Modeling Complexity in Secure Distributed Computing*

Christian Cachin

IBM Research, Zurich Research Laboratory
CH-8803 Rüschlikon, Switzerland
cca@zurich.ibm.com

1 Introduction

Security considerations play an increasingly important role for distributed computing. In the future, dependable distributed systems for open networks can no longer be designed without taking malicious attacks into account. The enabling technology for security is cryptography, which has been placed on sound theoretical foundations during the last twenty years. The formal model of modern cryptography is based on computational complexity theory because of the need for modeling computational difficulty; it differs substantially from the formal models of distributed computing, which do not usually deal with bounds on time complexity or with randomization. We argue that an integration of these two approaches is necessary for reasoning about the security of cryptographic protocols in distributed systems. We discuss the notion of a *uniformly bounded protocol statistic* that allows for composing protocols with computational security; it has recently been proposed for constructing cryptographic randomized atomic broadcast protocols for asynchronous systems.

2 Two Formal Approaches

Distributed computing. The prevalent formal models in distributed computing today are based on finite automata and on infinite time. In the *I/O automaton* model [10], for instance, liveness and fairness properties of a system are expressed in terms of the system's external behavior (its trace) as observed during a potentially infinite run.

Take the classical problem of *Byzantine agreement* [11], where a set of n parties must reach the same decision despite the fact that up to t of them fail in arbitrary, potentially malicious ways. The standard formalization consists of three conditions:

* This work was supported by the European IST Project MAFTIA (IST-1999-11583). However, it represents the view of the author(s). The MAFTIA project is partially funded by the European Commission and the Swiss Department for Education and Science.

Validity: If all non-faulty parties start with the same value v, every non-faulty party that terminates decides for v.
Agreement: No two non-faulty parties decide on different values.
Termination: All non-faulty parties eventually terminates.

No bound is placed on the time it takes to reach agreement; the parties are allowed to perform an a priori unbounded number of computation steps.

Because of its adversarial nature, Byzantine agreement is perhaps the best problem to illustrate the use of cryptographic techniques in distributed computing. If one augments the model with a digital signature scheme such that every party can add an unforgeable authentication tag to any message, which can be verified by all parties, the problem becomes easier to solve. The formal statement of the authenticated model requires an *ideal* digital signature scheme, where it is *impossible* for any faulty party to generate a tag that is recognized as a valid signature by a non-faulty party.

Cryptography. Since the discovery of public-key cryptography [5], research in theoretical cryptography has concentrated on appropriate models for cryptographic tasks, such as encryption and digital signatures. The security notions of modern cryptography, originating with [7], are based on asymptotic formalizations in the tradition of complexity theory (see [6] for an introduction). This is because the efficient cryptographic algorithms available today provide only *computational* security guarantees against adversaries whose resources are bounded.

As an example, consider a *one-way function* — one of the fundamental concepts in modern cryptography. Such a function is easy to evaluate but hard (on average) to invert. It is intuitively clear that a digital signature scheme must involve a one-way function because every party should be able to verify a signature issued by party P with an efficient algorithm, but no feasible computation by a malicious party must be able to come up with P's signature on a message that P has not signed. Unfortunately, real-world cryptographic primitives are not ideal: a simple algorithm may guess an input for the one-way function and hit one that is mapped to a given output with non-zero probability, or, for that matter, forge a signature of P with non-zero probability.

Modern cryptography takes this into account by introducing a security parameter k and formalizing the asymptotic behavior of a primitive in dependence of k. The security parameter may be thought of as indicating the key length of the primitive. The basic assumptions are that polynomial-time algorithms are efficient and that a "negligible" probability of failure cannot be ruled out. A function $\epsilon(k)$ is called *negligible* if it decreases faster than any inverse polynomial, i.e., if for all $c > 0$, there exists a constant k_0 such that $\epsilon(k) < \frac{1}{k^c}$ for $k > k_0$.

