that can be used by a non-expert in secure computation. Several tools, e.g., [8], [13], [24], [36], [48], [53], [68], [75], have started to bring this vision towards reality by introducing an abstract language that is compiled into a protocol representation, thereby relieving a developer from having to specify the functionality in the protocol's (often complex) underlying representation. These languages and compilers, however, are often tailored to one particular secure computation protocol and translate programs directly into the protocol's representation. The efficiency of protocols that are generated by these compilers is hence bounded by the possibility to efficiently represent the function in the particular representation, e.g., multiplication of two ℓ-bit numbers has a very large Boolean circuit representation of size $\mathcal{O}(\ell^2)$.

To overcome the dependence on an efficient function representation and to improve efficiency, several works proposed to mix secure computation protocols based on homomorphic encryption with Yao's garbled circuits protocol, e.g., [3], [10], [14], [31], [39], [46], [59], [60], [71]. The general idea behind such mixed-protocols is to evaluate operations that have an efficient representation as an Arithmetic circuit (i.e., additions and multiplications) using homomorphic encryption and operations that have an efficient representation as a Boolean circuit (e.g., comparisons) using Yao's garbled circuits. These previous works show that using a mixed-protocol approach can result in better performance than using only a single protocol. Several tools have been developed for designing mixed-protocols, e.g., [11], [12], [35], [72], which allow the developer to specify the functionality and the assignment of operations to secure computation protocols. The assignment can even be done automatically as shown recently in [44]. However, since the conversion costs between homomorphic encryption and Yao's garbled circuits protocol are relatively expensive and the performance of homomorphic encryption scales very poorly with increasing security parameter, these mixed-protocols achieve only relatively small run-time improvements over using a single protocol.

NDSS '15, 8-11 February 2015, San Diego, CA, USA
Copyright 2015 Internet Society, ISBN 1-891562-38-X

“how many rounds are needed?”

Round-Optimal Black-Box MPC in the Plain Model

Yuval Ishai* Dakshita Khurana† Amit Sahai‡ Akshayaram Srinivasan§

Abstract

We give the first construction of a fully black-box round-optimal secure multiparty computation (MPC) protocol in the plain model. Our protocol makes black-box use of a sub-exponentially secure two-message statistical sender private oblivious transfer (SSP-OT), which in turn can be based on (sub-exponential variants of) almost all of the standard cryptographic assumptions known to imply public-key cryptography.

1 Introduction

The exact round complexity of secure computation has been a focus of research in cryptography over the past two decades. This has been especially well-studied in the synchronous setting in the plain model, with up to all-but-one static malicious corruptions. It is known that general-purpose secure multiparty computation (MPC) protocols in this setting admitting a *black-box simulator* require at least 4 rounds of simultaneous exchange [GK96b, KO04, GMPP16].[1] In this work we focus on MPC with black-box simulation. On the positive side, there has been a long sequence of works [GMPP16, BHP17, ACJ17, KS17, BGI+17, BGJ+18, CCG+20] improving the round complexity, culminating in a round-optimal construction that relies on the minimal assumption that a 4-round malicious-secure OT protocol exists [CCG+20].

Black-Box Use of Cryptography. Notably, all MPC protocols discussed above make non-black-box use of cryptography, which is typically associated with significant overheads in efficiency. It is interesting, from both a theoretical and a practical perspective, to realize *fully black-box protocols* [RTV04] where not only does the simulator make black-box use of an adversary, but also the construction itself can be fully specified given just oracle access to the input-output relation of the underlying cryptographic primitives, and without being given any explicit representation of these primitives. In the following, we refer to this standard notion of fully black-box protocols as simply *black-box protocols*. The focus of this work is on the following natural question:

What is the round complexity of black-box MPC in the plain model?

*Technion. Email: yuvali@cs.technion.ac.il

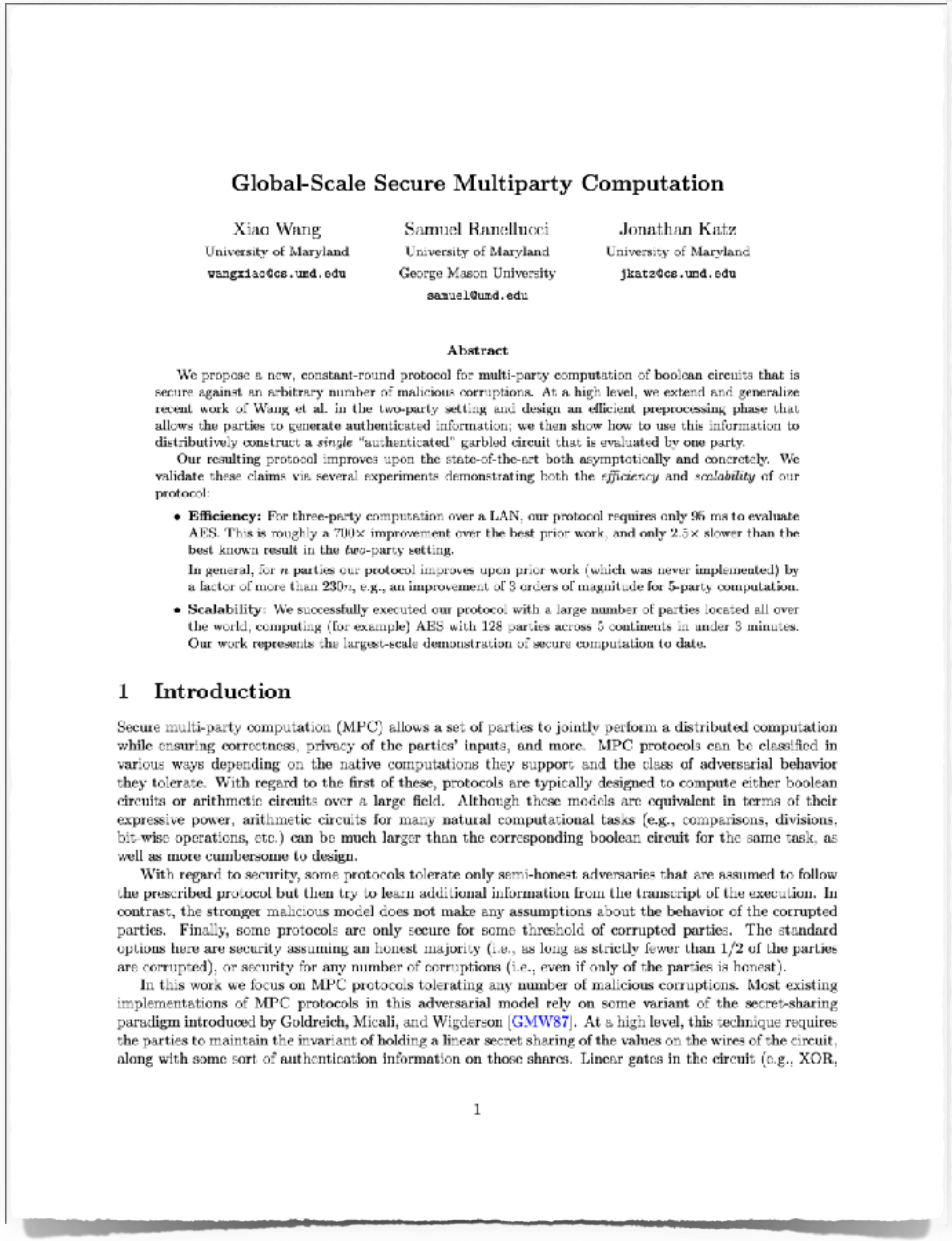
†UIUC. Email: dakshita@illinois.edu

‡UCLA. Email: sahai@cs.ucla.edu

§Tata Institute of Fundamental Research. Email: akshayaram.srinivasan@tifr.res.in

[1] By simultaneous message exchange we mean that in each round, every party can send a message over a broadcast

“malicious garbled circuits?”



Global-Scale Secure Multiparty Computation

Xiao Wang
University of Maryland
wangxiao@cs.umd.edu

Samuel Ranellucci
University of Maryland
George Mason University
samuel@umd.edu

Jonathan Katz
University of Maryland
jkatz@cs.umd.edu

Abstract

We propose a new, constant-round protocol for multi-party computation of boolean circuits that is secure against an arbitrary number of malicious corruptions. At a high level, we extend and generalize recent work of Wang et al. in the *two-party* setting and design an efficient preprocessing phase that allows the parties to generate authenticated information; we then show how to use this information to distributively construct a *single* "authenticated" garbled circuit that is evaluated by one party.

Our resulting protocol improves upon the state-of-the-art both asymptotically and concretely. We validate these claims via several experiments demonstrating both the *efficiency* and *scalability* of our protocol:

- **Efficiency:** For three-party computation over a LAN, our protocol requires only 95 ms to evaluate AES. This is roughly a 700× improvement over the best prior work, and only 2.5× slower than the best known result in the *two*-party setting.

