Esecurity: secure internet & e-cash, summer 2012

MICHAEL NÜSKEN, RAOUL BLANKERTZ

4. Exercise sheet
Hand in solutions until Sunday, 06 May 2012, 23:59

Note: On this exercise sheet groups are written *multiplicatively*, instead of *additively* as in the lecture.

Exercise 4.1 (Repetition: Security notions). (12 points)

Recall the following notions from your Cryptography lecture (or read Chapter 7 in Stinson (2006) or Chapter 10 in Bellare & Goldwasser (2008)): There are several levels of security

- Unbreakability (UB),
- Universal Unforgeability (UUF; also called *selective* unforgeability),
- Existential Unforgeability (EUF);

along with different means for an attacker:

- Key-Only Attack (KOA),
- Known Signature Attack (KSA),
- Chosen Message Attack (CMA).

Pairing an adversarial goal with an attack model defines a security notion, e.g. EUF-CMA.

(i) Give a short description of each security level and each attack. Does security in one notion imply security in some other notions? Picture the implications in a suitable way. [4]

(ii) Consider the ElGamal signature scheme with a cyclic group G. Assume that the discrete logarithm problem for G (DL_G) is hard, ie. it is hard to compute a from g^a where g is a generator of G. Decide for each of the 9 security notions whether the scheme is [6]

- ◦ secure,
- ◦ not secure, or
- ◦ the answer is unknown.

Give for each claim a short hint or quote.

2 (iii) What can you say, if you assume that DL_G is easy?

Exercise 4.2. (16 points)

Let $G = \langle g \rangle$ be a cyclic group. In this exercise we prove that ElGamal is IND-KOA secure if the decisional Diffie–Hellman problem (DDH) is hard in the underlying group G.

Let $\mathcal{A}$ be an IND-KOA attacker of ElGamal. That is $\mathcal{A}$ is called with a key A; interacts with a challenger $\mathcal{C}$ by sending two messages $x_1, x_2 \in G$ and receiving a challenge $(B, E) \in G^2$ (if the challenger is fair this is an encryption $(B, x_i \cdot K)$ of x_i for $i \in \{0, 1\}$ with $B = g^b$ and $K = A^b$); and finally outputs $j \in \{0, 1\}$. We call $\mathcal{A}$ successful (under a fair challenger) if $i = j$.

4 (i) Give an algorithm that calls $\mathcal{A}$ and solves the DDH in G. That is an algorithm with input $A = g^a$, $B = g^b$, and $C \in G$ and output TRUE if $C = g^{ab}$ and FALSE otherwise.

Hint: The algorithm should call $\mathcal{A}$ with a certain input, simulate the challenger (receive x_1, x_2 from $\mathcal{A}$ and send back a challenge), and output TRUE or FALSE depending on the output of $\mathcal{A}$.

4 (ii) Prove that your algorithm returns TRUE on input $A = g^a$, $B = g^b$, $C = g^{ab} \in G$ if $\mathcal{A}$ is successful.

4 (iii) Prove that your algorithm returns FALSE on input $A = g^a$, $B = g^b$, $C \neq g^{ab} \in G$ with probability $1/2$.

Hint: Choose the challenge randomly.

2 (iv) Assume $\mathcal{A}$ succeeds with probability p. What is the success probability of your algorithm if for an input $A = g^a$, $B = g^b$, C, in half of all cases $C = g^{ab}$ holds?

2 (v) Assume that DDH is hard in G and conclude that ElGamal is IND-KOA secure.

Exercise 4.3 (Security estimate). (8 points)

RSA is a public-key encryption scheme that can also be used for generating signatures. It is necessary for its security that it is difficult to factor large numbers (which are a product of two primes). The best known factoring algorithms achieve the following (heuristic, expected) running times:

method	year	time for n-bit integers
trial division	$-\infty$	$\mathcal{O}^{\sim}(2^{n/2})$
Pollard's $p-1$ method	1974	$\mathcal{O}^{\sim}(2^{n/4})$
Pollard's ϱ method	1975	$\mathcal{O}^{\sim}(2^{n/4})$
Pollard's and Strassen's method	1976	$\mathcal{O}^{\sim}(2^{n/4})$
Morrison's and Brillhart's continued fractions	1975	$2^{\mathcal{O}(1)n^{1/2}\log_2^{1/2} n}$
Dixon's random squares	1981	$2^{(\sqrt{2}+o(1))n^{1/2}\log_2^{1/2} n}$
Lenstra's elliptic curves method	1987	$2^{(1+o(1))n^{1/2}\log_2^{1/2} n}$
quadratic sieve		$2^{(1+o(1))n^{1/2}\log_2^{1/2} n}$
general number field sieve	1990	$2^{((64/9)^{1/3}+o(1))n^{1/3}\log_2^{2/3} n}$

It is not correct to think of $o(1)$ as zero, but for the following rough estimates just do it, instead add a $\mathcal{O}(1)$ factor. Factoring the 768-bit integer RSA-768 needed about 1500 2.2 GHz CPU years (ie. 1500 years on a single 2.2 GHz AMD CPU) using the general number field sieve. Estimate the time that would be needed to factor an n-bit RSA number assuming the above estimates are accurate with $o(1) = 0$ (which is wrong in practice!)

(i) for $n = 1024$ (standard RSA), 1

(ii) for $n = 2048$ (as required for Document Signer CA), 1

(iii) for $n = 3072$ (as required for Country Signing CA). 1

(iv) Now assume that the attacker has 1000 times as many computers and 1000 times as much time as in the factoring record. Which n should I choose to be just safe from this attacker? 2

Repeat the estimate assuming that only Pollard's ϱ method is available

(v) for $n = 1024$, 1

(vi) for $n = 2048$, 1

(vii) for $n = 3072$. 1

Remark: The statistics for discrete logarithm algorithms are somewhat similar as long as we consider groups $\mathbb{Z}_p^\times$. For elliptic curves (usually) only generic algorithms are available with running time $2^{n/2}$.

Exercise 4.4 (Hardcore bit for the discrete logarithm). (6 points)

Let G be a cyclic group of even order d with a generator g, and let $\omega = g^{d/2}$. Furthermore suppose that an algorithm for computing square roots in G is known. Let BitZero be a probabilistic algorithm that, given g^i, computes the least significant bit of i in expected polynomial time.

The square root algorithm is given g^{2i} with $0 \leq i < d/2$ and computes either the square root g^i or the square root ωg^i. Let Oracle be a probabilistic expected polynomial time algorithm that decides, which of the two square roots is g^i. [Note: This could be done by an oracle for the second least significant bit, $\mathrm{bit}_1(i)$, of the discrete logarithm of g^i, where $0 \leq i < d$.]

4 (i) Formulate an algorithm for the discrete logarithm that uses at most polynomially many calls to Oracle and otherwise uses expected polynomial time. (*Recall:* The algorithm gets as input g^i and should compute the discrete logarithm $\mathrm{dlog}_g(g^i) = i$ with $0 \leq i < d$.)

2 (ii) What implications does this have on the security of ElGamal encryption scheme?

References

MIHIR BELLARE & SHAFI GOLDWASSER (2008). Lecture Notes on Cryptography. URL `http://cseweb.ucsd.edu/~mihir/papers/gb.html`.

DOUGLAS R. STINSON (2006). *Cryptography - Theory and Practice*. Discrete Mathematics and its Applications. Chapman & Hall / CRC Press, Boca Raton FL, third edition. ISBN 1584885084, 593pp.