Now, a *one-way function* may be defined as a *family of functions* $f_k : \{0,1\}^k \to \{0,1\}^k$ for $k > 0$ such that (1) there exists a polynomial-time algorithm that computes $f_k(x)$ for all $x \in \{0,1\}^k$, and (2) for any probabilistic polynomial-time algorithm A, there exists a negligible function $\epsilon_A(k)$ such that

$$\Pr\Big[f\big(A(f(x))\big) = f(x)\Big] \leq \epsilon_A(k)$$

where the probability is over the uniform choice of $x \in \{0,1\}^k$ and the random choices of A. Algorithms are modeled as Turing machines that take k as an (implicit) auxiliary input to enforce their dependence on k (for technical reasons k is often given in unary notation as 1^k).

It is clear that these two formal models cannot be combined in a straightforward manner, say, for analyzing randomized Byzantine agreement protocols that use cryptography. At the very least, an appropriate model should allow a protocol to fail with negligible probability because a cryptographic primitive has been broken.

3 Towards a Unified Model

We propose to explore a new formal system model for distributed computing that bridges this gap. It is a refinement of traditional models in distributed computing, such as the I/O automaton model, which allows for reasoning about the computational complexity of a protocol and for implementing protocols with cryptographic primitives. Because the model must allow to bound the running time of a system, most changes will affect the formal treatment of termination. We introduce the notion of *uniformly bounded statistics* for this purpose. Our model allows also for randomized protocols, whose running time cannot be bounded a priori, through the concept of *probabilistic uniformly bounded statistics*.

A preliminary version of such a model has been developed in connection with asynchronous cryptographic protocols for distributed systems with Byzantine faults [3, 2, 9, 1, 4]; two of its components, Turing machines and uniformly bounded protocol statistics, are presented next.

Turing machines. A party executing a particular protocol is modeled by a probabilistic interactive Turing machine [8] that runs in polynomial time in the security parameter k. Two interactive Turing machines communicate through a pair of special communication tapes, where each party may only write on one tape and read from the other tape. Input and output actions of the protocol are also represented as messages on communication tapes.

There is a single adversary A, which is a probabilistic interactive Turing machine that runs in polynomial time in k. W.l.o.g. every party communicates only with the adversary, who therefore also implements the network (assuming no secret channels are required). We sometimes restrict the adversary's behavior such that it implements a reliable network by saying that it *delivers all messages*. The parties are completely reactive and receive service requests (input actions) from the adversary and deliver their payload (output actions) also to the adversary. The notion of compatible protocols and their composition may be defined analogously to the I/O automaton model.

In the Byzantine agreement example, there are n parties ($n \leq k$), of which up to t are *faulty* and controlled by the adversary; for simplicity, faulty parties are absorbed into the adversary.

Uniformly bounded statistics. We say that a message written to a communication tape is *associated* to a given protocol if it was generated by a *non-faulty* party on behalf of the protocol. The *message complexity* of a protocol is defined as the number of associated messages (generated by non-faulty parties). It is a random variable that depends on the adversary and on k. The *communication complexity* of a protocol may be defined analogously.

For a particular protocol, a *protocol statistic* X is a family of real-valued, non-negative random variables $X_A(k)$, parameterized by adversary A and security parameter k, where each $X_A(k)$ is a random variable induced by running the system with A. We restrict ourselves to *bounded protocol statistics* X such that for all A, there exists a polynomial p_A with $X_A(k) \leq p_A(k)$ for $k > 0$ (this bound may depend on A). Message complexity is an example of such a bounded protocol statistic.

We say that a bounded protocol statistic X is *uniformly bounded* if there exists a fixed polynomial $p(k)$ such that for all adversaries A, there is a negligible function ϵ_A, such that for all $k \geq 0$,

$$\Pr\big[X_A(k) > p(k)\big] \leq \epsilon_A(k).$$

A protocol statistic X is called *probabilistically uniformly bounded* if there exists a fixed polynomial $p(k)$ and a fixed negligible function δ such that for all adversaries A, there is a negligible function ϵ_A, such that for all $l \geq 0$ and $k \geq 0$,

$$\Pr\big[X_A(k) > lp(k)\big] \leq \delta(l) + \epsilon_A(k).$$

In other words, (probabilistically) uniformly bounded protocol statistics are *independent* of the adversary except with negligible probability. Assuming that the adversary delivers all messages, termination of a cryptographic protocol may be defined by requiring that the message complexity is (probabilistically) uniformly bounded.