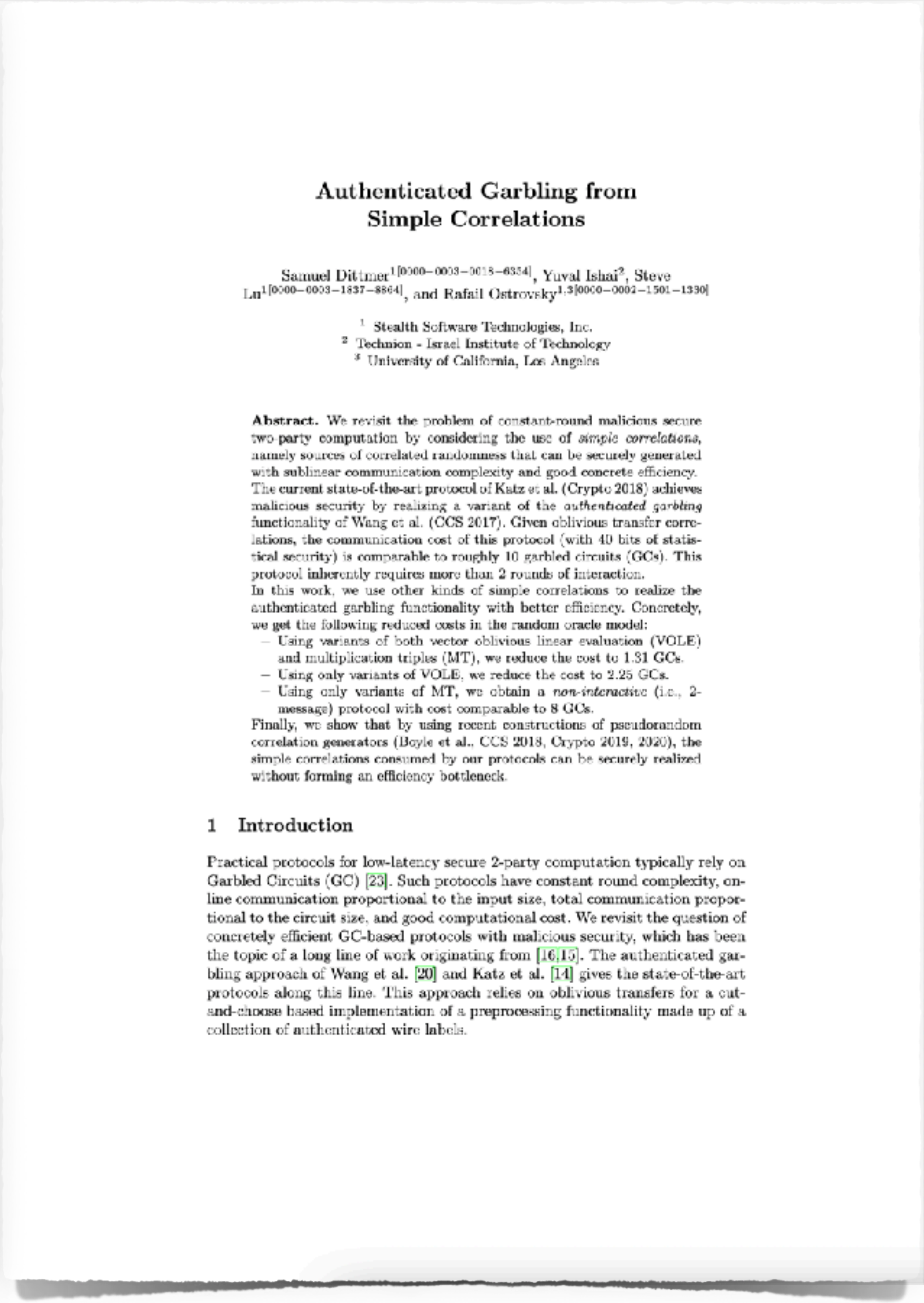
  In general, for n parties our protocol improves upon prior work (which was never implemented) by a factor of more than $230n$, e.g., an improvement of 3 orders of magnitude for 5-party computation.
- **Scalability**: We successfully executed our protocol with a large number of parties located all over the world, computing (for example) AES with 128 parties across 5 continents in under 3 minutes. Our work represents the largest-scale demonstration of secure computation to date.

1 Introduction

Secure multi-party computation (MPC) allows a set of parties to jointly perform a distributed computation while ensuring correctness, privacy of the parties' inputs, and more. MPC protocols can be classified in various ways depending on the native computations they support and the class of adversarial behavior they tolerate. With regard to the first of these, protocols are typically designed to compute either boolean circuits or arithmetic circuits over a large field. Although these models are equivalent in terms of their expressive power, arithmetic circuits for many natural computational tasks (e.g., comparisons, divisions, bit-wise operations, etc.) can be much larger than the corresponding boolean circuit for the same task, as well as more cumbersome to design.

With regard to security, some protocols tolerate only semi-honest adversaries that are assumed to follow the prescribed protocol but then try to learn additional information from the transcript of the execution. In contrast, the stronger malicious model does not make any assumptions about the behavior of the corrupted parties. Finally, some protocols are only secure for some threshold of corrupted parties. The standard options here are security assuming an honest majority (i.e., as long as strictly fewer than 1/2 of the parties are corrupted), or security for any number of corruptions (i.e., even if only of the parties is honest).

In this work we focus on MPC protocols tolerating any number of malicious corruptions. Most existing implementations of MPC protocols in this adversarial model rely on some variant of the secret-sharing paradigm introduced by Goldreich, Micali, and Wigderson [GMW87]. At a high level, this technique requires the parties to maintain the invariant of holding a linear secret sharing of the values on the wires of the circuit, along with some sort of authentication information on those shares. Linear gates in the circuit (e.g., XOR,



Authenticated Garbling from Simple Correlations

Samuel Dittmer[1][0000-0003-0018-6354], Yuval Ishai[2], Steve Lu[1][0000-0003-1837-8864], and Rafail Ostrovsky[1,3][0000-0002-1501-1330]

[1] Stealth Software Technologies, Inc.
[2] Technion - Israel Institute of Technology
[3] University of California, Los Angeles

Abstract. We revisit the problem of constant-round malicious secure two-party computation by considering the use of *simple correlations*, namely sources of correlated randomness that can be securely generated with sublinear communication complexity and good concrete efficiency. The current state-of-the-art protocol of Katz et al. (Crypto 2018) achieves malicious security by realizing a variant of the *authenticated garbling* functionality of Wang et al. (CCS 2017). Given oblivious transfer correlations, the communication cost of this protocol (with 40 bits of statistical security) is comparable to roughly 10 garbled circuits (GCs). This protocol inherently requires more than 2 rounds of interaction.
In this work, we use other kinds of simple correlations to realize the authenticated garbling functionality with better efficiency. Concretely, we get the following reduced costs in the random oracle model:

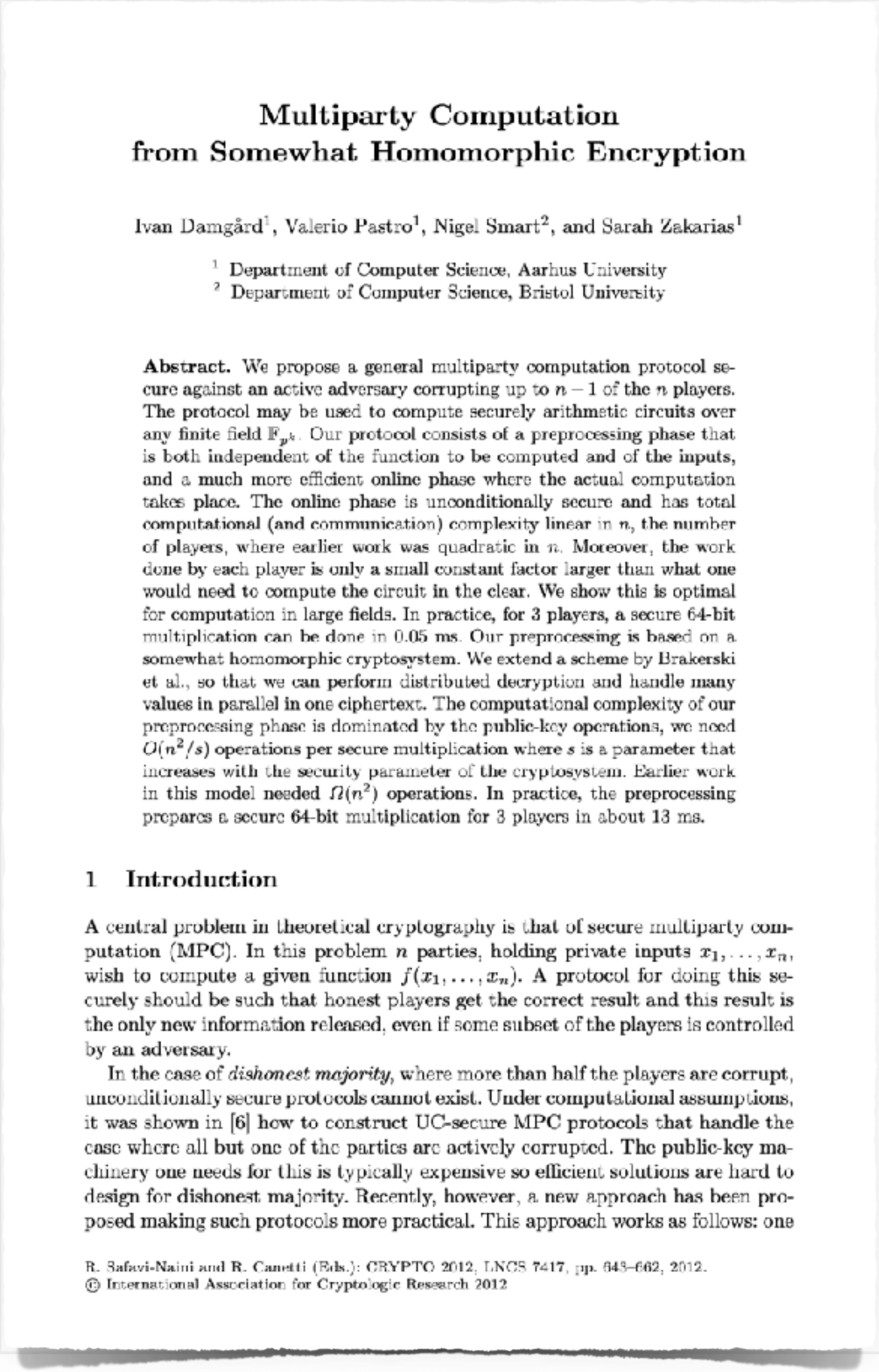
- Using variants of both vector oblivious linear evaluation (VOLE) and multiplication triples (MT), we reduce the cost to 1.31 GCs.
- Using only variants of VOLE, we reduce the cost to 2.25 GCs.
- Using only variants of MT, we obtain a *non-interactive* (i.e., 2-message) protocol with cost comparable to 8 GCs.

Finally, we show that by using recent constructions of pseudorandom correlation generators (Boyle et al., CCS 2018, Crypto 2019, 2020), the simple correlations consumed by our protocols can be securely realized without forming an efficiency bottleneck.

1 Introduction

Practical protocols for low-latency secure 2-party computation typically rely on Garbled Circuits (GC) [23]. Such protocols have constant round complexity, online communication proportional to the input size, total communication proportional to the circuit size, and good computational cost. We revisit the question of concretely efficient GC-based protocols with malicious security, which has been the topic of a long line of work originating from [16,15]. The authenticated garbling approach of Wang et al. [20] and Katz et al. [14] gives the state-of-the-art protocols along this line. This approach relies on oblivious transfers for a cut-and-choose based implementation of a preprocessing functionality made up of a collection of authenticated wire labels.

“Malicious GMW?”



