

8 Further comments on RSA and ElGamal

Example 166. Does Alice have to choose a new y if she sends several messages to Bob using ElGamal?

Solution. Yes, she absolutely has to randomly choose a new y every time! Here's why:

If she was using the same y to encrypt messages $m^{(1)}$ and $m^{(2)}$, Alice would be sending the ciphertexts $(c_1^{(1)}, c_2^{(1)}) = (g^y, g^{xy}m^{(1)})$ and $(c_1^{(2)}, c_2^{(2)}) = (g^y, g^{xy}m^{(2)})$.

That means, Eve can immediately figure out $c_2^{(1)} / c_2^{(2)} = m^{(1)} / m^{(2)}$ (the division is a modular inverse and everything is modulo p). That's a combination of the plaintexts, and Eve should never be able to get her hands on such a thing.

(Also, Eve would know right away that Alice is doing the mistake of reusing y because $c_1^{(1)} = c_1^{(2)}$.)

Comment. The situation is just like for the one-time pad (in that case, reusing the key reveals $m^{(1)} \oplus m^{(2)}$).

Definition 167. Bob's public key cryptosystem is **semantically secure** if Eve cannot do better than guessing in the following challenge:

- Bob determines a random public and private key. The public key is given to Eve.
- Eve selects two plaintexts m_1 and m_2 .
- Alice flips a fair coin and, accordingly, using the public key encrypts m_1 or m_2 as c .
- Eve now needs to decide whether c is the encryption of m_1 or m_2 .

For this definition to make precise mathematical sense, we need to assume that Eve's computing power is somehow limited (typically, she is limited to polynomial-time algorithms).

Comment. Also, many variations exist of what semantic security exactly is. All of these try to capture the idea that an attacker does not learn anything about m from knowing c . The one above is often referred to as IND-CPA (Indistinguishability under Chosen Plaintext Attack).

Important comment. Realize that semantic security is a very strong property to ask for! In particular, this is much stronger than what we usually think about in terms of security: you might call a cipher secure if it is "impossible" for an attacker to get m from c . Semantic security is requiring that an attacker gets so little information from c that she cannot even tell whether it came from (her own choices) m_1 or m_2 .

Example 168. Is vanilla RSA semantically secure?

Solution. No. Eve can just encrypt both m_1 and m_2 herself, and compare with c . She then knows for sure which of the two was encrypted.

Comment. As mentioned before, in practice, RSA is never used in its vanilla (or "textbook") version. Instead, it is randomized (like ElGamal is by design) by padding the plaintext with random stuff.

Check out OAEP: https://en.wikipedia.org/wiki/Optimal_asymmetric_encryption_padding

The resulting RSA-OAEP has been proven semantically secure (and the "RSA assumption" that finding m from c is hard).

Example 169. Is ElGamal semantically secure?

Solution. Essentially, yes.

Recall that the public key is $(p, g, h) = (p, g, g^x)$.

The ciphertext is $(c_1, c_2) = (g^y, h^y m) = (g^y, g^{xy} m)$. Eve needs to decide whether the m in there is m_1 or m_2 .

Equivalently, she needs to decide whether $r = c_2 / m_1$ (or $r = c_2 / m_2$) equals g^{xy} or not.

This is essentially the DDH problem.

Strictly speaking. Because of the issue with quadratic residues mentioned when we introduced the DDH problem, ElGamal is not semantically secure in the sense we defined things. However, if we wanted (this is more of a theoretical point), this issue could be fixed by not computing with all invertible residues modulo p , but only with quadratic residues. We could further select p such that $(p - 1) / 2$ is prime again, in which case all quadratic residues (except 1) have order $(p - 1) / 2$ (so that no similar games can be played using orders of elements).

Practical implications. Indeed, Diffie–Hellman and ElGamal in practice often use such **safe primes** p , which are primes such that $(p - 1) / 2$ is prime, too. In that case, there is no elements of small order (besides 1 and -1). Since generating such primes can be a bit expensive, it is common to use preselected ones. For instance, RFC 3526 lists six such primes (together with a generator g) with 1536, 2048, ..., 8192 bits.

<https://www.ietf.org/rfc/rfc3526.txt>

Important. It is perfectly fine that p and g are not random in Diffie–Hellman or ElGamal. However, it is absolutely crucial that x (and y) are random (generated using a cryptographically secure PRG).