A second look at Shoup's lemma

Bruno Blanchet
INRIA, École Normale Supérieure, CNRS, Paris
blanchet@di.ens.fr

In order to show that a security protocol satisfies a security property P in the computational model, cryptographers generally use proofs by sequences of games [4]. The first game G_0 is the protocol to prove. Each game G_i is transformed into the next one G_{i+1} using game transformations, such that the probability p_i that an adversary distinguishes G_i from G_{i+1} is bounded by a small value, for instance the probability of solving a difficult computational problem. Finally, the last game G_n is such that the desired security property P is obvious from the form of the game, so that the probability that an adversary breaks P is 0 in this game. The probability that an adversary breaks P in the initial game is then at most the sum $p_0 + \cdots + p_{n-1}$.

Shoup's lemma [4] is a technique frequently used in proofs by sequences of games. One transforms the game G_i into G_{i+1} by introducing some event e: G_{i+1} behaves differently from G_i only when it executes the event e. The probability p_i of distinguishing G_i from G_{i+1} is then the probability that G_{i+1} executes e. This probability is itself bounded by performing a proof by sequences of games starting from G_{i+1}. Often, in order to bound the probability of e, we wait until a later game of the sequence $G_1, \dots, G_n$ in which that probability is easier to bound, so that the proof that serves in bounding the probability of e shares some or all game transformations with the proof of the initial security property P: this is the "deferred analysis" technique [3]. Suppose that the sequence $G_{i+1}, \dots, G_n$ ends with a game G_n that never executes e and such that P is always true. Then the probability that G_{i+1} executes e is at most $p_i = p_{i+1} + \ldots + p_{n-1}$. The probability that an adversary breaks P is then at most $p_0 + \cdots + p_{n-1} = p_0 + \cdots + p_{i-1} + 2(p_{i+1} + \cdots + p_{n-1})$.

In this talk, we will show that the factor 2 in this formula can be avoided. (More generally, other constant factors that appear when several events are introduced can also be avoided.) We prove this result by considering the property "e is executed or P is broken" instead of considering separately the event e and the property P.

This result has been shown in the context of the automatic protocol prover CryptoVerif [1]; it is implemented in this prover, but it also applies to manual proofs. It allows us to obtain better probability bounds than with the standard computation of probabilities. For example, in the proof of the password-based protocol One-Encryption Key Exchange [2], [2] shows that the adversary can test at most 3 passwords per interaction with the protocol. By applying our improvement in the computation of probabilities, we can show using the same sequence of games that the adversary can in fact test at most one password per session of the protocol, which is the optimal result.

References

[1] B. Blanchet. A computationally sound mechanized prover for security protocols. *IEEE Transactions on Dependable and Secure Computing*, 5(4):193–207, Oct.–Dec. 2008.

[2] E. Bresson, O. Chevassut, and D. Pointcheval. Security proofs for an efficient password-based key exchange. In V. Atluri, editor, *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, pages 241–250, Washington DC, 2003. ACM Press.

[3] R. Gennaro and V. Shoup. A note on an encryption scheme of Kurosawa and Desmedt. Cryptology ePrint Archive, Report 2004/194, Aug. 2004. Available at http://eprint.iacr.org/2004/194.

[4] V. Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, Nov. 2004. Available at http://eprint.iacr.org/2004/332.