Multiparty Computation from Somewhat Homomorphic Encryption

Ivan Damgård[1], Valerio Pastro[1], Nigel Smart[2], and Sarah Zakarias[1]

[1] Department of Computer Science, Aarhus University
[2] Department of Computer Science, Bristol University

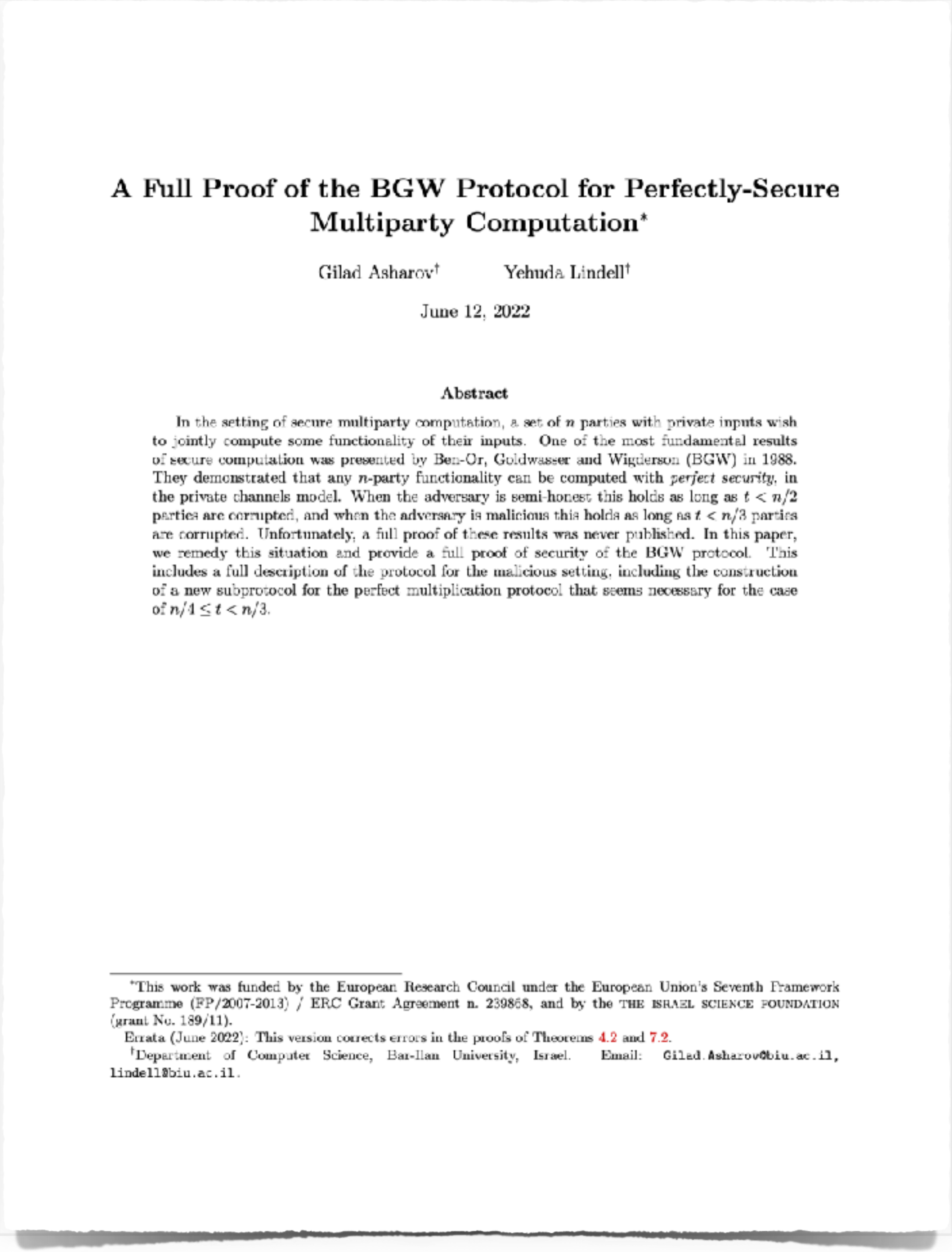
Abstract. We propose a general multiparty computation protocol secure against an active adversary corrupting up to $n-1$ of the n players. The protocol may be used to compute securely arithmetic circuits over any finite field $\mathbb{F}_{p^k}$. Our protocol consists of a preprocessing phase that is both independent of the function to be computed and of the inputs, and a much more efficient online phase where the actual computation takes place. The online phase is unconditionally secure and has total computational (and communication) complexity linear in n, the number of players, where earlier work was quadratic in n. Moreover, the work done by each player is only a small constant factor larger than what one would need to compute the circuit in the clear. We show this is optimal for computation in large fields. In practice, for 3 players, a secure 64-bit multiplication can be done in 0.05 ms. Our preprocessing is based on a somewhat homomorphic cryptosystem. We extend a scheme by Brakerski et al., so that we can perform distributed decryption and handle many values in parallel in one ciphertext. The computational complexity of our preprocessing phase is dominated by the public-key operations, we need $O(n^2/s)$ operations per secure multiplication where s is a parameter that increases with the security parameter of the cryptosystem. Earlier work in this model needed $\Omega(n^2)$ operations. In practice, the preprocessing prepares a secure 64-bit multiplication for 3 players in about 13 ms.

1 Introduction

A central problem in theoretical cryptography is that of secure multiparty computation (MPC). In this problem n parties, holding private inputs $x_1, \ldots, x_n$, wish to compute a given function $f(x_1, \ldots, x_n)$. A protocol for doing this securely should be such that honest players get the correct result and this result is the only new information released, even if some subset of the players is controlled by an adversary.

In the case of *dishonest majority*, where more than half the players are corrupt, unconditionally secure protocols cannot exist. Under computational assumptions, it was shown in [6] how to construct UC-secure MPC protocols that handle the case where all but one of the parties are actively corrupted. The public-key machinery one needs for this is typically expensive so efficient solutions are hard to design for dishonest majority. Recently, however, a new approach has been proposed making such protocols more practical. This approach works as follows: one

R. Safavi-Naini and R. Canetti (Eds.): CRYPTO 2012, LNCS 7417, pp. 643–662, 2012.
© International Association for Cryptologic Research 2012



A Full Proof of the BGW Protocol for Perfectly-Secure Multiparty Computation*

Gilad Asharov† Yehuda Lindell†

June 12, 2022

Abstract

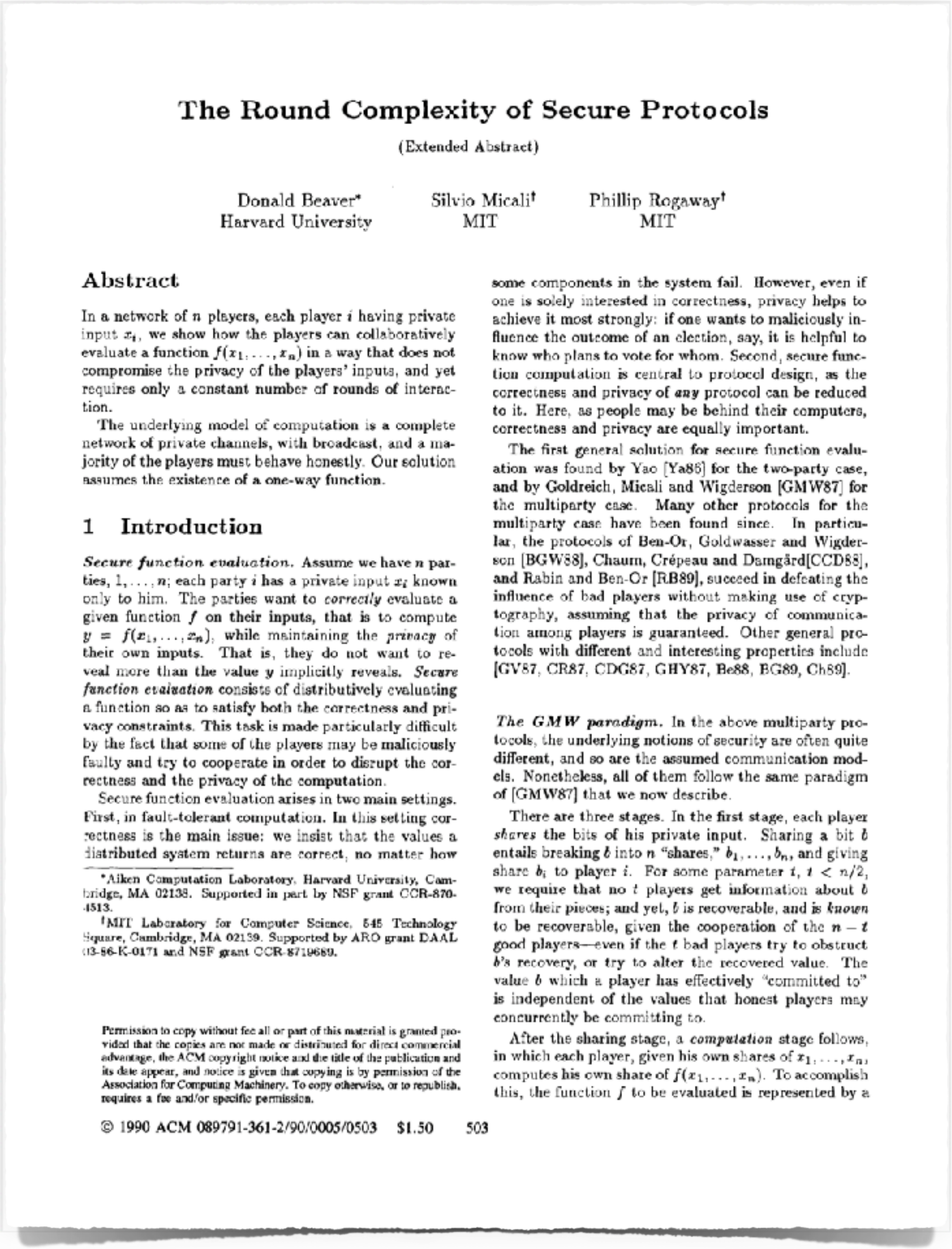
In the setting of secure multiparty computation, a set of n parties with private inputs wish to jointly compute some functionality of their inputs. One of the most fundamental results of secure computation was presented by Ben-Or, Goldwasser and Wigderson (BGW) in 1988. They demonstrated that any n-party functionality can be computed with *perfect security*, in the private channels model. When the adversary is semi-honest this holds as long as $t < n/2$ parties are corrupted, and when the adversary is malicious this holds as long as $t < n/3$ parties are corrupted. Unfortunately, a full proof of these results was never published. In this paper, we remedy this situation and provide a full proof of security of the BGW protocol. This includes a full description of the protocol for the malicious setting, including the construction of a new subprotocol for the perfect multiplication protocol that seems necessary for the case of $n/4 \leq t < n/3$.

*This work was funded by the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 239868, and by the THE ISRAEL SCIENCE FOUNDATION (grant No. 189/11).

Errata (June 2022): This version corrects errors in the proofs of Theorems 4.2 and 7.2.

†Department of Computer Science, Bar-Ilan University, Israel. Email: Gilad.Asharov@biu.ac.il, lindell@biu.ac.il.

“Constant-round multiparty protocols?”



