Plain-Text Aware Encryption Via Public-Key Registration

Silvio Micali*

Laboratory for Computer Science
MIT
545 Technology Square
Cambridge, MA 02139

November 20, 2002

Abstract

We show that a simple public-key registration protocol yields very practical encryption schemes that are provably plain-text aware in a standard complexity-theoretic model (e.g., without relying on random oracles in any way).

1 Introduction

The notion of *plain-text awareness (PT-awareness)* has been defined and constructed by Bellare and Rogaway [**?**] using random oracles. In essence, in a PT-aware encryption scheme, any ciphertext vouches that "he who generated it knows the corresponding plaintext."

PT-awareness is the strongest known form of encryption. In particular, it immediately implies security against adaptive chosen-ciphertext attack[1] and appears to be strictly stronger. Indeed, none of the schemes proved so far secure against chosen-ciphertext attack (e.g., [**?**], [**?**], [**?**]) are PT-aware.

A natural question thus arises: *Can we construct practical PT-Aware cryptosystems?* As already said, Bellare and Rogaway only construct them assuming random oracles. Such oracles, however, are not readily available, nor could one hope to replace them with traditional cryptographic functions. Never mind, if history is a good teacher, that it would be hard (if not impossible) to *prove* PT-awareness after such a replacement: it is even unclear how to *define* PT-awareness without random oracles!

Our Contributions. We show that a simple pre-processing step (indeed, an almost traditional key-registration process) enables us both to *define* and to *construct* PT-aware cryptosystems without random oracles.

Our construction is reasonably general in the sense that it transforms cryptosystems that are less secure, but satisfying a technical property, into PT-aware ones. Our construction also is actually quite *practical.* Indeed, applying our transformation to the traditional Diffie-Hellman system, we obtain a PT-aware cryptosystem in which the basic encryption and decryption operations essentially require as much computation as the underlying Diffie-Hellman system alone. We note that an older construction of Rackoff and Simon [**?**],

*E-mail: silvio@theory.lcs.mit.edu

[1]Informally, an enemy conducting a chosen-ciphertex attack is entitled to generate a ciphertext C and receive the corresponding plain-text P in response. However, if the cryptosystem is PT-aware, then this extraordinary ability provides no useful information: anyone generating C knows P anyway!

for cryptosystems secure against adaptive chosen-ciphertext attack, might be also deemed to be PT-aware, but was rather impractical. It involved a trusted party assigning a public key and its corresponding secret decryption key to every user, and required a new, non-interactive zero-knowledge for complex NP statements for each new ciphertext.[2]

Our Key Registration. Public-key encryption requires a correct association between users and their public keys. (Else, one might send messages encrypted with the key of an adversary.) This correct association is accomplished by a key registration process. Typically, a user U generates a pair of matching public and secret keys, PK_U and SK_U, and gives PK_U to a certification authority (CA) providing a proof of his own identity (e.g., U visits the CA in person and presents his own picture ID to him). The CA responds by issuing a *digital certificate*, C_U, vouching that PK_U indeed belongs to user U. He does so by digitally signing together PK_U and a suitable identifier of U (and possibly additional quantities): in symbols $C_U = SIG_CA = (PK_U, U, \ldots)$.

It has been widely recognized, however, that U should also be prevented from presenting to the CA the public key, PK_V, of some other user V and obtaining a certificate stating that PK_V is Us public key. To this end, it is quite common nowadays to include in the key-registration process for the public key PK_U of a user U a "proof that U indeed knows the corresponding secret key SK_U. Such proof typically takes many forms. (For instance, the CA may challenge U with the encryption of a random string with PK_U and U provide the correct decryption.)

Our key-registration process essentially is a simple strengthening of this step: it requires that, in order to register PK_U, U provide the CA with a zero-knowledge proof of knowledge of the corresponding key SK_U. Often, as in the case of the Diffie-Hellman system, this may be accomplished by means of sufficiently simple protocol.

Our Construction. We actually break our construction in two steps.

1. We show that our simple key-registration protocol, combined with the random function generator of [**?**],

   yields a very efficient cryptosystem that is provably secure against chosen ciphertext attacks under the *computational Diffie-Hellman* assumption.

2. We show how to transform our first cryptosystem into one that is PT-aware by adding a proper authentication step.

This two-stage construction makes the relative contribution of our key-registration and authentication steps very clear: the first already adds value on its own. Our first construction is in fact of independent interest. Indeed, with the exception of the Cramer Schoup cryptosystem, all cryptosystems proved secure against chosen-ciphertext attack required non-interactive zero-knowledge proofs, and thus were very impractical. As for the Cramer Shoup cryptosystem, it relies on the *decisional* Diffie-Hellman assumption (a much stronger assumption than ours!) and requires twice the amount of computation of our system, both for encrypting and decrypting.

[2]Notice that even producing a single non-interactive zero-knowledge proofs is impractical, let alone producing such a proof each time one wants to encrypt a message! In fact, such ZK proofs needed in their case to be dependent on the message to be encrypted, and thus could not be computed off-line, that is, before choosing the message to be encrypted.