TTM4135 Worksheet 5: RSA

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) trapdoor one-way function;
   (b) RSA equations;
   (c) prime number theorem;
   (d) square-and-multiply algorithm;
   (e) Håstad's attack;
   (f) Miller's theorem;
   (g) discrete logarithm problem;
   (h) generator of $\mathbb{Z}_p^*$;
   (i) Diffie–Hellman key exchange.

2. Suppose that an RSA public key is chosen with primes $p = 13$ and $q = 17$. Suppose that the public key is $e = 5$.
   (a) Find the value of d.
   (b) Find the encryption of $m_1 = 4$ and $m_2 = 13$.
   (c) Decrypt the ciphertexts and verify that the correct value is recovered.

   Solution:
   (a) The modulus is $n = pq = 221$. Then $d = e^{-1} \bmod \phi(n) = 5^{-1} \bmod 12 \cdot 16 = 5^{-1} \bmod 192$. Using Euclidean algorithm we can find that $5^{-1} \bmod 192 = 77$.

   (b) For the message $m_1 = 4$ the ciphertext is $c_1 = m_1^e \bmod n = 4^5 \bmod 221 = 140$.

   For the message $m_2 = 13$ the ciphertext is $c_2 = m_2^e \bmod n = 13^5 \bmod 221 = 13$. Notice that this illustrates that it is quite possible for messages to be sent to themselves in RSA. This will happen with negligible probability in practice.

   (c) To decrypt we compute $m_1 = c_1^d \bmod n = 140^{77} \bmod 221 = 4$. Similarly $m_2 = c_2^d \bmod n = 13^{77} \bmod 221 = 13$.

3. Suppose that RSA encryption uses a modulus n of 3000 bits. Assuming that the square-and-multiply method is used for exponentiation, compare the computational cost of encryption, measured in the number of squarings and the number of multiplications, in the following cases:
   (a) $e = 3$
   (b) $e = 2^{16} + 1$
   (c) e is chosen randomly between 0 and n.

How much computation is required for decryption in each case?

Solution: Regard the exponent e as a binary string. If there are s bits in the exponents and t of these are '1' bits. The square-and-multiply method uses $s-1$ squarings and $t-1$ multiplications.

	squarings	multiplications
$e = 3$	1	1
$e = 2^{16}+1$	16	1
e random	2999	1499

The last row is an average estimate. Note that e will usually have around 3000 bits if chosen randomly.

The decryption computation depends on the value of d. In all cases this can be regarded as being randomly chosen in the range $0 < d < \phi(n)$. Note that $\phi(n)$ also has approximately 3000 bits. Therefore the decryption computation will always take similar computation to the last row in the table.

4. Suppose that the same message m has been encrypted for three recipients with different RSA moduli: 205, 319 and 391. Each recipient uses public exponent $e = 3$. Suppose also that no random padding has been added. The three ciphertexts found are: 180, 43 and 218 respectively.

   Demonstrate Håstad's CRT attack by finding the value of m without making use of the factorization of the moduli.

Solution: We have three modular equations:

$$\begin{aligned} m^3 &\equiv 180 \pmod{205} \\ m^3 &\equiv 43 \pmod{319} \\ m^3 &\equiv 218 \pmod{391} \end{aligned}$$

These can be solved using the Chinese remainder theorem. Let

$$\begin{aligned} y_1 &= (319 \cdot 391)^{-1} \bmod 205 = 129 \\ y_2 &= (205 \cdot 391)^{-1} \bmod 319 = 115 \\ y_3 &= (205 \cdot 319)^{-1} \bmod 391 = 4 \end{aligned}$$

Then $m^3 = 180 \cdot (319 \cdot 391 \cdot y_1) + 43 \cdot (205 \cdot 391 \cdot y_2) + 218 \cdot (205 \cdot 319 \cdot y_3) \bmod 25569445 = 1000$. Finally the message can be found by extracting the ordinary cube root to obtain $m = 10$.

5. Consider RSA with values $p = 23$, $q = 31$, $n = 713$ and $d = 233$. Suppose the received ciphertext is $c = 266$.

   Examine the faster decryption method using the CRT, using these values:

   (a) Compute $m_p = c^{d \bmod p-1} \bmod p$.

   (b) Similarly compute m_q.

   (c) Combine these results using the CRT to decrypt the ciphertext.

Solution:

$d \pmod{p-1} = 233 \pmod{22} = 13$

$d \pmod{q-1} = 233 \pmod{30} = 23$

(a) $m \bmod p = c^{d \bmod p-1} \bmod p = 266^{13} \bmod 23 = 13^{13} \bmod 23 = 8$.