The Round Complexity of Secure Protocols

(Extended Abstract)

Donald Beaver*
Harvard University

Silvio Micali†
MIT

Phillip Rogaway†
MIT

Abstract

In a network of n players, each player i having private input x_i, we show how the players can collaboratively evaluate a function $f(x_1, \ldots, x_n)$ in a way that does not compromise the privacy of the players' inputs, and yet requires only a constant number of rounds of interaction.

The underlying model of computation is a complete network of private channels, with broadcast, and a majority of the players must behave honestly. Our solution assumes the existence of a one-way function.

1 Introduction

Secure function evaluation. Assume we have n parties, $1, \ldots, n$; each party i has a private input x_i known only to him. The parties want to *correctly* evaluate a given function f on their inputs, that is to compute $y = f(x_1, \ldots, x_n)$, while maintaining the *privacy* of their own inputs. That is, they do not want to reveal more than the value y implicitly reveals. ***Secure function evaluation*** consists of distributively evaluating a function so as to satisfy both the correctness and privacy constraints. This task is made particularly difficult by the fact that some of the players may be maliciously faulty and try to cooperate in order to disrupt the correctness and the privacy of the computation.

Secure function evaluation arises in two main settings. First, in fault-tolerant computation. In this setting correctness is the main issue: we insist that the values a distributed system returns are correct, no matter how some components in the system fail. However, even if one is solely interested in correctness, privacy helps to achieve it most strongly: if one wants to maliciously influence the outcome of an election, say, it is helpful to know who plans to vote for whom. Second, secure function computation is central to protocol design, as the correctness and privacy of *any* protocol can be reduced to it. Here, as people may be behind their computers, correctness and privacy are equally important.

The first general solution for secure function evaluation was found by Yao [Ya86] for the two-party case, and by Goldreich, Micali and Wigderson [GMW87] for the multiparty case. Many other protocols for the multiparty case have been found since. In particular, the protocols of Ben-Or, Goldwasser and Wigderson [BGW88], Chaum, Crépeau and Damgård[CCD88], and Rabin and Ben-Or [RB89], succeed in defeating the influence of bad players without making use of cryptography, assuming that the privacy of communication among players is guaranteed. Other general protocols with different and interesting properties include [GV87, CR87, CDG87, GHY87, Be88, BG89, Ch89].

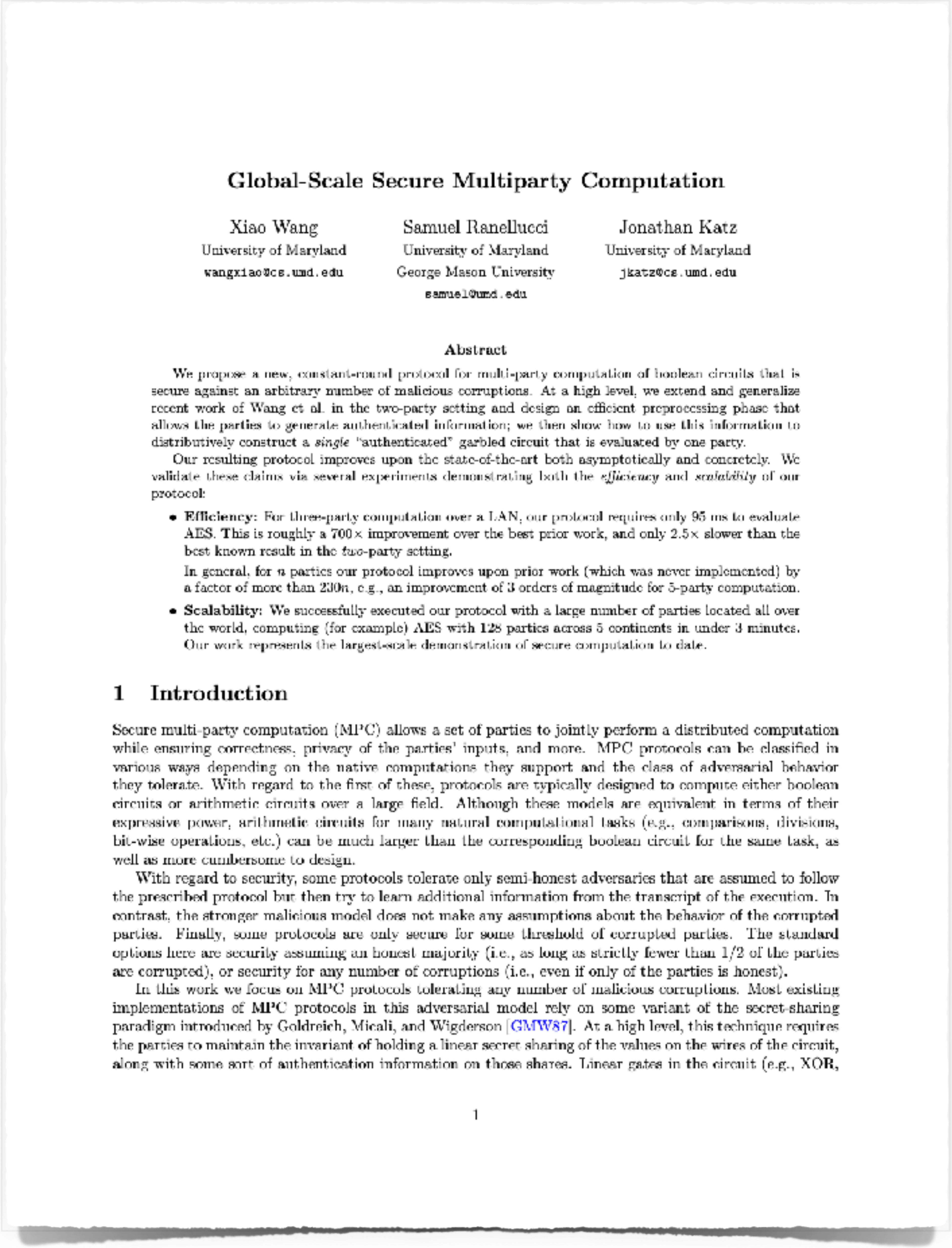
The GMW paradigm. In the above multiparty protocols, the underlying notions of security are often quite different, and so are the assumed communication models. Nonetheless, all of them follow the same paradigm of [GMW87] that we now describe.

There are three stages. In the first stage, each player *shares* the bits of his private input. Sharing a bit b entails breaking b into n “shares,” $b_1, \ldots, b_n$, and giving share b_i to player i. For some parameter t, $t < n/2$, we require that no t players get information about b from their pieces; and yet, b is recoverable, and is *known* to be recoverable, given the cooperation of the $n - t$ good players—even if the t bad players try to obstruct b's recovery, or try to alter the recovered value. The value b which a player has effectively “committed to” is independent of the values that honest players may concurrently be committing to.

After the sharing stage, a *computation* stage follows, in which each player, given his own shares of $x_1, \ldots, x_n$, computes his own share of $f(x_1, \ldots, x_n)$. To accomplish this, the function f to be evaluated is represented by a

*Aiken Computation Laboratory, Harvard University, Cambridge, MA 02138. Supported in part by NSF grant CCR-870-4513.

†MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 02139. Supported by ARO grant DAAL 03-86-K-0171 and NSF grant CCR-8719689.



Global-Scale Secure Multiparty Computation

Xiao Wang
University of Maryland
wangxiao@cs.umd.edu

Samuel Ranellucci
University of Maryland
George Mason University
samuel@umd.edu

Jonathan Katz
University of Maryland
jkatz@cs.umd.edu

Abstract

We propose a new, constant-round protocol for multi-party computation of boolean circuits that is secure against an arbitrary number of malicious corruptions. At a high level, we extend and generalize recent work of Wang et al. in the two-party setting and design an efficient preprocessing phase that allows the parties to generate authenticated information; we then show how to use this information to distributively construct a *single* “authenticated” garbled circuit that is evaluated by one party.

Our resulting protocol improves upon the state-of-the-art both asymptotically and concretely. We validate these claims via several experiments demonstrating both the *efficiency* and *scalability* of our protocol:

- **Efficiency:** For three-party computation over a LAN, our protocol requires only 95 ms to evaluate AES. This is roughly a 700× improvement over the best prior work, and only 2.5× slower than the best known result in the *two*-party setting.

  In general, for n parties our protocol improves upon prior work (which was never implemented) by a factor of more than $230n$, e.g., an improvement of 3 orders of magnitude for 5-party computation.
- **Scalability:** We successfully executed our protocol with a large number of parties located all over the world, computing (for example) AES with 128 parties across 5 continents in under 3 minutes. Our work represents the largest-scale demonstration of secure computation to date.

