Electronic elections, winter 2007

MICHAEL NÜSKEN

2. Exercise sheet
Hand in solutions until Sunday, 9 December 2007.

Exercise 2.1 (RSA). (0+7 points)

Let's 'play' at RSA. Use the primes $p = 23$, $q = 79$ and $e = 17$.

(i) Compute secret and public key. +3

(ii) Encrypt $x = 1663$. +1

(iii) Decrypt $y = 366$. +1

(iv) Explain why encrypting x giving y and then decrypting the y giving z, always gives $z = x$. Go back to the theorem of Lagrange, Euler or Fermat. (Cite the theorem that you use.) +2

Exercise 2.2 (Blind signatures). (7 points)

It is sometimes required that a signature protocol between two parties A and B runs in such way that B signs *implicitly* a message m on behalf of A, but does not know the explicit message he is signing. Thus B cannot associate the signature to the user A. Such protocols are called *blind signatures* and play a key role in electronic cash schemes.

The following questions describe a blinding protocol based on the RSA signature scheme. Let B have the RSA public key (N, e) and secret exponent d. In order to receive blind signatures from B, party A uses an own *blinding key* $k \in \mathbb{Z}_N$ with $\gcd(k, N) = 1$.

(i) Suppose that A wants to have B sign the message $m \in \mathbb{Z}_N$, so that the signature can be verified but B cannot recover the value of m. Show that the following protocol fulfills the requirements for a blind signature scheme: 4

(a) A sends $M = m \cdot k^e \in \mathbb{Z}_N$ to B.

(b) B produces the signature $S(M) = M^d \in \mathbb{Z}_N$ and sends it to A.

(c) A recovers $S(m) = k^{-1} \cdot S(M) \in \mathbb{Z}_N$. Then $S(m)$ is a valid signature of m by B.

3 (ii) Let $N = p \cdot q$ where $p = 1000000000039$, $q = 10000001000029$ and $e = 2^{16} + 1 = 65537$. Compute the secret exponent d of B. Let $k \in \mathbb{Z}_N$ be a random number and $m \in \mathbb{Z}_N$ be the integer value of the ASCII text: *blinded*.

(a) Compute the blinded message M.

(b) Compute B's blinded signature $S(M)$ and also B's clear text signature $S'(m)$, using B's secret key d.

(c) Compute the clear text signature $S(m)$ such as A recovers it using k. Compare this signature to the value $S'(m)$ above.

Exercise 2.3 (Signed key exchange). (0+7 points)

From the web site `http://www.cryptool.com/` you can download the program `Cryptool` for the Windows operating system (a license for educational purposes has been granted).

The program offers all the algorithms discussed during this course, so that you can easily test them. The key length for public keys is limited to 768 due to (outdated) legal reasons.

In this exercise we shall simulate the start of an electronic conversation of Alice and Bob, using RSA authentification and subsequent key exchange.

+1 (i) Download the program `cryptool` onto your computer and intall it it to your hard drive. You will see a tutorial window and below that the work window. Using the latter window you are to solve the following tasks:

+1 (ii) Using *Key management* create 768 bit RSA keys for both Alice and Bob.

+2 (iii) In a new file compose a text to be signed. After that this text should be signed first using Alice's key (*extract signature*) and verified (by Bob) and then using Bob's secret key (and verified by Alice).

+2 (iv) Generate a random 128 bit key and save it to a second file. Encrypt this file using Alice's public key and subsequently decrypt it using her secret key. Now Alice and Bob share a common secret key.

+1 (v) Compose a short text (two to three sentences), save it to a third file and encrypt this file in Alice's stead using AES and the common secret key. Afterwards, decrypt the text in the name of Bob.

Note: Those parts of the protocols, that are not fully specified in the instructions of this exercise, are to be chosen by you (with ample comments).

Exercise 2.4 (Security estimate). (0+7 points)

RSA is a public-key encryption scheme that can also be used for generating signatures. It is neccessary 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.