(b) $m \bmod q = c^{d \bmod q-1} \bmod q = 266^{23} \bmod 31 = 18^{23} \bmod 31 = 7$.

(c) $m = 8 \cdot 31 \cdot r_1 + 7 \cdot 23 \cdot r_2 \bmod 713$.

Here $r_1 = 31^{-1} \bmod 23 = 8^{-1} \bmod 23 = 3$ and $r_2 = 23^{-1} \bmod 31 = 27$.

$$\begin{aligned} m &= 8 \cdot 31 \cdot 3 + 7 \cdot 23 \cdot 27 \bmod 713 \\ &= 744 + 4347 \bmod 713 \\ &= 31 + 69 \\ &= 100 \end{aligned}$$

6. Suppose that an attacker obtains an RSA private key $d = 233$ and also has the public key $e = 17$ and $n = 713$. Apply Miller's algorithm to factorize n.

Solution: First write $ed-1 = 2^v \cdot u$ for u odd. Since $ed-1 = 3960 = 2^3 \cdot 495$ we have $v = 3$ and $u = 495$.

Choose $a = 2$ and compute $2^u \bmod n = 1$. Unfortunately we do not get a non-trivial square root this time since all later squares of this must also be 1.

Choose $a = 3$ and compute $3^u \bmod n = 185$. We are lucky and this is not ± 1 so we can find a non-trivial square root of 1 by squaring.

$185^2 \bmod n = 1$ so $a = 185$ is a non-trivial square root of 1.

Therefore we can find a factor of n by computing $\gcd(a-1, n) = \gcd(184, 713) = 23$. The other factor is then $713/23 = 31$. Therefore we have factorized the modulus: $713 = 23 \cdot 31$.

7. In this question we show that $f(x) = x^2 \bmod n$ is a trapdoor one-way function, when $n = pq$ and $p \bmod 4 = q \bmod 4 = 3$ and p and q are different primes. We do this in three steps.
   (a) Explain why we know that $(p+1)/4$ is an integer, and explain why we know that if x has a square root in $\mathbb{Z}_p^*$ then show that $x^{(p+1)/4} \bmod p$ is a square root of x in $\mathbb{Z}_p^*$.
   (b) Use the part above to show that if p and q are known, then a square root modulo n can be efficiently computed (assume we have an efficient exponentiation function). Thus p and q are a trapdoor to invert f.
   (c) Now suppose that there exists an algorithm A that finds integers x, y, z such that $x \equiv y^2 \equiv z^2 \bmod n$ with $z \not\equiv y \bmod n$ and $z \not\equiv -y \bmod n$, then this can be used to factorize n. Hence deduce that inverting f is as hard as factorizing n, and that f is one-way.

Solution:

(a) Note that only half of the integers $\{1, 2, \ldots, p-1\}$ are squares of some other value modulo p. Since $x \equiv y^2 \bmod p$ for some y,

$$\begin{aligned} \left(x^{\frac{p+1}{4}}\right)^2 \bmod p &= x^{\frac{p+1}{2}} \bmod p \\ &= x^{\frac{p-1}{2}} \cdot x \bmod p \\ &= y^{p-1} \cdot x \bmod p \\ &= x \end{aligned}$$

where the last step uses Fermat's theorem.

(b) Given any x which is a square modulo n, the above algorithm can be used separately for both p and q. So we can efficiently obtain a and b so that $a^2 \bmod p = x \bmod p$ and $b^2 \bmod q = x \bmod q$. Now we can use the Chinese Remainder Theorem to solve

$$\begin{aligned} y &\equiv a \bmod p \\ y &\equiv b \bmod q \end{aligned}$$

Thus we have $y^2 \bmod p = x \bmod p$ and $y^2 \bmod q = x \bmod q$. In other words $y^2 - x$ is a multiple of both p and q so is a multiple of $n = pq$. Thus we have $y^2 \bmod n = x \bmod n$, meaning y is a square root.

(c) Since $x \equiv y^2 \bmod n$ and $x \equiv z^2 \bmod n$ we have $y^2 - z^2 \equiv 0 \bmod n$, so $(y-z)(y+z)$ is a multiple of n. If $(y-z)$ or $(y+z)$ is a multiple of n then we have either $z \equiv y \mod n$ or $z \equiv -y \mod n$. If neither $(y-z)$ nor $(y+z)$ is a multiple of n then one must be a multiple of p and the other must be a multiple of q so that $\gcd(y-z, n) > 1$ will efficiently compute a factor of. n.