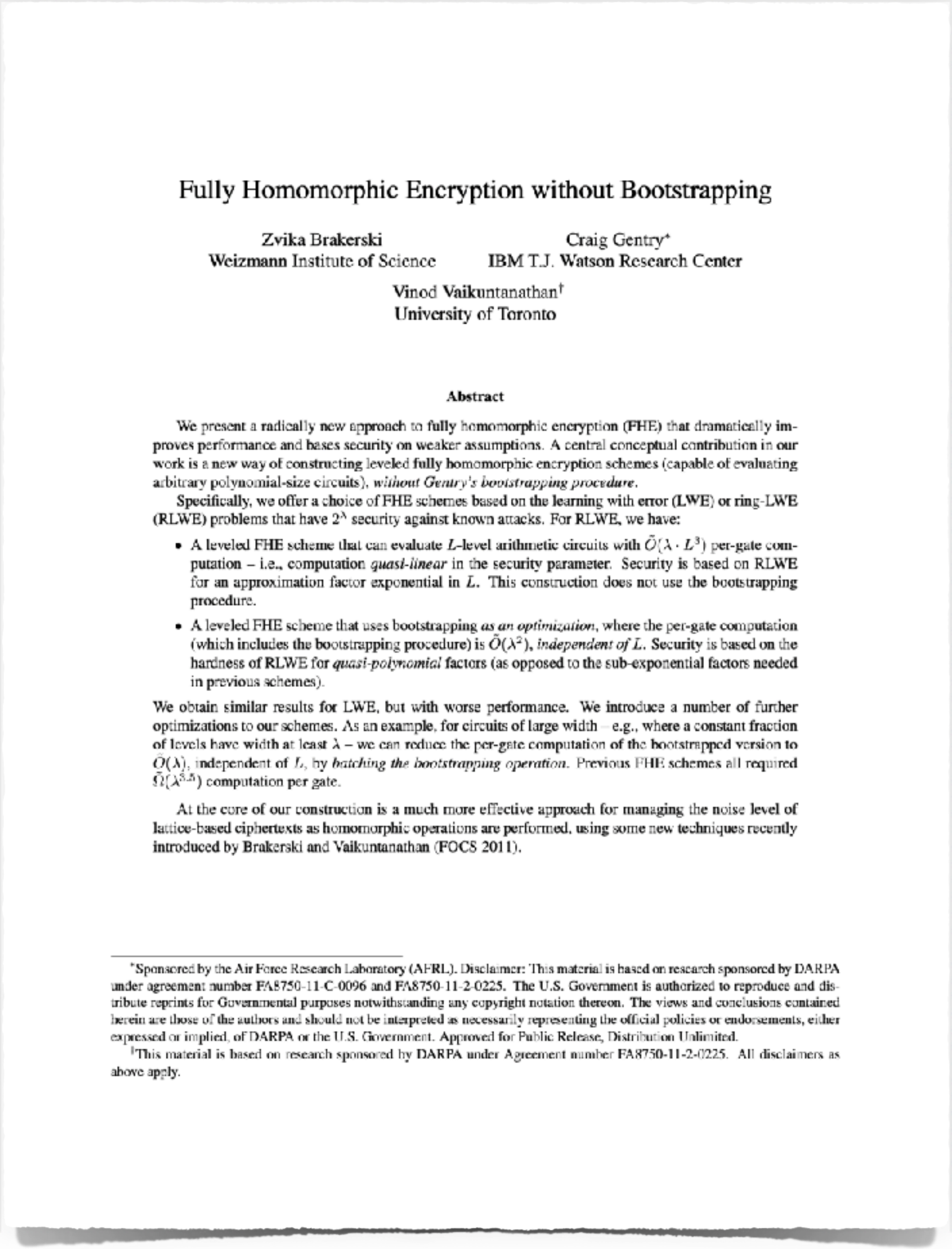
1 Introduction

Secure multi-party computation (MPC) allows a set of parties to jointly perform a distributed computation while ensuring correctness, privacy of the parties' inputs, and more. MPC protocols can be classified in various ways depending on the native computations they support and the class of adversarial behavior they tolerate. With regard to the first of these, protocols are typically designed to compute either boolean circuits or arithmetic circuits over a large field. Although these models are equivalent in terms of their expressive power, arithmetic circuits for many natural computational tasks (e.g., comparisons, divisions, bit-wise operations, etc.) can be much larger than the corresponding boolean circuit for the same task, as well as more cumbersome to design.

With regard to security, some protocols tolerate only semi-honest adversaries that are assumed to follow the prescribed protocol but then try to learn additional information from the transcript of the execution. In contrast, the stronger malicious model does not make any assumptions about the behavior of the corrupted parties. Finally, some protocols are only secure for some threshold of corrupted parties. The standard options here are security assuming an honest majority (i.e., as long as strictly fewer than 1/2 of the parties are corrupted), or security for any number of corruptions (i.e., even if only of the parties is honest).

In this work we focus on MPC protocols tolerating any number of malicious corruptions. Most existing implementations of MPC protocols in this adversarial model rely on some variant of the secret-sharing paradigm introduced by Goldreich, Micali, and Wigderson [GMW87]. At a high level, this technique requires the parties to maintain the invariant of holding a linear secret sharing of the values on the wires of the circuit, along with some sort of authentication information on those shares. Linear gates in the circuit (e.g., XOR,

“Are there other paradigms for secure computation?”



Fully Homomorphic Encryption without Bootstrapping

Zvika Brakerski
Weizmann Institute of Science

Craig Gentry[*]
IBM T.J. Watson Research Center

Vinod Vaikuntanathan[†]
University of Toronto

Abstract

We present a radically new approach to fully homomorphic encryption (FHE) that dramatically improves performance and bases security on weaker assumptions. A central conceptual contribution in our work is a new way of constructing leveled fully homomorphic encryption schemes (capable of evaluating arbitrary polynomial-size circuits), *without Gentry's bootstrapping procedure*.

Specifically, we offer a choice of FHE schemes based on the learning with error (LWE) or ring-LWE (RLWE) problems that have 2^λ security against known attacks. For RLWE, we have:

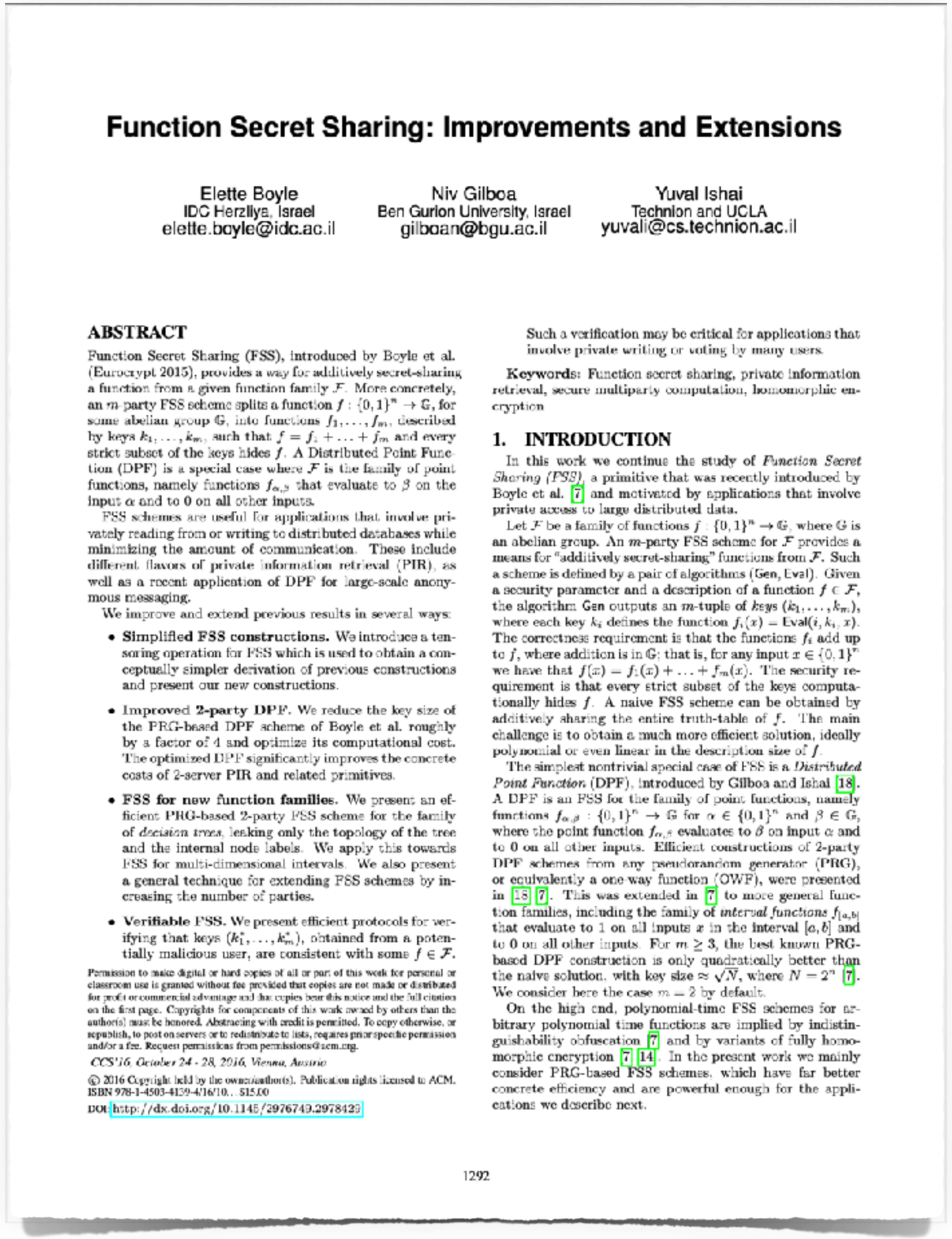
- A leveled FHE scheme that can evaluate L-level arithmetic circuits with $\tilde{O}(\lambda \cdot L^3)$ per-gate computation – i.e., computation *quasi-linear* in the security parameter. Security is based on RLWE for an approximation factor exponential in L. This construction does not use the bootstrapping procedure.
- A leveled FHE scheme that uses bootstrapping *as an optimization*, where the per-gate computation (which includes the bootstrapping procedure) is $\tilde{O}(\lambda^2)$, *independent of L*. Security is based on the hardness of RLWE for *quasi-polynomial* factors (as opposed to the sub-exponential factors needed in previous schemes).

We obtain similar results for LWE, but with worse performance. We introduce a number of further optimizations to our schemes. As an example, for circuits of large width – e.g., where a constant fraction of levels have width at least λ – we can reduce the per-gate computation of the bootstrapped version to $\tilde{O}(\lambda)$, independent of L, by *batching the bootstrapping operation*. Previous FHE schemes all required $\tilde{\Omega}(\lambda^{3.5})$ computation per gate.

At the core of our construction is a much more effective approach for managing the noise level of lattice-based ciphertexts as homomorphic operations are performed, using some new techniques recently introduced by Brakerski and Vaikuntanathan (FOCS 2011).

[*] Sponsored by the Air Force Research Laboratory (AFRL). Disclaimer: This material is based on research sponsored by DARPA under agreement number FA8750-11-C-0096 and FA8750-11-2-0225. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government. Approved for Public Release, Distribution Unlimited.

[†] This material is based on research sponsored by DARPA under Agreement number FA8750-11-2-0225. All disclaimers as above apply.



Function Secret Sharing: Improvements and Extensions

Elette Boyle
IDC Herzliya, Israel
elette.boyle@idc.ac.il

Niv Gilboa
Ben Gurion University, Israel
gilboan@bgu.ac.il

Yuval Ishai
Technion and UCLA
yuvali@cs.technion.ac.il

ABSTRACT

Function Secret Sharing (FSS), introduced by Boyle et al. (Eurocrypt 2015), provides a way for additively secret-sharing a function from a given function family $\mathcal{F}$. More concretely, an m-party FSS scheme splits a function $f : \{0,1\}^n \to \mathbb{G}$, for some abelian group $\mathbb{G}$, into functions $f_1, \ldots, f_m$, described by keys $k_1, \ldots, k_m$, such that $f = f_1 + \ldots + f_m$ and every strict subset of the keys hides f. A Distributed Point Function (DPF) is a special case where $\mathcal{F}$ is the family of point functions, namely functions $f_{\alpha,\beta}$ that evaluate to β on the input α and to 0 on all other inputs.

FSS schemes are useful for applications that involve privately reading from or writing to distributed databases while minimizing the amount of communication. These include different flavors of private information retrieval (PIR), as well as a recent application of DPF for large-scale anonymous messaging.

