Integrity, Consistency, and Verification of Remote Computation

Christian Cachin
IBM Research - Zurich
cca@zurich.ibm.com

1. SUMMARY

With the advent of cloud computing, many clients have outsourced computation and data storage to remote servers. This has led to prominent concerns about the *privacy* of the data and computation placed outside the control of the clients. On the other hand, the *integrity* of the responses from the remote servers has been addressed in-depth only recently. Violations of correctness are potentially more dangerous, however, in the sense that the *safety* of a service is in danger and that the clients rely on the responses. Incidental computation errors as well as deliberate and sophisticated manipulations on the server side are nearly impossible to discover with today's technology. Over the last few years, there has been rising interest in technology to *verify* the results of a remote computation and to check the *consistency* of responses from a cloud service. These advances rely on recently introduced cryptographic techniques, including authenticated data types (ADT), probabilistically checkable proofs (PCPs), fully-homomorphic encryption (FHE), quadratic programs (QP), and more. With multiple clients accessing the remote service, a further dimension is added to the problem in the sense that clients isolated from each other need to guarantee that their verification operations relate to the same "version" of the server's computation state.

This tutorial will survey the recent work in this area and provide a broad introduction to some of the key concepts underlying verifiable computation, towards single and multiple verifiers. The aim is to give a systematic survey of techniques in the realm of verifiable computation, remote data integrity, authenticated queries, and consistency verification.

The approaches rely on methods from *cryptography* and from *distributed computing*. The presentation will introduce the necessary background techniques from these fields, describe key results, and illustrate how they ensure integrity in selected cases.

The tutorial consists of three parts:

1. Verifiable computation;
2. Authenticated data types;
3. Distributed consistency enforcement.

2. VERIFIABLE COMPUTATION

In this basic model, *one* client outsources a computation to a powerful remote, untrusted service. Verifiable computation protocols [6, 4] ensure that a weak client obtains a guarantee that the result of a computation by the server on inputs supplied by the client is correct. For this to make sense, the client should be able to verify the proof supplied by the server significantly faster than re-running the computation by itself.

Impressive progress has been made recently towards this goal. Starting with work in cryptography [6, 7], the technique of probabilistically checkable proofs (PCPs) has been optimized and scaled down, so as to achieve near-practical performance in certain cases [13, 14, 1]. More recently quadratic arithmetic programs (QPs) have been introduced as a representation of arithmetic circuits that compute the computation, together with an efficient method for cryptographically verifying the correctness of the outputs [5]. The powerful tool of Fully Homomorphic Encryption (FHE) has also been employed for verifying output correctness [4]. These lines of work have led to multiple nearly practical systems such as Pinocchio [12] and Pantry [1].

The tutorial surveys this work and provides an introduction to the methods underlying the systems based on QPs.

3. AUTHENTICATED DATA TYPES

Starting before the recent work on general verifiable computation, the notion of *authenticated data types (ADTs)* has been formulated to capture a *multi-party* model of interaction with a remote service. Restricted to *one writer*, which is the only client that may perform updates, an ADS permits *many readers* to query the remote service and to retrieve aggregate information about the service state held by the remote server. Starting from the ubiquitous Merkle tree, many different schemes for verifiable retrieval operations on particular data types have been formulated, such as sets, search trees, or skip lists [15, 9, 11].

The tutorial reviews the basic concept and some key results in the realm of authenticated data types.

4. DISTRIBUTED CONSISTENCY ENFORCEMENT

In parallel to the above developments, and departing from a single client that may perform updates, further work has considered a model where *multiple clients* interact by *writing to and reading from* a stateful remote service. Assuming the clients do not communicate with each other and in the absence of synchronization, a fundamental impossibility result prevents complete consistency among all clients. In particular, since the service may mount a replay attack and answer with responses from outdated state to a tar-

CCS'14, November 3–7, 2014, Scottsdale, Arizona, USA.
ACM 978-1-4503-2957-6/14/11.
http://dx.doi.org/10.1145/2660267.2660575.

get client, the victim cannot discover that more recent operations have been performed by other clients.

Starting with SUNDR [10, 8], several contributions in this context have improved the achievable consistency guarantees, explored the fundamental tradeoffs regarding the interaction between clients and the service, and reduced the implementation cost [3, 16, 2].

Recent protocols guarantee atomic operations to all clients when the service is correct and so-called *fork-linearizable semantics* when the service is faulty. Fork-linearizability makes it much easier for the clients to detect violations of integrity and consistency by the service; specifically, it means that all clients which observe each other's operations are consistent, in the sense that their own operations, plus those operations whose effects they see, have occurred atomically in one sequence. Otherwise, a faulty service could answer with arbitrary values from past operations and return diverging results to different clients.

