Security on the Internet, summer 2007

MICHAEL NÜSKEN, DANIEL LOEBENBERGER

8. Exercise sheet
Hand in solutions until Thursday, 7 June 2007.

Exercise 8.1 (Exam date). (0 points)

The exam should take place between 10 July and 12 October 2007. Look up your calendar. Suggest a date. [0]

Exercise 8.2 (Repetition). (0 points)

Use the excursion week as a chance to repeat the course. [0]

Exercise 8.3 (ElGamal signatures). (7 points)

Compute an ElGamal signature for your student identification number represented in binary. Use $p = 467$ and $g = 3 \in \mathbb{Z}_p^\times$ and work in $G = \langle g \rangle$. For simplicity, we take the function $\mathrm{HASH}\colon \{0,1\}^* \to \mathbb{Z}_{233},\ x \mapsto (\sum_{0 \le i < |x|} x_i 2^i) \bmod 233$. (Eg. 18 translates to the string 10010 which in turn translates into the number $18 \bmod 233$.)

(i) Here $\#G = 233$ and thus $\exp_g\colon \mathbb{Z}_{233} \to G,\ a \mapsto g^a$ is an isomorphism. [Note that $166^2 = 3$ and thus $g^{233} = 1$. Since $g \neq 1 \ldots$] [1]

(ii) Setup: Compute Alice' public key with $\alpha = 9$. [1]

(iii) Sign: Sign the hash value of your student identification number. [3]

(iv) Verify: Verify the signature. [2]

Exercise 8.4 (Hash crisis). (11+3 points)

Read Arjen Lenstra's article on *Colliding X.509 Certificates* <`http://eprint.iacr.org/2005/067.pdf`>.

(i) What is the purpose of X.509 certificates? [2]

(ii) Where are they used? [1]

(iii) How does such a certificate ensure a connection between a secret key and identification information (name, birth, and so on) of a person? [2]

(iv) Who verifies this connection? [1]

(v) How can I check that this verification was done (assuming the verification authority is honest)? In other words, how can I check the certificate? [2]

(vi) What is the consequence of Lenstra's observation? [3]

(vii) Add further observations. [+3]

Exercise 8.5 (Security estimate). (6+1 points)

On 21 May 2007 a new factoring record was announced. Using computer clusters in various locations, NTT in Japan, Franke and Kleinjung at the university of Bonn, and A. Lenstra at the EPFL in Lausanne have successfully factored $2^{1039} - 1$. See links on the webpage.

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. 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 assuming the (strongest of the) above estimates are accurate with $o(1) = 0$ (which is wrong in practice!)

1 (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).

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

1 (iv) for $n = 1024$,
1 (v) for $n = 2048$,
1 (vi) 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 $2^{n/2}$.

+1 (vii) Why don't we take into account the latest record that was mentioned above?

Exercise 8.6 (Key exchange threats). (0+15 points)

We have considered the Diffie-Hellman key exchange: Given a group G consisting of powers of a generator g, so $G = \{1, g, g^2, \ldots, g^{\#G-1}\}$ such that the discrete log problem is difficult, ie. given $h \in G$ there is no efficient (ie. randomized polynomial time) algorithm to determine i with $h = g^i$. To fix a shared secret key, Alice sends g^a and Bob sends g^b. Then both can compute the shared key g^{ab}.

Protocol 8.7. Diffie-Hellman key exchange.

1. Alice chooses $a \in \mathbb{N}_{<\#G}$ and computes g^a.
2. Bob chooses $b \in \mathbb{N}_{<\#G}$ and computes g^b.
3. Alice computes $(g^b)^a = g^{ab}$.
4. Bob computes $(g^a)^b = g^{ab}$.

$$\xrightarrow{\quad g^a \quad}$$
$$\xleftarrow{\quad g^b \quad}$$

Now both can use g^{ab} to derive common secrets for the subsequent message exchanges. What if Wilma puts herself in the middle? She will have a common secret g^{aw} with Alice and a common secret $g^{w'b}$ with Bob, and as long as she continues to pass all messages on, neither Bob nor Alice will notice anyhting apart possibly from a slighlty slower connection. So we modify this.

Protocol 8.8. Signed and acknowledged Diffie-Hellman key exchange.