We improve and extend previous results in several ways:

- **Simplified FSS constructions.** We introduce a tensoring operation for FSS which is used to obtain a conceptually simpler derivation of previous constructions and present our new constructions.
- **Improved 2-party DPF.** We reduce the key size of the PRG-based DPF scheme of Boyle et al. roughly by a factor of 4 and optimize its computational cost. The optimized DPF significantly improves the concrete costs of 2-server PIR and related primitives.
- **FSS for new function families.** We present an efficient PRG-based 2-party FSS scheme for the family of *decision trees*, leaking only the topology of the tree and the internal node labels. We apply this towards FSS for multi-dimensional intervals. We also present a general technique for extending FSS schemes by increasing the number of parties.
- **Verifiable FSS.** We present efficient protocols for verifying that keys $(k_1^*, \ldots, k_m^*)$, obtained from a potentially malicious user, are consistent with some $f \in \mathcal{F}$. Such a verification may be critical for applications that involve private writing or voting by many users.

Keywords: Function secret sharing, private information retrieval, secure multiparty computation, homomorphic encryption

1. INTRODUCTION

In this work we continue the study of *Function Secret Sharing (FSS)*, a primitive that was recently introduced by Boyle et al. [7] and motivated by applications that involve private access to large distributed data.

Let $\mathcal{F}$ be a family of functions $f : \{0,1\}^n \to \mathbb{G}$, where $\mathbb{G}$ is an abelian group. An m-party FSS scheme for $\mathcal{F}$ provides a means for "additively secret-sharing" functions from $\mathcal{F}$. Such a scheme is defined by a pair of algorithms (Gen, Eval). Given a security parameter and a description of a function $f \in \mathcal{F}$, the algorithm Gen outputs an m-tuple of keys $(k_1, \ldots, k_m)$, where each key k_i defines the function $f_i(x) = \mathsf{Eval}(i, k_i, x)$. The correctness requirement is that the functions f_i add up to f, where addition is in $\mathbb{G}$: that is, for any input $x \in \{0,1\}^n$ we have that $f(x) = f_1(x) + \ldots + f_m(x)$. The security requirement is that every strict subset of the keys computationally hides f. A naive FSS scheme can be obtained by additively sharing the entire truth-table of f. The main challenge is to obtain a much more efficient solution, ideally polynomial or even linear in the description size of f.

The simplest nontrivial special case of FSS is a *Distributed Point Function* (DPF), introduced by Gilboa and Ishai [18]. A DPF is an FSS for the family of point functions, namely functions $f_{\alpha,\beta} : \{0,1\}^n \to \mathbb{G}$ for $\alpha \in \{0,1\}^n$ and $\beta \in \mathbb{G}$, where the point function $f_{\alpha,\beta}$ evaluates to β on input α and to 0 on all other inputs. Efficient constructions of 2-party DPF schemes from any pseudorandom generator (PRG), or equivalently a one-way function (OWF), were presented in [18, 7]. This was extended in [7] to more general function families, including the family of *interval functions* $f_{[a,b]}$ that evaluate to 1 on all inputs x in the interval $[a, b]$ and to 0 on all other inputs. For $m \geq 3$, the best known PRG-based DPF construction is only quadratically better than the naive solution, with key size $\approx \sqrt{N}$, where $N = 2^n$ [7]. We consider here the case $m = 2$ by default.

On the high end, polynomial-time FSS schemes for arbitrary polynomial time functions are implied by indistinguishability obfuscation [7] and by variants of fully homomorphic encryption [7, 14]. In the present work we mainly consider PRG-based FSS schemes, which have far better concrete efficiency and are powerful enough for the applications we describe next.

CCS'16, October 24 - 28, 2016, Vienna, Austria

© 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ISBN 978-1-4503-4139-4/16/10. . . $15.00

DOI: http://dx.doi.org/10.1145/2976749.2978429

“Are there special-case problems with interesting solutions?”

Efficient Batched Oblivious PRF with Applications to Private Set Intersection

Private Information Retrieval

BENNY CHOR

Technion, Haifa, Israel

ODED GOLDREICH

Weizmann Institute of Science, Rehovot, Israel

EYAL KUSHILEVITZ

Technion, Haifa, Israel

AND

MADHU SUDAN

Massachusetts Institute of Technology, Cambridge, Massachusetts

Abstract. Publicly accessible databases are an indispensable resource for retrieving up-to-date information. But they also pose a significant risk to the privacy of the user, since a curious database operator can follow the user’s queries and infer what the user is after. Indeed, in cases where the users' intentions are to be kept secret, users are often cautious about accessing the database. It can be shown that when accessing a single database, to completely guarantee the privacy of the user, the whole database should be down-loaded; namely n bits should be communicated (where n is the number of bits in the database).

In this work, we investigate whether by replicating the database, more efficient solutions to the private retrieval problem can be obtained. We describe schemes that enable a user to access k replicated copies of a database ($k \geq 2$) and *privately* retrieve information stored in the database. This means that each individual server (holding a replicated copy of the database) gets no information on the identity of the item retrieved by the user. Our schemes use the replication to gain substantial saving. In particular, we present a two-server scheme with communication complexity $O(n^{1/3})$.

A preliminary version of this paper appeared in *Proceedings of the 36th Annual IEEE Conference on Foundations of Computer Science* (Oct.). IEEE, New York, 1995, pp. 41–50.

Some of this work was done while M. Sudan was at IBM.

The work of O. Goldreich was supported by grant No. 92-00226 from the Israel-US Binational Science Foundation (BSF), Jerusalem, Israel.

Authors’ addresses: B. Chor, Computer Science Dept., Technion, Haifa, Israel, e-mail: {benny, eyalk}@cs.technion.ac.il, internet: http://www.cs.technion.ac.il/~eyalk; O. Goldreich, Computer Science and Applied Mathematics Dept., Weizmann Institute of Science, Rehovot, Israel, e-mail: oded@wisdom.weizmann.ac.il; M. Sudan, Laboratory for Computer Science, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139, e-mail: madhu@theory.lcs.mit.edu.

Extending Oblivious Transfers Efficiently

Yuval Ishai[1], Joe Kilian[2], Kobbi Nissim[2*], and Erez Petrank[1**]

Efficient Pseudorandom Correlation Generators: Silent OT Extension and More*

Elette Boyle[1], Geoffroy Couteau[2], Niv Gilboa[3], Yuval Ishai[4], Lisa Kohl[2], and Peter Scholl[5]

SoftSpokenOT: Communication–Computation Tradeoffs in OT Extension

Lawrence Roy*

February 17, 2022

Ferret: Fast Extension for coRRElated oT with small communication

Kang Yang
State Key Laboratory of Cryptology
yangk@sklc.org

Chenkai Weng
Northwestern University
ckweng@u.northwestern.edu

Xiao La
Sichuan Univ
lanxiao@scu.

Jiang Zhang
State Key Laboratory of Cryptology
jiangzhang09@gmail.com

Xiao Wang
Northwestern University
wangxiao@cs.northwestern.edu

September 6, 2020

Abstract

Correlated oblivious transfer (COT) is a crucial building block for secure multi-party computation (MPC) and can be generated efficiently via OT extension. Recent works based on the pseudorandom correlation generator (PCG) paradigm presented a new way to generate random COT correlations using only communication sublinear to the output length. However, due to their high computational complexity, these protocols are only faster than the classical IKNP-style OT extension under restricted network bandwidth.

SecureML: A System for Scalable Privacy-Preserving Machine Learning

PAYMAN MOHASSEL*   YUPENG ZHANG†

Abstract

Machine learning is widely used in practice to produce predictive models for applications such as image processing, speech and text recognition. These models are more accurate when trained on large amount of data collected from different sources. However, the massive data collection raises privacy concerns.

In this paper, we present new and efficient protocols for privacy preserving machine learning for linear regression, logistic regression and neural network training using the stochastic gradient descent method. Our protocols fall in the two-server model where data owners distribute their private data among two non-colluding servers who train various models on the joint data using secure two-party computation (2PC). We develop new techniques to support secure arithmetic operations on shared decimal numbers, and propose MPC-friendly alternatives to non-linear functions such as sigmoid and softmax that are superior to prior work.

We implement our system in C++. Our experiments validate that our protocols are several orders of magnitude faster than the state of the art implementations for privacy preserving linear and logistic regressions, and scale to millions of data samples with thousands of features. We also implement the first privacy preserving system for training neural networks.

1 Introduction

Machine learning techniques are widely used in practice to produce predictive models for use in medicine, banking, recommendation services, threat analysis, and authentication technologies. Large amount of data collected over time have enabled new solutions to old problems, and advances in deep learning have led to breakthroughs in speech, image and text recognition.

Large internet companies collect users’ online activities to train recommender systems that predict their future interest. Health data from different hospitals, and government organization can be used to produce new diagnostic models, while financial companies and payment networks can combine transaction history, merchant data, and account holder information to train more accurate fraud-detection engines.

While the recent technological advances enable more efficient storage, processing and computation on big data, *combining data from different sources* remains an important challenge. Competitive advantage, privacy concerns and regulations, and issues surrounding data sovereignty and jurisdiction prevent many organizations from openly sharing their data. Privacy-preserving machine learning via

*Visa Research. **Email:** pmohasse@visa.com.

†University of Maryland. **Email:** zhangyp@umd.edu. This work was partially done when the author was interning at Visa Research.

“MPC that goes beyond circuits?”

Secure Two-Party Computation in Sublinear (Amortized) Time

S. Dov Gordon
Columbia University
gordon@cs.columbia.edu

Jonathan Katz
University of Maryland
jkatz@cs.umd.edu

Vladimir Kole
Alcatel-Lucent B
kolesnikov@res
labs.com

Fernando Krell
Columbia University
fernando@cs.columbia.edu

Tal Malkin
Columbia University
tal@cs.columbia.edu

Mariana Raykova
Columbia University
mariana@cs.columbia.edu

Yevgeniy Vahlis
AT&T Security Research Center
evahlis@att.com

ABSTRACT

Traditional approaches to generic secure computation begin by representing the function f being computed as a circuit. If f depends on each of its input bits, this implies a protocol with complexity at least linear in the input size. In fact, linear running time is *inherent* for non-trivial functions since each party must “touch” every bit of their input lest information about the other party's input be leaked. This seems to rule out many applications of secure computation (e.g., database search) in scenarios where inputs are huge.

Adapting and extending an idea of Ostrovsky and Shoup, we present an approach to secure two-party computation that yields protocols running in *sublinear* time, in an amortized sense, for functions that can be computed in sublinear time on a random-access machine (RAM). Moreover, each party is required to maintain state that is only (essentially) linear in its own input size. Our approach combines generic secure two-party computation with *oblivious RAM* (ORAM) protocols. We present an optimized version of our approach using Yao's garbled-circuit protocol and a recent ORAM construction of Shi et al.

