IMPLEMENTING LEAKAGE-RESILIENT CRYPTOGRAPHY

MICHAŁ ZAJĄC

Ülikooli 17-305 (Paabel)

INTRODUCTION

Leakage-resilient cryptography. Modern cryptography (usually) relies on some secret (like a private key) that is assumed to be picked from uniform distribution. Unfortunately, this assumption is hard to justify in the real world, where an adversary can learn some side information on the secret from e.g., observing physical effects on a device running a cryptographic scheme or harvesting remains of information that weren't properly removed from device's memory. Leakage-resilient cryptography has been introduced to propose schemes that are immune to such kinds of attacks.

Leakage-resilient cryptography comes with a number of flavors. Some models assumes that only computation leaks information (i.e., you cannot learn information from memory) or that some parts of memory leaks while other don't. We will focus on a more general model, bounded retrieval model [Dzi06]. This model assumes that a secret stored on a device is so huge that adversary cannot obtain (by any means) any significant amount of information of it. However, in many schemes computational cost is proportional to the length of a key. Hence, making key, say, a few GB large makes it unusable. Bounded retrieval model deals with this problem by requiring that the length of a key doesn't affect computational cost (security parameter is independent from the length of a key). This requirement is quite hard to fulfill, so no many BRM cryptographic schemes are known. However, [ADW09] proposes first schemes that connect world of public key cryptography with leakage resiliency in the bounded retrieval model. And in this project I would like you to read the paper, understand the scheme and implement it.

PROJECT OBJECTIVES

We name the following objectives that make project successful.

Read a paper and understand what is in it: In this project I would like you to understand basic principles and techniques used in the leakage-resilient cryptography. You will learn some notions from information theory that are especially useful in cryptography. The stuff in the paper will be probably new for you and a little challenging, but it doesn't require wide background knowledge. If you have seen a cryptographic black-box proof you will have some advantage.

Implement public key crypto primitives: One of the tasks will be to implement proposed identification scheme and authenticated key agreement. For comparison you will be asked to implement a classical identification scheme (Okamoto scheme, described also in [ADW09]).

Make it works: Another result of the project is a mobile-device application (Os X, Android, your call). There should be a client (an application) that (depending on what has been chosen in an item above) that both identifies itself and agrees with a server on a key.

E-mail address: michal.zajac@ut.ee.

Report efficiency: Although BRM schemes efficiency doesn't depend on a key length, it doesn't mean that it is as efficient as schemes in more standard models. Thus, the last task will be to tell how less efficient is Okamoto BRM version from a classical one. I would like you to prepare a report that will show time needed to identification for a number of system parameters (the specific values for the parameters will be discussed later).

How can I help

I can help you understand the paper, but I will not be able to help you during the implementation. However, I spend quite a lot of time in Paabel, so you can always come and discuss any problem occurred.

References

[ADW09] Joël Alwen, Yevgeniy Dodis, and Daniel Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 36–54. Springer, 2009.

[Dzi06] Stefan Dziembowski. Intrusion-resilience via the bounded-storage model. In Shai Halevi and Tal Rabin, editors, *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 207–224. Springer, 2006.