In May 2005 the factorization of RSA-200 was completed. Up to now it is the largest general factored RSA number. It has 663 bits and the factorization took approximately one year of computation on large computer arrays.

Factoring the 663-bit integer RSA-200 needed about 165 1GHz CPU years (ie. 165 years on a single 1GHz Opteron CPU) using the general number field sieve. Estimate the time that would be needed to factor an n-bit RSA number with the general number field sieve 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

+1 (iv) Now, we assume that an attacker has an ultimate laptop[1] with 10^{20} times as much memory and 10^{40} times faster than a 1GHz Opteron CPU. (Actually such a thing is quite difficult to handle. Though its weight of only one kilogramm and the size of one liter is wonderful, it is quite difficult to carry around because it is not only much hotter than the sun but also emits much more radiation, not to speak of the difficulties when trying to 'read' the result.) What is the largest bit length that this attacker could manage in the same time?

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

+1 (v) for $n = 1024$,

+1 (vi) for $n = 2048$,

+1 (vii) for $n = 3072$.

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 2.5 (ElGamal signatures). (7+2 points)

We choose a prime number $p = 12347$, $q = 6173$, and the group $G = \langle g \rangle < \mathbb{Z}_p^\times$ with $g = 2^2$. We use $\alpha = 5432$ as the secret part of the key $K = (p, q, g, \alpha, a)$. The function $^*\colon \langle g \rangle \to \mathbb{Z}_q$ is defined by $^*(k \bmod p) = k \bmod q$ for $0 < k < p$. The hash function is essentially the identity: $\mathrm{hash}(x) = x \bmod q$. The message x to be signed consists of the least significant four digits of your student registration number. Use $\beta = 399$ as your random number from $\mathbb{Z}_{p-1}^\times$. *Example:* If the student registration number is 1234567, then $x = 4567$.

2+2 (i) Show: g generates G, or better: $h = 2$ generates $\mathbb{Z}_p^\times$. (Ie. $\#\langle h \rangle = p - 1$.)

2 (ii) Compute the public key $a = g^\alpha \in G$.

2 (iii) Compute the signature $\mathrm{sig}_K(x, \beta) = (x, b, \gamma)$.

1 (iv) Verify your signature.

[1] See `http://www.ar-tiste.com/qcomp_onion/jan2002/UltimateLaptop.htm` or `http://arstechnica.com/wankerdesk/01q2/limits/limits-1.html` or `http://www.edge.org/3rd_culture/lloyd/lloyd_index.html`.

Exercise 2.6 (Attacks on the ElGamal signature scheme). (8 points)

After prior failures princess Jasmin and Genie have been doing a lot of thinking and research. Genie has proposed to use the ElGamal signature scheme. They have chosen the prime number $p = 1\,334\,537$, $q = 23831$ and the generator $g = 3^{\frac{p-1}{q}} = 429997$. Here, $\#\langle g\rangle = q$. The function ${}^*\colon \langle g\rangle \to \mathbb{Z}_q$ is defined by ${}^*(k \bmod p) = k \bmod q$ for $0 < k < p$. The hash function is essentially the identity: $\mathrm{hash}(x) = x \bmod q$. The public key of the princess Jasmin is $a = 18\,013$.

(i) They have sent the message $(x, b, \gamma) = (7\,654, 89\,221, 810)$. Unfortunately, Genie was not very careful. He wrote down the number β somewhere and forgot to burn the scratch paper after calculating the signature. So Jaffar knows the number $\beta = 377$. Compute the secret key α. 4

(ii) Princess Jasmin has changed her secret key. She now has the public key $a = 580\,110$. This time Jaffar could not find the number β. Because of this he used an enchantment so that Jasmin's random number generator has output the same value for β twice in a row. This was the case for the messages $(2\,001, 349\,167, 15\,932)$ and $(234, 349\,167, 15\,308)$. Again compute Jasmin's secret key α. 4