TTM4135 Worksheet 6: DLOG and Digital Signatures

Tjerand Silde and Emil August Hovd Olaisen
{tjerand.silde, emil.august.olaisen}@ntnu.no

Spring 2026

1. Review the definitions of the following concepts:
   (a) ElGamal cryptosystem
   (b) digital signature
   (c) existential forgery and selective forgery
   (d) Schnorr, DSA, ECDSA

2. Let $p = 43$. Verify that $g = 3$ is a generator of $\mathbb{Z}_p^*$. Suppose that Alice and Bob execute the Diffie–Hellman key exchange protocol with Alice's input being $a = 5$ and Bob's input $b = 13$. Show that both Alice and Bob will compute the same shared secret.

3. Suppose that Alice has a public key $y = 5, g = 2, p = 11$ for the ElGamal encryption algorithm. Here $g = 2$ is a generator for $\mathbb{Z}_{11}^*$. Compute a valid ciphertext for the message $M = 3$ intended for Alice, showing the steps required.

4. Compare the efficiency of the ElGamal cryptosystem in $\mathbb{Z}_p^*$ and the RSA cryptosystem with modulus n. Assume that the size of the modulus p and n is the same in each case. Compare:
   - the cost of key generation;
   - the computation required for encryption;
   - the computation required for decryption;
   - the size of the public keys and ciphertexts.

5. In 2019 elections for the Moscow city parliament an electronic voting system used a simple variant of ElGamal encryption to protect a user vote M. For the public key $K = (K_1, K_2, K_3) = (y_1, y_2, y_3) = (g^{x_1}, g^{x_2}, g^{x_3})$ the following details the encryption algorithm. First the encrypting party chooses random k_1, k_2, k_3 and computes the following values.

   $c_1 = E(M, K_1) = (g^{k_1} \bmod p, M \cdot y_1^{k_1} \bmod p) = (a_1, b_1)$

   $c_2 = E(a_1, K_2) = (g^{k_2} \bmod p, a_1 y_2^{k_2} \bmod p) = (a_2, b_2)$

   $c_3 = E(a_2, K_3) = (g^{k_3} \bmod p, a_2 y_3^{k_3} \bmod p) = (a_3, b_3)$

   Finally the ciphertext is the 4-tuple $c = (b_1, b_2, a_3, b_3)$. Note that c_1, c_2 and c_3 are ordinary ElGamal ciphertexts.

   For unclear reasons, the implementation used a prime p with only 256-bits for computations in $\mathbb{Z}_p^*$. You can read the details of the analysis here: https://arxiv.org/abs/1908.05127.

   (a) Explain how the message M can be recovered from c given the private key (x_1, x_2, x_3)
   (b) How long is the private key?
   (c) If an attacker can find discrete logarithms in $\mathbb{Z}_p^*$ in time T, how long will take the same attacker to find the message M?

6. Suppose an attacker can break the hash function H used to form a digital signature (RSA or DSA) by finding collisions. How can this lead to attacks on the signature? Is this attack existential or selective?

7. Suppose that RSA signatures are used with a hash function that is not one-way (that is, the attacker can invert the hash function). Show how an existential forgery is possible against such a signature: an attacker can form valid signatures from e and n alone.

8. (a) Show that the verification equation for DSA signatures works.
   (b) Similarly check that the verification equation works for ECDSA signatures.

9. Suppose the parameters $p = 23$, $q = 11$, $g = 3$ are used for the DSA signature.
   (a) Show that g has order q as required.
   (b) If the private key is $x = 5$, what is the public key y?
   (c) Compute a valid signature for a message m whose hash value is assumed to be $\mathsf{H}(M) = 8$.
   (d) Show that the verification equation works for your signature.

10. Compare the efficiency of DSA signatures, ElGamal signatures, and RSA signatures assuming that the modulus size is 2048 bits in each case, and that the prime q in DSA signatures is of length 256 bits. Compare:

    - the cost of key generation;
    - the computation required for signature generation;
    - the computation required for signature verification;
    - the size of the public keys and signatures.

11. Suppose that the same value of the random k is used to generate two different DSA signatures. Show that this is sufficient for an attacker to find the private signing key. (This was the attack used to break the software verification on the Sony PlayStation 3 in 2010 because their implementation used a fixed k. Sony used the elliptic curve version: https://arstechnica.com/gaming/2010/12/ps3-hacked-through-poor-implementation-of-cryptography.)

12. Recall that ECDSA signatures consist of a pair of value (r, s)
    (a) Show that if (r, s) is valid ECDSA signature for a message M then $(r, -s)$ is also a valid signature for M.
    (b) Show that the ECDSA signature verification equation will be satisfied by the values $(r, s) = (0, 0)$ for *any* message M, as long as other checks are omitted. You can read about why this property led to a huge vulnerability in Java cryptography known as *psychic signatures*.