We describe an implementation of our resulting protocol, and evaluate its performance for obliviously searching a database with over 1 million entries. Our implementation outperforms off-the-shelf secure-computation protocols for databases containing more than 2^{18} entries.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—security and protection

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CCS'12, October 16–18, 2012, Raleigh, North Carolina, USA.
Copyright 2012 ACM 978-1-4503-1651-4/12/10 ...$15.00.

Keywords

Theory, Security, Cryptography, Secu

1. INTRODUCTION

Consider the task of searching over
items. Using binary search, this can be
Now consider a secure version of thi
wishes to learn whether an item is i
a server, with neither party learning
plying generic secure computation [2
would begin by expressing the compu
arithmetic) circuit of size at least n,
of complexity Ω(n). Moreover, (at
ity is *inherent*: in any secure protoco
server must “touch” every entry of th
the server learns information about th
serving which entries of its database
This linear lower bound seems to rul
ever performing practical secure comp
datasets. However, tracing the sourc
one may notice two opportunities for

- Many interesting functions (such
be computed in *sublinear* time o
chine (RAM). Thus, it would b
cols for generic secure computat
rather than circuits — as their r
- The fact that linear work is inhe
tation of any non-trivial functio
f is computed *once*. However, it
possibility of doing better, in an
the parties compute the same fu

Inspired by the above, we explore s
computation with *sublinear* amortized
focus on a setting where a client and s
uate a function f, maintaining state a
with the server's (huge) input D cha
tween executions, and the client's (s
anew each time f is evaluated. (It
mind the concrete application of a

513

Path ORAM: An Extremely Simple Oblivious RAM Protocol

18

EMIL STEFANOV, UC Berkeley
MARTEN VAN DIJK, University of Connecticut
ELAINE SHI, Cornell University
T.-H. HUBERT CHAN, University of Hong Kong
CHRISTOPHER FLETCHER, University of Illinois at Urbana-Champaign
LING REN, XIANGYAO YU, and SRINIVAS DEVADAS, MIT CSAIL

We present
Partly due t
storage. We
hits. For su
small client
since its pro

Categories :
curity and I

General Ter

Additional I

ACM Refe
Emil Stefan
and Srinivas
18 (April 20
https://doi.o

A conference
2013.
This work is
DoD NDSEG
zon Web Ser
material are t
The research
Authors' add
USA: email:
cut, Storrs M
University, It
University of
ment, Univer
Devadas, MIT
Permission t
provided tha
the full citati
Abstracting v
prior specific

Circuit ORAM: On Tightness of the Goldreich-Ostrovsky Lower Bound

Xiao Shaun Wang
wangxiao@cs.umd.edu
University of Maryland

T-H. Hubert Chan
hubert@cs.hku.hk
University of Hong Kong

Elaine Shi
runting@gmail.com
University of Maryland

November 27, 2016

1

Obliv
crypt
durin
it has
outso

1.1

When
its b

Tri-State Circuits
A Circuit Model that Captures RAM*

David Heath[1], Vladimir Kolesnikov[2], and Rafail Ostrovsky[3]

[1] daheath@illinois.edu, UIUC
[2] kolesnikov@gatech.edu, Georgia Tech
[3] rafail@cs.ucla.edu, UCLA

Abstract. We introduce *tri-state circuits* (TSCs). TSCs form a natural model of computation that, to our knowledge, has not been considered by theorists. The model captures a surprising combination of simplicity and power. TSCs are simple in that they allow only three wire values (0, 1, and undefined – $\mathcal{Z}$) and three types of fan-in two gates; they are powerful in that their statically placed gates fire (execute) eagerly as their inputs become defined, implying orders of execution that depend on input. This behavior is sufficient to efficiently evaluate RAM programs.

We construct a TSC that emulates T steps of any RAM program and that has only $O(T \cdot \log^3 T \cdot \log\log T)$ gates. Contrast this with the reduction from RAM to Boolean circuits, where the best approach scans all of memory on each access, incurring quadratic cost.

We connect TSCs with cryptography by using them to improve Yao's Garbled Circuit (GC) technique. TSCs capture the power of garbling far better than Boolean Circuits, offering a more expressive model of computation that leaves per-gate cost essentially unchanged.

As an important application, we construct *authenticated Garbled* RAM (GRAM), enabling constant-round maliciously-secure 2PC of RAM programs. Let λ denote the security parameter. We extend authenticated garbling to TSCs; by simply plugging in our TSC-based RAM, we obtain authenticated GRAM running at cost $O(T \cdot \log^3 T \cdot \log\log T \cdot \lambda)$, outperforming all prior work, including prior semi-honest GRAM.

We also give semi-honest garbling of TSCs from a one-way function (OWF). This yields OWF-based GRAM at cost $O(T \cdot \log^3 T \cdot \log\log T \cdot \lambda)$,
F-based GRAM by more than factor λ.

Random Access Machines, Circuits, Obliv-
AM.

implest complete model of computation.
gate types, each of which computes a basic

ngs of IACR Crypto 2023.

OptORAMa: Optimal Oblivious RAM*

Gilad Asharov
Bar-Ilan University

Ilan Komargodski
NTT Research and
Hebrew University

Wei-Kai Lin
Cornell University

Kartik Nayak
VMware and Duke University

Enoch Peserico
Univ. Padova

Elaine Shi
Cornell University

November 18, 2020

“More efficient approaches to Zero Knowledge?”

Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures

Jonathan Katz
University of Maryland
jkatz@cs.umd.edu

Vladimir Kolesnikov
Georgia Tech
kolesnikov@gatech.edu

Xiao Wang
University of Maryland
wangxiao@cs.umd.edu

Ligero: Lightweight Sublinear Arguments Without a Trusted Setup

Scott Ames
University of Rochester
sames@cs.rochester.edu

Carmit Hazay
Bar-Ilan University
carmit.hazay@cs.biu.ac.il

Yuval Ishai
Technion and UCLA
yuvali@cs.technion.ac.il

Muthuramakrishnan Venkitasubramaniam
University of Rochester
muthuv@cs.rochester.edu

ABSTRACT

We design and implement a simple zero-knowledge argument protocol for NP whose communication complexity is proportional to the square-root of the verification circuit size. The protocol can be based on any collision-resistant hash function. Alternatively, it can be made non-interactive in the random oracle model, yielding concretely efficient zk-SNARKs that do not require a trusted setup or public-key cryptography.

Our protocol is attractive not only for very large verification circuits but also for moderately large circuits that arise in applications. For instance, for verifying a SHA-256 preimage in zero-knowledge with 2^{-40} soundness error, the communication complexity is roughly 44KB (or less than 34KB under a plausible conjecture), the prover running time is 140 ms, and the verifier running time is 62 ms. This proof is roughly 4 times shorter than a similar proof of ZKB++ (Chase et al., CCS 2017), an optimized variant of ZKBoo (Giacomelli et al., USENIX 2016).

The communication complexity of our protocol is independent of the circuit structure and depends only on the number of gates. For 2^{-40} soundness error, the communication becomes smaller than the circuit size for circuits containing roughly 3 million gates or more. Our efficiency advantages become even bigger in an amortized setting, where several instances need to be proven simultaneously.

Our zero-knowledge protocol is obtained by applying an optimized version of the general transformation of Ishai et al. (STOC 2007) to a variant of the protocol for secure multiparty computation of Damgård and Ishai (Crypto 2006). It can be viewed as a simple zero-knowledge interactive PCP based on "interleaved" Reed-Solomon codes.

1 INTRODUCTION

Verifying outsourced computations is important for tasks and scenarios when there is an incentive for the party performing the computation to report incorrect answers. In this work, we present a concretely efficient argument protocol for NP whose communication complexity is proportional to the square root of the size of a circuit verifying the NP witness. Our argument system is in fact a zero-knowledge argument of knowledge, and it only requires the verifier to send public coins to the prover. The latter feature implies that it can be made non-interactive via the Fiat-Shamir transform [19], yielding an efficient implementation of zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARKs [11]) without a trusted setup.

To put our work in the proper context, we give some relevant background. The last half decade has seen tremendous progress in designing and implementing efficient systems for verifiable computation (see [4, 47] for recent surveys). These efforts can be divided into three broad categories according to the underlying combinatorial machinery.

Doubly efficient interactive proofs: This line of work, initiated by Goldwasser, Kalai, and Rothblum [23] (following a rich line of work on interactive proofs with computationally unbounded provers [24, 37, 44]), provides sublinear communication, efficiently verifiable proofs for low-depth polynomial-time computations.[1] See [15, 41, 45, 46] and references therein for a survey of works along this line.

Probabilistically checkable proofs (PCPs) and their interactive variants: Originating from the works of Kilian [36] and Micali [39], recent works [4, 6, 8] have shown how to obtain efficient sublinear arguments for NP from PCPs [1–3]. Classical PCPs have been extended to allow additional interaction with the prover, first in the model of interactive PCP (IPCP) [35] and then in the more general setting of interactive oracle proofs (IOP) [9], also known as probabilistically checkable interactive proofs (PCIP) [41]. Arguments obtained via PCPs and IOPs have the advantages of not relying on public-key cryptography, not requiring a trusted setup, and offering conjectured security against quantum attacks. However, current implementations along this line are still quite far from

Wolverine: Fast, Scalable, and Communication-Efficient Zero-Knowledge Proofs for Boolean and Arithmetic Circuits

Chenkai Weng
Northwestern University

Kang Yang*
State Key Laboratory of Cryptology

Jonathan Katz[†]
University of Maryland

Xiao Wang*
Northwestern University

2021 IEEE Symposium on Security and Privacy (SP) | 978-1-7281-8934-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/SP40001.2021.00056

QuickSilver: Efficient and Affordable Zero-Knowledge Proofs for Circuits and Polynomials over Any Field

Kang Yang
State Key Laboratory of Cryptology
yangk@sklc.org

Pratik Sarkar
Boston University
pratik93@bu.edu

Chenkai Weng
Northwestern University
ckweng@u.northwestern.edu

Xiao Wang
Northwestern University
wangxiao@cs.northwestern.edu

Mac'n'Cheese: Zero-Knowledge Proofs for Boolean and Arithmetic Circuits with Nested Disjunctions

Carsten Baum
Aarhus University

Alex J. Malozemoff
Galois, Inc.

Marc B. Rosen
Galois, Inc.

Peter Scholl
Aarhus University

Abstract

Zero knowledge proofs are an important building block in many cryptographic applications. Unfortunately, when the proof statements become very large, existing zero-knowledge proof systems easily reach their limits: either the computational overhead, the memory footprint, or the required bandwidth exceed levels that would be tolerable in practice.

