

SECURITY AND PRIVACY IN CLOUD COMPUTING

With the advent of cloud computing, services like Amazon EC2, Google Cloud, and Microsoft Azure offer easily accessible cloud servers that allow users to outsource their data storage and computation. Cloud computing alleviates users’ local storage and computational costs, and provides elasticity and geographical distribution of computing power. The rise of cloud computing also introduces new security challenges: as users give up control of their data and outsource their computation, it becomes more difficult to ensure both correctness (e.g., verifiability of the results returned) and privacy.

My research is focused on solving the above problems by designing efficient protocols and systems with rigorous guarantees of verifiability and privacy. My approach combines expertise in both cryptography and systems-building, and my works enjoy both provable security and good performance for practical applications. With regard to verifiability, I have built systems supporting verifiable queries over graph, verifiable SQL queries on structured databases, and verifiable execution of arbitrary RAM programs. For privacy, I have built a privacy-preserving machine-learning system allowing users to encrypt and delegate their data to two cloud servers who can then run machine-learning algorithms on the encrypted data without learning anything about the underlying content. I have also explored security issues that arise in the context of searchable encryption schemes, which enhance users’ privacy by allowing cloud servers to perform searches over encrypted files without learning the search keywords and file contents.

These two lines of my research have broader impacts in other applications. For example, verifiable computation with an additional zero-knowledge property can be applied to cryptocurrencies and smart contracts to provide privacy for transactions and computations done on a public blockchain; privacy-preserving computation can also be used to enable multiple companies or hospitals to jointly perform computations (e.g., machine learning) on their collective data without disclosing their individual data to each other. In the following, I summarize my research so far and discuss my future research plans.

Verifiable Computation

Most of my work falls in the area of verifiable outsourcing of storage and computation. Here, a user can delegate her data and computation to an untrusted server while still being able to validate the integrity of the results returned by the server. Below I describe some of my work in this area.

Verifiable Graph Algorithms. In one of my first projects [6], I built a system supporting verifiable graph algorithms. Here, a data owner can upload a graph to a cloud server, and users can then make queries on the graph (e.g., asking for the shortest path between two nodes) and validate that the results returned are correctly computed on the graph outsourced by the data owner. The proofs for correctness are short (in particular, shorter than the representation of the graph itself) and efficient to verify; the additional storage at the server is also low. We tested the system on a street map of the city map of Rome, and found that the overhead (compared to the setting with no guarantees) is only a few milliseconds. Our system is 8 orders of magnitude faster than the prior best work, and brings verifiable graph algorithms one step closer to being practical.

Verifiable Databases. Databases-as-a-Service allow users to host a database on a cloud server, and later issue SQL queries to that database. I have built systems that provide efficient verifiability for such queries,
thus allowing users to validate the correctness of the results returned by the server while storing only a short digest of their original database.

The first system I built, called IntegriDB [5], supports a rich subset of expressive SQL queries including multi-dimensional range queries, join queries, aggregation functions, and nested queries. (Prior work was only able to support simple queries, and did not allow nesting. Note that nesting is extremely common in regular usage of SQL.) IntegriDB was able to successfully execute on complicated queries and large databases from the TPC-H benchmark that is standard in the databases community. The client-side overhead in our system was less than one second, and the server-side overhead was orders of magnitude lower than in prior work.

In subsequent work [2], I took an entirely different approach to develop a new system called vSQL that improves on IntegriDB in several ways: it is more expressive (supporting arbitrary SQL queries), handles updates (i.e., insertions and deletions), and is also more efficient than IntegriDB in many cases. The core of vSQL is a new cryptographic protocol I designed for verifiable computation of arbitrary functions on outsourced data. This protocol combined two cryptographic primitives—interactive proofs and verifiable polynomial delegation—to get the best properties of each. Compared to existing approaches for general verifiable computation, which are mainly based on succinct non-interactive arguments of knowledge (SNARKs), my scheme reduces the number of expensive cryptographic operations (e.g., modular exponentiations) to linear to the size of the input, rather than linear to the size of the circuit describing the function being computed. Our scheme is up to two orders of magnitude faster than SNARKs, and reduces the memory consumption of the server by around 99%.

Verifiable RAM Programs. Most prior work for generic verifiable computation, including vSQL, models the function being verified as a circuit. Building on my work described above, I recently developed an efficient scheme for verifying arbitrary computations expressed as RAM programs [1]. This can lead to a significant improvement in some cases; for example, the size of a circuit implementing binary search on a sorted array is linear in the length of the array, whereas the complexity of a RAM program for binary search is only logarithmic. The scheme I developed is around $9-30 \times$ faster than prior work, and scales to support programs running for up to 2 million CPU cycles, $65 \times$ more than the best prior record.

Private Computation

A second line of my research has focused on privacy-preserving (secure) computation. The goal here is to allow users who outsource their data to the cloud to ensure the privacy of their data while still allowing the cloud servers to perform certain computations over the data. In the following, I highlight some of the work I have done in this area.

Privacy-Preserving Machine Learning Companies such as Google, Microsoft, and Amazon offer personalized products and services that are constructed by training machine-learning models over large amounts of user data. Guaranteeing confidentiality of users’ data while still allowing meaningful models to be derived is still a significant challenge.