We summarize the algorithm for computing square roots. If we can compute p, q from pq, then we select random $x \in \mathbb{Z}_{pq}^*$ until $x = (x^{\frac{p+1}{4}})^2$ in $\mathbb{Z}_p^*$ and $x = (x^{\frac{q+1}{4}})^2$ in $\mathbb{Z}_q^*$. This will happen $1/4$ of the time. We denote $a = x^{\frac{p+1}{4}} \bmod p, b = x^{\frac{q+1}{4}} \bmod q$. We use $(p+q)^{-1}$ in $\mathbb{Z}_{pq}^*$ to compute four different square roots of x in $\mathbb{Z}_{pq}^*$:

$$(p+q)^{-1}(aq+bp), (p+q)^{-1}(aq-bp), (p+q)^{-1}(-aq+bp), (p+q)^{-1}(-aq-bp).$$

We know that the first two numbers satisfy the criteria from the algorithm A.

8. It is common in the ElGamal encryption algorithm for users to share the modulus p and generator g. Why is it not possible for users to share the same modulus n in the RSA cryptosystem?

Solution: At one level we can simply say that users generate their own moduli. If user Alice tries to use Bob's modulus n then Alice will not be able to factorize n and will be unable to generate a valid (e, d) pair. Thus it just won't work.

More interesting is the case that a trusted party could generate n and many valid (e, d) pairs and distribute them to different parties. This works, but it is totally insecure. We know from Miller's algorithm that knowledge of any valid (e, d) pair is sufficient to factorize n. Therefore Alice could use her own (e_A, d_A) pair to obtain the factors of n and can then obtain Bob's private d_B value from the public e_B and n values.

9. The Diffie–Hellman protocol can be defined, but is not necessarily secure, in any group.
   (a) Define Diffie–Hellman in the *additive* group modulo p for some prime p, instead of the multiplicative group where it is usually defined. Would this be secure for sufficiently large values of p?
   (b) Write down the equations for Diffie-Hellman on elliptic curves (ECDH) using the notation from the lecture slides. Show that for this to be secure the elliptic curve discrete log problem must not be an easy problem.

Solution:

(a) We define a generator q for the additive group $\mathbb{Z}_p$, any non-zero integer will in fact generate $\mathbb{Z}_p$. The key exchange would proceed as follows:

- Alice sends ag to Bob.
- Bob sends bg to Alice.

- Bob computes $b(ag) = bag$.
- Alice computes $a(bg) = bag$.

Computing b from bg is a simple matter of inverting g in $\mathbb{Z}_p$, that is $b = (bg)g^{-1}$. The Euclidean algorithm is sub-linear in p (it is in fact $O(\log(p))$), it has the same asymptotic runtime as a square and multiply exponentiation algorithm. Thus, it takes the same effort to use the scheme as to break it, and it provides no security at all.

(b) Suppose that the point G is a generator for the elliptic curve group. Then

- Alice sends $[a]G$ to Bob.
- Bob sends $[b]G$ to Alice.
- Bob computes $[b]([a]G) = [ba]G$.
- Alice computes $[a]([b]G) = [ba]G$.

If an attacker can compute discrete logs on the elliptic curve then they can find a from $[a]G$ and compute $[a]([b]G) = [ba]G$ in the same way as Alice.

10. In the ElGamal cryptosystem Alice and Bob have public keys g^{x_A} and g^{x_B} respectively, with corresponding private keys x_A and x_B. When Alice wants to send a message confidentially to Bob she chooses an ephemeral private key a and constructs a new shared secret g^{ax_B}.

Consider the following variant of the ElGamal cryptosystem. Instead of choosing a new random value a, Alice simply computes the static Diffie–Hellman value $X = (y_B)^{x_A} \bmod p$ and sends $c = mX \bmod p$ to Bob as the ciphertext.

(a) How does Bob decrypt?

(b) What could be the advantages of such a scheme as compared with normal ElGamal encryption?

(c) Show that this scheme is, unfortunately, completely insecure against a known plaintext attack.

Solution:

(a) The recipient needs to compute $X = (y_A)^{x_B} \bmod p$ and then $m = cX^{-1}$.

(b) One advantage is that the new cryptosystem is computationally cheaper since only one exponentiation is required. Furthermore, ciphertexts are shorter than in ElGamal encryption.

(c) Knowing one ciphertext and corresponding plaintext message allows an attacker to obtain X and therefore all plaintexts. This is enough to say that the cryptosystem should be avoided, but another disadvantage is that the recipient needs to know the public key of the sender.