

Advanced Cryptography — G22.3220-001 Spring 2007 — Problem Set 2
Due: Tuesday, April 3

This problem set develops the notion of *semantically secure verifiable encryption of discrete logarithms*. Roughly speaking, there are three parties involved: a prover, a verifier, and a decryptor. The decryptor generates a public key/secret key pair (pk, sk) , and publishes the public key pk . Also assume we are working with a finite cyclic group G with generator g . The prover and verifier receive as common inputs pk and a group element $h \in G$. The prover receives as an additional, private input x such that $g^x = h$. The prover and verifier then engage in an interactive protocol, at the end of which, the verifier either *accepts* or *rejects*. Finally, the verifier may submit the transcript of an accepting conversation to the decryptor, who applies a specific decryption algorithm using the secret key sk and the transcript to obtain the discrete logarithm x .

Intuitively, the requirements of such a verifiable encryption system are as follows:

Completeness: If prover and verifier follow the protocol, then the verifier accepts with overwhelming probability.

Soundness: It is infeasible for a cheating prover to make an honest verifier accept a conversation that does not decrypt to a correct discrete logarithm.

Simulatability: A conversation between an honest prover and an honest verifier can be efficiently simulated in such a way that simulated conversations are computationally indistinguishable from real conversations.

1. Give precise, formal definitions of the above intuitively defined properties. For each, give a detailed attack game between a challenger and an adversary, and define the appropriate notion of the adversary's advantage. Assume that the description of the group G and generator g is a "system parameter" generated honestly (and not adversarially). Similarly, assume that the key pair (pk, sk) is always generated honestly (and not adversarially).
2. Consider the following protocol, which makes use of an arbitrary, semantically secure encryption scheme. The public key/secret key pair for the protocol is the corresponding key pair for the encryption scheme. Let E be the (probabilistic) encryption algorithm for the encryption scheme, and assume the message space for the scheme includes \mathbb{Z}_q , where q is the order of the group G (and not necessarily prime). We write $E(pk, m; \omega)$ for an encryption of m under the public key pk , where ω represents the randomness used by the encryption algorithm, chosen uniformly from some finite set Ω . The protocol runs as follows, where t is an appropriately chosen parameter, and the common inputs are pk and $h \in G$, and the private input to the prover is $x \in \mathbb{Z}_q$, where $g^x = h$:

- for $i = 1, \dots, t$, the prover computes

$$r_i \leftarrow_R \mathbb{Z}_q, a_i \leftarrow g^{r_i}, z_{i0} \leftarrow r \in \mathbb{Z}_q, z_{i1} \leftarrow r + x \in \mathbb{Z}_q, \\ \omega_{i0} \leftarrow_R \Omega, e_{i0} \leftarrow E(pk, z_{i0}; \omega_{i0}), \omega_{i1} \leftarrow_R \Omega, e_{i1} \leftarrow E(pk, z_{i1}; \omega_{i1}),$$

and sends

$$\{(a_i, e_{i0}, e_{i1})\}_{i=1}^t$$

to the verifier;

- for $i = 1, \dots, t$, the verifier computes $c_i \leftarrow_R \{0, 1\}$ and sends

$$\{c_i\}_{i=1}^t$$

to the prover;

- for $i = 1, \dots, t$, the prover sets

$$z_i \leftarrow z_{ic_i}, \omega_i \leftarrow \omega_{ic_i},$$

and sends

$$\{(z_i, \omega_i)\}_{i=1}^t$$

to the verifier;

- for $i = 1, \dots, t$, the verifier checks that

$$e_{ic_i} = E(\text{pk}, z_i; \omega_i) \text{ and } g^{z_i} = a_i h^{c_i};$$

if this holds, the verifier *accepts*, and otherwise *rejects*.

Show that this protocol satisfies all the requirements of a verifiable encryption system, as outlined above. You will have to indicate how t should be chosen, and how the decryptor's algorithm for retrieving a discrete logarithm should work.

3. The system in the previous exercise uses single-bit challenges, which leads to many repetitions and a corresponding loss in efficiency. This type of approach is sometimes called a "cut and choose" strategy. In this exercise, we use Paillier encryption to avoid this, obtaining a more efficient system.

Recall that for Paillier encryption, the public key is $N = PQ$, where P and Q are large, distinct primes, such that $P \nmid Q - 1$ and $Q \nmid P - 1$. To encrypt a message $m \in \mathbb{Z}_N$, the decryption algorithm chooses $\omega \in \mathbb{Z}_N^*$ at random, and computes $e \leftarrow \omega^N \alpha^m \in \mathbb{Z}_{N^2}^*$, where $\alpha = [1 + N \bmod N^2] \in \mathbb{Z}_{N^2}^*$.

In our verifiable encryption system, let us assume that G is a cyclic group of order N . For example, after choosing N , one could generate a prime $\mathfrak{p} \equiv 1 \pmod{N}$, and then take G to be the subgroup of $\mathbb{Z}_{\mathfrak{p}}^*$ of order N . However, for this exercise, how G is generated is not important.

The protocol runs as follows, where T is an appropriately chosen parameter, and the common inputs are N and $h \in G$, and the private input to the prover is $x \in \mathbb{Z}_N$, where $g^x = h$:

- the prover computes

$$\omega, \omega' \leftarrow_R \mathbb{Z}_N^*, x' \leftarrow_R \mathbb{Z}_N, e \leftarrow \omega^N \alpha^x \in \mathbb{Z}_{N^2}^*, e' \leftarrow (\omega')^N \alpha^{x'} \in \mathbb{Z}_{N^2}^*, h' \leftarrow g^{x'} \in G,$$

and sends e, e', h' to the verifier;

- the verifier computes

$$c \leftarrow_R \{0, \dots, T - 1\},$$

and sends c to the prover;

- the prover computes

$$\hat{\omega} \leftarrow \omega' \omega^c \in \mathbb{Z}_N^*, \hat{x} \leftarrow x' + cx \in \mathbb{Z}_N,$$

and sends $\hat{\omega}, \hat{x}$ to the verifier;

- the verifier checks that

$$\hat{\omega}^N \alpha^{\hat{x}} = e' e^c \text{ and } g^{\hat{x}} = h' h^c$$

and if so *accepts*, and otherwise, *rejects*.

Show that this protocol satisfies all the requirements of a verifiable encryption system, as outlined above, assuming the decisional composite residuosity (DCR) assumption holds. You will have to indicate how T should be chosen, and how the decryptor's algorithm for retrieving a discrete logarithm should work.

4. A severe drawback of the protocol in the previous exercise is that the order of the group G must be equal to N . Show how to modify that protocol so that it works with an arbitrary group G of prime order q . In addition to the DCR assumption, you may also use the Strong RSA assumption. You may also make any reasonable assumptions about the relative sizes of q, N, P, Q , and T , but be explicit. You may want to augment the system parameters or public key with some additional values. Give a complete analysis of your protocol.