1. Alice chooses $a \in \mathbb{N}_{<\#G}$, computes g^a and signs $[\text{'Alice'}, g^a]$.
2. Bob chooses $b \in \mathbb{N}_{<\#G}$, computes g^b and signs $[\text{'Bob'}, g^b]$.
3. Alice computes $(g^b)^a = g^{ab}$ and a hash.
4. Bob computes $(g^a)^b = g^{ab}$ and a hash.

$$\xrightarrow{\quad [\text{'Alice'}, g^a]_{\text{Alice}} \quad}$$
$$\xleftarrow{\quad [\text{'Bob'}, g^b]_{\text{Bob}} \quad}$$
$$\xrightarrow{\quad \text{hash}(0, g^{ab}) \quad}$$
$$\xleftarrow{\quad \text{hash}(1, g^{ab}) \quad}$$

Sorry, we forgot to be polite. We should first say Hello, shouldn't we?

Protocol 8.9. Polite Diffie-Hellman key exchange with a cookie.

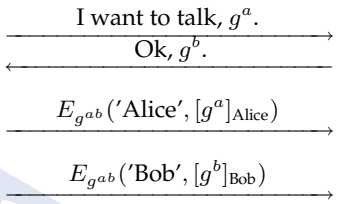
1. Alice wants to talk.
2. Bob agrees and chooses a cookie c, which is a suitably random number, for example, the hash value of *her* IP address and some fixed secret of Bob. (It's nice if the number is deterministically determined!)
3. Alice chooses $a \in \mathbb{N}_{<\#G}$, computes g^a and signs $[\text{'Alice'}, g^a]$.
4. Bob chooses $b \in \mathbb{N}_{<\#G}$, computes g^b and signs $[\text{'Bob'}, g^b]$.
5. Bob computes $(g^a)^b = g^{ab}$ and a hash.
6. Alice computes $(g^b)^a = g^{ab}$ and a hash.

$$\xrightarrow{\quad \text{I want to talk} \quad}$$
$$\xleftarrow{\quad \text{Ok, I listen for cookie } c. \quad}$$
$$\xrightarrow{\quad c, [\text{'Alice'}, g^a]_{\text{Alice}} \quad}$$
$$\xleftarrow{\quad [\text{'Bob'}, g^b]_{\text{Bob}}, \text{hash}(1, g^{ab}) \quad}$$
$$\xrightarrow{\quad \text{hash}(0, g^{ab}) \quad}$$

Here is a further polite variant.

Protocol 8.10. Modified Diffie-Hellman key exchange.

1. Alice chooses $a \in \mathbb{N}_{<\#G}$, computes g^a.
2. Bob chooses $b \in \mathbb{N}_{<\#G}$, computes g^b.
3. Alice computes $(g^b)^a = g^{ab}$ and uses it to encrypt her name and a signature to her share g^a.
4. Bobd computes $(g^a)^b = g^{ab}$ and uses it to encrypt his name and a signature to his share g^b.



Consider each of the above protocols in the following questions. (Be brief, but don't forget the essential arguments.)

+2 (i) *Woman in the middle*: Try to put Wilma in the middle. What happens?

+2 (ii) *Mutual authentication*: Examine which of the given protocols ensure that Alice' partner is Bob.

+2 (iii) *Perfect Forward Security*: Next, suppose that the Beagle Boys intercepted the conversation between Alice and Bob. Then after the conversation is terminated the Beagle Boys take over Alice' and Bob's entire equipment including their secret keys. Will they be able to read what Alice and Bob told each other?

+2 (iv) *Denial of Service*: Daniel is a weird person that only wants to prevent say Bobs' computer to do good work. So he floods Bob with tons of requests. For each of these requests Bob's computer is forced to compute and send an answer. Consider vaguely the effort which Daniel and Bob have to spend for their first messages and vote for the 'best' protocol.

+2 (v) *Endpoint Identifier Hiding*: Eve does not want to be spotted, so she only listens on the conversation. If she can detect who the partners are, this is already valuable information for her. Which protocols hide the identity of Alice and/or Bob?

+2 (vi) *Live Partner Reassurance*: Romeo likes repetions and so after listening to a conversation, he calls Bob with replayed messages from the overheared talk making him think he is Alice. (Imagine this could be successfully done when you log in to your home banking account!) Examine the given protocols under this attack.

+3 (vii) Devise a protocol that Romeo cannot trick. (Do not forget to argue!)

+0 (viii*) Devise a protocol that is not vulnerable to any of these attacks.