To address this problem, I designed a system for privacy-preserving machine learning [3]. The system allows users to encrypt and upload their data to two servers and enables the servers to run machine-learning algorithms over users’ data without learning anything about the underlying data itself. In this way, the servers can still derive meaningful models and use them for providing personalized services while the privacy of users’ data is protected. The system I developed supports privacy-preserving linear regression, logistic regression, and deep learning using neural networks. It is up to $1100 \times$ faster than prior systems for linear regression, and is the first to support privacy-preserving logistic regression and neural networks. The system scales to run machine-learning algorithms on millions of data records with thousands of features each.
Searchable Encryption  Searchable encryption allows a server to perform keyword search on encrypted files without learning the keywords being searched for or the contents of the files. There are numerous prior works improving the efficiency of searchable encryption schemes, and the overhead of encrypted search is acceptable in many practical applications. However, this efficiency comes at the price of allowing certain leakage of information to the server (e.g., the number of files matching a given query), of which the practical consequences are not yet well understood.

In one work [4], I explored generic attacks that could be carried out against arbitrary searchable-encryption schemes as long as they leaked certain information (which is leaked by almost all schemes in the literature). The attacks I proposed allow the server to learn the exact keyword the user is searching for. Interestingly, the attack I proposed is an active attack that involves the server injecting files into the users’ file collection (something that can be easily done, e.g., if the files represent incoming emails). I showed that in practice, a server would only need to inject around 10 files and would then be able to learn all keywords being searched for by the user. The attacks have significantly higher success rate compared to prior work and can be generalized to searchable-encryption schemes supporting searches with multiple keywords. Besides being an important step forward in terms of understanding the practical effects of leakage, my work also led to follow-up works constructing searchable-encryption schemes that specifically avoided the leakage that made my attack possible.

Future Directions

My future research will mainly lie in the intersection of cryptography and system security, with a particular focus on computations with privacy and verifiability. Though shown to be powerful theoretically, existing cryptographic tools for verifiable and privacy-preserving computation usually incur dramatic overhead on the efficiency and are not practical yet. I would like to seek new theoretical refinement to improve their performance, as well as develop concrete optimizations tailored for real-world applications and different practical settings. I also believe security and privacy issues are major concerns in the development of new technologies such as blockchain and machine learning, and cryptographic tools can play an important role to address these problems. I plan to integrate verifiable and privacy-preserving computation to these newly developed techniques to provide privacy and verifiability guarantees, which will allow more people to enjoy the benefits of these technologies. In the following, I highlight two particular ongoing projects that are in alignment of my long term goals.

Privacy-preserving Smart Contracts. Verifiable computation with an additional zero-knowledge property can also be applied to privacy-preserving cryptocurrencies and smart contracts. In existing systems like Bitcoin and Ethereum, information such as the amount of money in a transaction and the inputs to a smart contract are all published on the blockchain in the clear, so that users can check the validity of transactions and contracts (e.g., the money in a transaction belongs to the sender, there is no double-spending, the contract is executed correctly and the total amount of money sent and received are consistent.), which forms the core of the decentralized blockchain technique. However, these everlasting public information on the blockchain also raises the privacy concerns. Several attacks have shown that it is possible to link the history of a user’s transactions and de-anonymize the user. Zero-knowledge verifiable computation can be applied to solve this problem. Instead of publishing all the information as before, users can now generate efficiently verifiable proofs that the transactions and contracts are executed according to the rules; all other participants can still check the validity of the transactions and contracts by verifying the correctness of these proofs, without learning anything about the underlying information, which is guaranteed by the zero-knowledge property. This idea is already in widespread use (for transactions) in Zcash.

A serious drawback of existing verifiable-computation schemes in this context is that in order to generate and validate the proofs, users need to access public parameters published by a trusted party on a per-contract
basis, which violates the original goal of decentralization of the blockchain technique. A dishonest party who publishes these parameters could potentially subvert the entire system. Because of this problem, Zcash only supports verifying basic transaction operations and cannot be generalized to support expressive smart contracts. To address this problem, I plan to construct a scheme based on my general-purpose verifiable-computation scheme that does not rely on public parameters at all. As a first step [7], I have shown how to adapt my scheme to be zero knowledge. With this scheme, we can eliminate the trusted party involved in every contract and potentially improve the efficiency and scalability of the original scheme, as shown in my prior work. This will eventually lead to a privacy-preserving smart contract for users with privacy concerns such as financial institutions to deploy blockchain techniques.

**Privacy-preserving Machine Learning and Differential Privacy.** I also plan to continue my work on privacy-preserving machine learning. In particular, I would like to design an efficient protocol that allows multiple participants to jointly train a machine-learning model on all of their data, without sharing their data with each other. In addition, I plan to build on my prior work and integrate support for secure computation of other machine-learning algorithms, such as decision trees and SVMs.

An orthogonal line of work in privacy-preserving machine learning involves differential privacy. This technique is concerned not with information leakage during the computation of the machine-learning model, but rather with minimizing the information leaked by the eventual model itself. Very few works have considered both secure computation and differential privacy together, yet the techniques in these two areas can potentially improve not only the security guarantees but even the overall efficiency of the system. For example, if the data remains private during the training phase using secure computation, the noise level added to the final model could be reduced significantly, thus the accuracy of the differentially private model can be improved. I plan to explore possibilities along this line to combine techniques in both areas, so that not only the data remains confidential during the training, but the final model leaks minimal information about the training data as well.

**References**