The tutorial describes the basic principles underlying these protocols, addressing outsourced storage services and arbitrary remote computations.

5. REFERENCES

[1] B. Braun, A. J. Feldman, Z. Ren, S. T. V. Setty, A. J. Blumberg, and M. Walfish. Verifying computations with state. In *Proc. 24th ACM Symposium on Operating Systems Principles (SOSP)*, pages 341–357, 2013.

[2] C. Cachin, I. Keidar, and A. Shraer. Fail-aware untrusted storage. *SIAM Journal on Computing*, 40(2):493–533, Apr. 2011. Preliminary version appears in *Proc. DSN 2009*.

[3] C. Cachin, A. Shelat, and A. Shraer. Efficient fork-linearizable access to untrusted shared memory. In *Proc. 26th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 129–138, 2007.

[4] R. Gennaro, C. Gentry, and B. Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In T. Rabin, editor, *Advances in Cryptology: CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 465–482. Springer, 2010.

[5] R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct NIZKs without PCPs. In T. Johansson and P. Q. Nguyen, editors, *Advances in Cryptology: EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*. Springer, 2013.

[6] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum. Delegating computation: Interactive proofs for muggles. In *Proc. 40th ACM Symposium on Theory of Computing (STOC)*, pages 113–122, 2008.

[7] J. Groth. Short pairing-based non-interactive zero-knowledge arguments. In M. Abe, editor, *Advances in Cryptology: ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 321–340. Springer, 2010.

[8] J. Li, M. Krohn, D. Mazières, and D. Shasha. Secure untrusted data repository (SUNDR). In *Proc. 6th Symp. Operating Systems Design and Implementation (OSDI)*, pages 121–136, 2004.

[9] C. Martel, G. Nuckolls, P. Devanbu, M. Gertz, A. Kwong, and S. G. Stubblebine. A general model for authenticated data structures. *Algorithmica*, 39:21–41, 2004.

[10] D. Mazières and D. Shasha. Building secure file systems out of Byzantine storage. In *Proc. 21st ACM Symposium on Principles of Distributed Computing (PODC)*, 2002.

[11] C. Papamanthou, R. Tamassia, and N. Triandopoulos. Optimal verification of operations on dynamic sets. In P. Rogaway, editor, *Advances in Cryptology: CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 91–110. Springer, 2011.

[12] B. Parno, C. Gentry, J. Howell, and M. Raykova. Pinocchio: Nearly practical verifiable computation. In *Proc. 34th IEEE Symposium on Security & Privacy*, 2013.

[13] S. T. V. Setty, R. McPherson, A. J. Blumberg, and M. Walfish. Making argument systems for outsourced computation practical (sometimes). In *Proc. 19th Network and Distributed Systems Security Symposium (NDSS)*, 2012.

[14] S. T. V. Setty, V. Vu, N. Panpalia, B. Braun, A. J. Blumberg, and M. Walfish. Taking proof-based verified computation a few steps closer to practicality. In *Proc. 21st USENIX Security Symposium*, pages 253–268, 2012.

[15] R. Tamassia. Authenticated data structures. In G. Di Battista and U. Zwick, editors, *Proc. 11th European Symposium on Algorithms (ESA)*, volume 2832 of *Lecture Notes in Computer Science*, pages 2–5. Springer, 2003.

[16] P. Williams, R. Sion, and D. Shasha. The blind stone tablet: Outsourcing durability to untrusted parties. In *Proc. Network and Distributed Systems Security Symposium (NDSS)*, 2009.

SPEAKER BIOGRAPHY

Christian Cachin is a researcher in cryptography and security at IBM Research - Zurich. He graduated with a Ph.D. in Computer Science from ETH Zurich and has held visiting positions at MIT and at EPFL.

Christian Cachin's research focuses on cryptology and distributed systems. He has authored many peer-reviewed publications in these fields, holds several patents on secure protocols, and has been a frequent member of program committees of technical conferences, of which he chaired several. He is an ACM Distinguished Scientist (2009) and has received multiple IBM Outstanding Technical Achievement Awards. Since 1998 he has been a member of the board of directors of the International Association for Cryptologic Research (IACR), currently as acting as its President.

He serves as an editor for international journals in the area of information security and is an author of the book "Introduction to Reliable and Secure Distributed Programming." Furthermore, he contributed to the OASIS Key Management Interoperability Protocol (KMIP) standard, which debuted in 2010. His current research addresses the security of cloud computing, secure protocols for distributed systems, and cryptography.

More information is available from `http://www.zurich.ibm.com/~cca/`