

# Cryptographic passports & biometrics, summer 2009

MICHAEL NÜSKEN, KONSTANTIN ZIEGLER

## 6. Exercise sheet

Hand in solutions until Monday, 8 June 2009.

Any claim needs a proof or argument.

**Exercise 6.1** (Lagrange's theorem).

(0+11 points)

We have seen that in a commutative group  $G$  we have  $x^{\#G} = 1$  for  $x \in G$ . There is a more general version of the theorem which says more and works also for noncommutative groups.

**Theorem** (Lagrange). *Suppose  $G$  is a finite group.*

(a) *If  $H$  is a subgroup of  $G$ , then  $\#H$  divides  $\#G$ .*

(b) *If  $x \in G$  then  $x^{\#G} = 1$  in  $G$ .*

We are going to prove the first part. Let  $a \in G$  be arbitrary group elements. We consider the so-called *cosets*  $aH = \{ah \mid h \in H\}$ .

(i) Prove that there is a  $c \in G$  such that  $a \in cH$ .

+1

(ii) Consider the map  $\lambda: H \rightarrow aH, x \mapsto ax$ . Prove that it is bijective.

+1

(iii) Conclude that  $\#(aH) = \#H$  is independent of  $a$ .

+1

(iv) Suppose we are given two group elements  $a, b \in G$ . Then only the following two cases are possible:

+2

- $aH = bH$ , or
- $aH \cap bH = \emptyset$ .

In other words: it never happens that  $aH$  and  $bH$  have some but not all elements in common.

Prove this. [Hint: Suppose  $x \in aH \cap bH$  (so we are not in the second case) and show that then  $aH = bH$  (this is the first case).]

(v) Conclude that  $G$  is the disjoint union of all cosets.

+1

(vi) Conclude that  $\#H$  divides  $\#G$ .

+1

We derive the second part from the first in the following steps:

(vii) Consider  $\langle a \rangle = \{\dots, a^{-2}, a^{-1}, 1, a, a^2, \dots\}$ . Prove that this is a subgroup of  $G$ . It is called the *subgroup generated by  $a$* . +1

(viii) Now let  $n$  be the *order* of  $a$ , i.e.  $a^n = 1$  and  $a^k \neq 1$  for all  $0 < k < n$ . Prove that  $\langle a \rangle = \{1, a, \dots, a^{n-1}\}$  and in particular  $\#\langle a \rangle = n$ . +2

+1

(ix) Conclude that  $a^{\#G} = 1$ .

### Exercise 6.2 (DSA Practice).

(11 points)

In this exercise you will make practical computations with the DSA algorithm, using *real life* key sizes. Use a computational system of your choice making sure that you can deal with modular arithmetic involving very long integers; we advice to use MuPAD which runs on all b-bit computers. In any case, provide the essential program code as part of your solution.

1

(i) Generate a random prime number  $\ell$  with exactly 160 bits.

2

(ii) Generate a prime  $p$  with exactly 1024 bits such that  $\ell$  divides  $p - 1$ .

1

(iii) Find a  $g \in \mathbb{Z}_p^\times$  which has the exact order  $\ell$ . Let  $G = \langle g \rangle \subset \mathbb{Z}_p^\times$  be the cyclic group with  $\ell$  elements generated by  $g$ .

For simplicity we use a trivial hash function and for  $b^*$  we simply take the integer  $b$  (in the interval  $[0, p - 1]$ , i.e. forgetting that it's actually in  $\mathbb{Z}_p$ ) reduced modulo  $\ell$ .

1

(iv) Let  $a < p$  be a random number and  $y = g^a \in G$ . We shall consider  $a$  to be Alice's secret key and  $y$  her public key.

2

(v) Let  $m \in \mathbb{Z}_\ell$  be the integer value of the ASCII text: DSA\_for\_real (note the two blanks in the text!). Using a random number  $k \in \mathbb{Z}_\ell$  produce a DSA signature  $S(m) = (m, x, b)$  on the message  $m$  on behalf of Alice.

1

(vi) Let Bob know the public key  $(p, g, y)$ . Verify the signature  $S(m)$  on behalf of Bob.

1

(vii) Let  $m'$  be the integer value of the ASCII text: *The Lord of the Rings has no secrets*. Can you produce a DSA signature of this text using the same setting as above? If no what additional steps are required?

2

(viii) The DSA system can be attacked in two different ways:

(a) By solving the discrete logarithm problem in the group  $G$  with  $\ell$  elements by applying the baby-step giant-step algorithm in this group.