Example. For illustration, we provide a definition of Byzantine agreement with computational security in the new model. For all polynomial-time adversaries, the following holds except with negligible probability:

Validity and *Agreement* as before.
Liveness: If all non-faulty parties have started and all associated messages have been delivered, then all non-faulty parties have decided.
Efficiency: The message complexity of the protocol is probabilistically uniformly bounded.

Hence, protocols are live only to the extent that the adversary chooses to deliver messages among the non-faulty parties, but they must not violate safety even if the network is unreliable.

Termination follows from the combination of *liveness* and *efficiency*. These properties ensure that the protocol generates some output and that the number of communicated messages is *independent* of the adversary, causing the protocol to terminate by ceasing to produce messages.

If X is (probabilistically) uniformly bounded by p, then for all adversaries A, we have $\mathrm{E}[X_A(k)] = O(p(k))$, with a hidden constant that is independent of A. Additionally, if Y is (probabilistically) uniformly bounded by q, then $X \cdot Y$ is (probabilistically) uniformly bounded by $p \cdot q$, and $X + Y$ is (probabilistically) uniformly bounded by $p + q$. Thus, (probabilistically) uniformly bounded statistics are closed under polynomial composition, which is their main benefit for analyzing the composition of (randomized) cryptographic protocols.

Outlook. A formal model that unites the approaches of distributed systems and cryptography has been sketched. It remains to be seen how much of the considerable work in distributed systems can be presented in such a framework. Many interesting topics, such as hybrid models that differentiate between benign and malicious faults, should be revisited with taking security and complexity considerations into account.

Acknowledgments

This paper is based on joint work with Victor Shoup and Klaus Kursawe.

References

1. C. Cachin, K. Kursawe, A. Lysyanskaya, and R. Strobl, "Asynchronous verifiable secret sharing and proactive cryptosystems," in *Proc. 9th ACM Conference on Computer and Communications Security (CCS)*, 2002. To appear.
2. C. Cachin, K. Kursawe, F. Petzold, and V. Shoup, "Secure and efficient asynchronous broadcast protocols (extended abstract)," in *Advances in Cryptology: CRYPTO 2001* (J. Kilian, ed.), vol. 2139 of *Lecture Notes in Computer Science*, pp. 524–541, Springer, 2001.
3. C. Cachin, K. Kursawe, and V. Shoup, "Random oracles in Constantinople: Practical asynchronous Byzantine agreement using cryptography," in *Proc. 19th ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 123–132, 2000.
4. C. Cachin and J. A. Poritz, "Secure intrusion-tolerant replication on the Internet," in *Proc. Intl. Conference on Dependable Systems and Networks (DSN-2002)*, pp. 167–176, June 2002.
5. W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, pp. 644–654, Nov. 1976.
6. O. Goldreich, *Foundations of Cryptography: Basic Tools.* Cambridge University Press, 2001.
7. S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of Computer and System Sciences*, vol. 28, pp. 270–299, 1984.
8. S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on Computing*, vol. 18, pp. 186–208, Feb. 1989.
9. K. Kursawe and V. Shoup, "Optimistic asynchronous atomic broadcast." Cryptology ePrint Archive, Report 2001/022, Mar. 2001. `http://eprint.iacr.org/`.
10. N. A. Lynch, *Distributed Algorithms.* San Francisco: Morgan Kaufmann, 1996.
11. M. Pease, R. Shostak, and L. Lamport, "Reaching agreement in the presence of faults," *Journal of the ACM*, vol. 27, pp. 228–234, Apr. 1980.