We present an interactive zero-knowledge proof system for boolean and arithmetic circuits, called Mac′n′Cheese, with a focus on supporting large circuits. Our work follows the commit-and-prove paradigm instantiated using information-theoretic MACs based on vector oblivious linear evaluation to achieve high efficiency. We additionally show how to optimize disjunctions, with a general OR transformation for proving the disjunction of m statements that has communication complexity proportional to the longest statement (plus an additive term logarithmic in m). These disjunctions can further be *nested*, allowing efficient proofs about complex statements with many levels of disjunctions. We also show how to make Mac′n′Cheese non-interactive (after

“What is the state-of-the-art in garbled circuits?”

Three Halves Make a Whole? Beating the Half-Gates Lower Bound for Garbled Circuits

Mike Rosulek(✉) and Lawrence Roy

Oregon State University, Corvallis, USA
{rosulekm,royl}@oregonstate.edu

Abstrac
XOR ga
bits. Th
Zahur, R
free and
bound o
garbling
novel tec
wire labe
first to b
XOR, ma
proven s
cular cor
than hal

1 Introdu

Garbled circui
the first secure
nique for const
extremely effic
of the circuit),
in realistic dep
reason, most i
their concrete
rent state of th
of Zahur, Rosu
garbled with si
The half-ga
circuits. Specif
tured all know

First author pa
by a DoE CSGI
© International A
T. Malkin and C.
https://doi.org/10

Stacked Garbling
Garbled Circuit Proportional to Longest Execution Path

David Heath and Vladimir Kolesnikov

Georgia Institute of Technology, Atlanta, GA, USA
{heath.davidanthony,kolesnikov}@gatech.edu

1 In

Secure
compu
output
wide a
of Yao

Efficient Arithmetic in Garbled Circuits*

David Heath
University of Illinois Urbana-Champaign
daheath@illinois.edu

Abstract

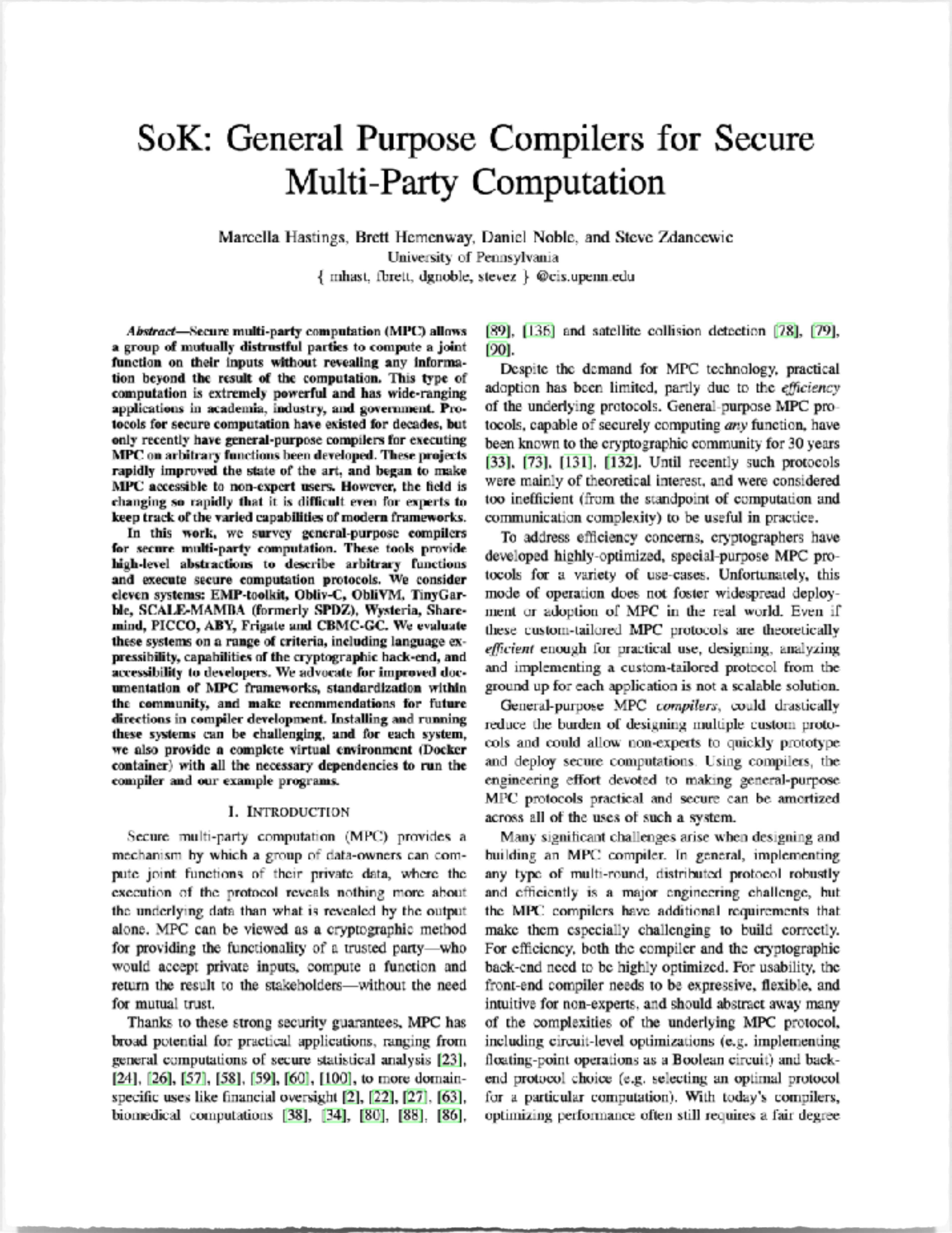
Garbled Circuit (GC) techniques usually work with Boolean circuits. Despite intense interest, efficient arithmetic generalizations of GC were only known from heavy assumptions, such as LWE.

We construct arithmetic garbled circuits from circular correlation robust hashes, the assumption underlying the celebrated Free XOR garbling technique. Let λ denote a computational security parameter, and consider the integers $\mathbb{Z}_m$ for any $m \geq 2$. Let $\ell = \lceil \log_2 m \rceil$ be the bit length of $\mathbb{Z}_m$ values. We garble arithmetic circuits over $\mathbb{Z}_m$ where the garbling of each gate has size $O(\ell \cdot \lambda)$ bits. Constrast this with Boolean-circuit-based arithmetic, requiring $O(\ell^2 \cdot \lambda)$ bits via the schoolbook multiplication algorithm, or $O(\ell^{1.585} \cdot \lambda)$ bits via Karatsuba's algorithm.

Our arithmetic gates are compatible with Boolean operations and with Garbled RAM, allowing to garble complex programs of arithmetic values.

Keywords— Secure Multiparty Computation, Garbled Circuits, Arithmetic Circuits

“What tools exist for using MPC?”



SoK: General Purpose Compilers for Secure Multi-Party Computation

Marcella Hastings, Brett Hemenway, Daniel Noble, and Steve Zdancewic
University of Pennsylvania
{ mhast, fbrett, dgnoble, stevez } @cis.upenn.edu

***Abstract*—Secure multi-party computation (MPC) allows a group of mutually distrustful parties to compute a joint function on their inputs without revealing any information beyond the result of the computation. This type of computation is extremely powerful and has wide-ranging applications in academia, industry, and government. Protocols for secure computation have existed for decades, but only recently have general-purpose compilers for executing MPC on arbitrary functions been developed. These projects rapidly improved the state of the art, and began to make MPC accessible to non-expert users. However, the field is changing so rapidly that it is difficult even for experts to keep track of the varied capabilities of modern frameworks.**

In this work, we survey general-purpose compilers for secure multi-party computation. These tools provide high-level abstractions to describe arbitrary functions and execute secure computation protocols. We consider eleven systems: EMP-toolkit, Obliv-C, ObliVM, TinyGarble, SCALE-MAMBA (formerly SPDZ), Wysteria, Sharemind, PICCO, ABY, Frigate and CBMC-GC. We evaluate these systems on a range of criteria, including language expressibility, capabilities of the cryptographic back-end, and accessibility to developers. We advocate for improved documentation of MPC frameworks, standardization within the community, and make recommendations for future directions in compiler development. Installing and running these systems can be challenging, and for each system, we also provide a complete virtual environment (Docker container) with all the necessary dependencies to run the compiler and our example programs.

I. Introduction

Secure multi-party computation (MPC) provides a mechanism by which a group of data-owners can compute joint functions of their private data, where the execution of the protocol reveals nothing more about the underlying data than what is revealed by the output alone. MPC can be viewed as a cryptographic method for providing the functionality of a trusted party—who would accept private inputs, compute a function and return the result to the stakeholders—without the need for mutual trust.

Thanks to these strong security guarantees, MPC has broad potential for practical applications, ranging from general computations of secure statistical analysis [23], [24], [26], [57], [58], [59], [60], [100], to more domain-specific uses like financial oversight [2], [22], [27], [63], biomedical computations [38], [34], [80], [88], [86], [89], [136] and satellite collision detection [78], [79], [90].

Despite the demand for MPC technology, practical adoption has been limited, partly due to the *efficiency* of the underlying protocols. General-purpose MPC protocols, capable of securely computing *any* function, have been known to the cryptographic community for 30 years [33], [73], [131], [132]. Until recently such protocols were mainly of theoretical interest, and were considered too inefficient (from the standpoint of computation and communication complexity) to be useful in practice.

To address efficiency concerns, cryptographers have developed highly-optimized, special-purpose MPC protocols for a variety of use-cases. Unfortunately, this mode of operation does not foster widespread deployment or adoption of MPC in the real world. Even if these custom-tailored MPC protocols are theoretically *efficient* enough for practical use, designing, analyzing and implementing a custom-tailored protocol from the ground up for each application is not a scalable solution.

General-purpose MPC *compilers*, could drastically reduce the burden of designing multiple custom protocols and could allow non-experts to quickly prototype and deploy secure computations. Using compilers, the engineering effort devoted to making general-purpose MPC protocols practical and secure can be amortized across all of the uses of such a system.

Many significant challenges arise when designing and building an MPC compiler. In general, implementing any type of multi-round, distributed protocol robustly and efficiently is a major engineering challenge, but the MPC compilers have additional requirements that make them especially challenging to build correctly. For efficiency, both the compiler and the cryptographic back-end need to be highly optimized. For usability, the front-end compiler needs to be expressive, flexible, and intuitive for non-experts, and should abstract away many of the complexities of the underlying MPC protocol, including circuit-level optimizations (e.g. implementing floating-point operations as a Boolean circuit) and back-end protocol choice (e.g. selecting an optimal protocol for a particular computation). With today's compilers, optimizing performance often still requires a fair degree

Today’s objectives

Survey possible topics for your project

Form a team

Start looking through the literature