- (b) By solving the general discrete logarithm problem in  $\mathbb{Z}_p^\times$  using the up to date Number Field Sieve. The complexity of this method is given by the function:

$$L(p) = 2\left(\frac{64}{9}\right)^{\frac{1}{3}} \cdot (\log_2 p)^{\frac{1}{3}} (\log_2 \log_2 p)^{\frac{2}{3}}.$$

Compare the two estimated times when  $p \sim 2^{1024}$  and  $\ell \sim 2^{160}$ .

**Exercise 6.3** (Attacks on the ElGamal signature scheme). (4 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  $p = 1\,334\,537$  and the generator  $g = 16$ . The public key of the princess Jasmin is  $a = 605\,828$ .

- (i) They have sent the message  $(x, b, \gamma) = (7\,654, 642\,260, 4\,427)$ . Unfortunately, Genie was not very careful. He wrote down the number  $\beta$  somewhere and forgot to burn the piece of paper after calculating the signature. Now Jaffar knows the number  $\beta = 377$ . Compute the secret key  $\alpha$ . [2]
- (ii) Princess Jasmin has changed her secret key. She now has the public key  $a = 436\,700$ . This time Jaffar could not find the number  $\beta$ . Because of this he used an enchantment so that Jasmin's random number generator has output the same value for  $\beta$  twice in a row. This was the case for the messages  $(2\,008, 14\,694, 21\,273)$  and  $(234, 14\,694, 10\,507)$ . Now compute Jasmin's secret key  $\alpha$ . [2]

**Exercise 6.4** (ElGamal-signatures and hash functions). (4 points)

Consider the ElGamal signature scheme with a hash function  $h$ . Assume that the attacker can find, for a given message  $x$ , another message  $y$  with the same hash value  $h(x) = h(y)$ . Prove that the attacker can then break the scheme. Conclude a theorem: "If ElGamalSign( $h$ ) is secure, then  $h \dots$ ". [4]

**Exercise 6.5** (Security estimate). (10 points)

The ElGamal signature scheme works over some publicly known group of (often prime) order  $\ell$ , where  $\ell$  has length  $n$ . In many cases this is a subgroup of some  $\mathbb{Z}_p^\times$  with another (larger) prime  $p$ ; then  $\ell \mid (p - 1)$ .

- (i) Show that the time for signing a message  $m$  is polynomial in  $n$ . For the generation of the hash value you may assume the call to a subroutine with time polynomial in  $n$ . [2]

For the security of the signature scheme it is necessary it is difficult to compute a discrete logarithm in the group and also, if applicable, in the surrounding group  $\mathbb{Z}_p^\times$ . The best known discrete logarithm algorithms achieve the following (heuristic, expected) running times:

method	year	time for a group size of $n$ -bit
brute force (any group)	$-\infty$	$\mathcal{O}^\sim(2^n)$
Baby-step Giant-step (any group)	1971	$\mathcal{O}^\sim(2^{n/2})$
Pollard's $\varrho$ method (any group)	1978	$\mathcal{O}(n^2 2^{n/2})$
Pohlig-Hellman (any group)	1978	$\mathcal{O}^\sim(2^{n/2})$
Index-Calculus for $\mathbb{Z}_p^\times$	1986	$2^{(\sqrt{2}+o(1))n^{1/2} \log_2^{1/2} n}$
Number-field sieve for $\mathbb{Z}_p^\times$	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. Estimate the time that would be needed to find a discrete logarithm in a group whose order has  $n$ -bits assuming the (strongest of the) above estimates are accurate with  $o(1) = 0$  (which is wrong in practice!)

- (ii) for  $n = 1024$  (standard size),
- (iii) for  $n = 2048$  (as required for Document Signer CA),
- (iv) for  $n = 3072$  (as required for Country Signing CA).

Repeat the estimate assuming that for the given group only Pollard's  $\varrho$  method is available, for example in case the group is a  $\ell$ -element subgroup of  $\mathbb{Z}_p^\times$  or an elliptic curve,

- (v) for  $n = 160$ ,
- (vi) for  $n = 200$ ,
- (vii) for  $n = 240$ .

In April 2001 Reynald Lercier reported (<http://perso.univ-rennes1.fr/reynald.lercier/file/nmbrJL01a.html>) that they can solve a discrete logarithm problem modulo a 397-bit prime  $p$  within 10 weeks on a 525MHz computer.

- (viii) Which bit size for the prime  $p$  is necessary to ensure that they cannot solve the DLP problem in  $\mathbb{Z}_p^*$  given —say— 10'000 10GHz computers and 1 year (disregarding memory requirements).

[Note: The record for computing discrete logs in  $\mathbb{F}_{2^n}^\times$  lies at  $n = 613$ , see Antoine Joux <http://perso.univ-rennes1.fr/reynald.lercier/file/nmbrJL05a.html>.]