CHALMERS — GÖTEBORGS UNIVERSITET

Cryptography

TDA352 (Chalmers) - DIT250 (GU)

11 April 2017, 8:30 - 12:30

No extra material is allowed during the exam except for pens and a simple calculator (not smartphones). *No other electronic devices allowed.* Your answers in the exam must be written in plaintext *English.* Your language skills will not be graded (but of course we cannot grade your answer if we do not understand it), so try to give *clear answers.* You can draw diagrams to explain concepts like security games, cryptographic protocols or simple ciphers modes of operation. In any case, your thoughts and ways of reasoning must be clearly understood!

Teacher: Elena Pagnin
Examiner: Aikaterini Mitrokotsa
Questions during the exam: Carlo Brunetta (phone 072 3515913)
(Elena Pagnin, phone: 072 9681552)
Inspection of exam: See web page for announcement.

The exam has 4 *topics* and *some bonus questions* to gain extra points.
The total number of points is 100 points (+ 8 bonus points).
Grades are :

CTH Grades:	50-64 → 3,	65-89 → 4,	90 or above → 5
GU Grades:	50-89 → G,	90 or above → VG	

Good luck!

Symmetric Ciphers (20p)

1. Let $(\mathbf{E}, \mathbf{D})$ be a secure block cipher.

   (a) Describe the ECB mode of operation (encryption and decryption). **(5p)**

   In ECB (Electronic Codebook) mode, the message M is divided into blocks $\{m_i\}$ of the same length. (1p) Each block m_i is encrypted separately (possibly in a parallel fashion) using $\mathbf{E}$ and the key $\mathbf{sk}$ (the same for all the messages). If we denote by c_i the i-th encrypted block, we can describe ECB as:

   $$c_i = \mathbf{E}(\mathbf{sk}, m_i) \quad (2p), \qquad m_i = \mathbf{D}(\mathbf{sk}, c_i) \quad (2p)$$

   (b) Show that ECB is not semantically secure if the length of the message is at least two blocks. **(9p)**
   (Hint: start by describing the semantic security game and show a successful attack strategy in the case $M = (m_0, m_1)$ is a two-block message).

   In the semantic security game, the adversary has to submit to the challenger two distinct messages of the same length. (1p) The challenger then returns an encryption of one of the two messages. (1p) The adversary wins the game if he (correctly) guesses which is the message corresponding to the received ciphertext. (2p)
   Let the adversary choose the two-block messages $M_0 = (m, m), M_1 = (m, \hat{m})$ with $m \neq \hat{m}$, (2p) and submit them to the challenger. Let $C = (c_1, c_2)$ denote the ciphertext returned by the challenger, which is an encryption of either M_0 or M_1. If $c_1 = c_2$, the adversary outputs $b' = 0$ as a guess that the challenger chose to encrypt the message M_0. (1p) This guess is correct, since ECB encryption is deterministic, i.e., identical blocks result in identical ciphertexts. (1p) In case $c_1 \neq c_2$, the adversary outputs $b' = 1$, since M_1 has distinct blocks by construction. (1p) This proves that ECB is not semantic secure.

   (c) *Bonus question: Give the definition of a secure Pseudo-Random Generator.* *(3 bonus points)*
   A function $\mathcal{G} : \{0,1\}^l \to \{0,1\}^n$ with $l \ll n$, (1p) is a *secure PseudoRandom Generator* if for every efficient statistical test $\mathcal{T}$, it holds that

   $$|\text{Prob}(\mathcal{T}(\mathcal{G}(u)) = 1) - \text{Prob}(\mathcal{T}(\mathcal{G}(u)) = 0)| \text{ is negligible} \quad (1p)$$

   for every $u \overset{R}{\leftarrow} \{0,1\}^n$ picked uniformly at random. The statistical test $\mathcal{T}$ outputs $\mathcal{T}(x) = \begin{cases} 1 & \text{if } \mathcal{G}\text{'s output considered not random} \\ 0 & \text{otherwise} \end{cases}$ (1p)

2. Let $(\mathbf{E},\mathbf{D})$ be a correct cipher. Write the formula for correct decryption in the encryption schemes below.
   (Hint: your solution should look like $\mathbf{D}'((k_1, k_2), c) = \mathbf{D}(k_1, c \oplus k_1) \oplus k_2$).

   (a) $\mathbf{E}'((k_1, k_2), m) = \mathbf{E}(k_1, m \oplus k_2)$. **(2p)**

   $\mathbf{D}'((k_1, k_2), c) = \mathbf{D}(k_1, c) \oplus k_2$ (2p)

   (b) $\mathbf{E}'((k_1, k_2), m) = \mathbf{E}(k_1, m) \oplus k_2$. **(2p)**

   $\mathbf{D}'((k_1, k_2), c) = \mathbf{D}(k_1, c \oplus k_2)$ (2p)

(c) $\mathbf{E}'((k_1,k_2),m) = \mathbf{E}(k_2,\mathbf{E}(k_1,m)) \oplus k_1$. **(2p)**

$\mathbf{D}'((k_1,k_2),c) = \mathbf{D}(k_1,\mathbf{D}(k_2,c \oplus k_1))$ (2p)

Public Key Encryption (30p)

3. Describe the RSA encryption scheme. **(7p)**

The RSA encryption scheme is made by the three algorithm (**KeyGen**, **Enc**, **Dec**), defined as follows.

KeyGen$(\lambda) \to (\mathbf{pk},\mathbf{sk})$, is the key generation algorithm. Given λ, the security parameter of the scheme, the algorithm does:
1. Generate two, distinct λ-bit primes p,q; and compute $N = pq$ and $\varphi(N)$, where $\varphi(\cdot)$ denotes Euler Totient function. (1p)
2. Choose a random integer $e \overset{R}{\leftarrow} \mathbb{Z}_{\varphi(N)}$ satisfying GCD$(e,\varphi(N))$=1. Compute $d = e^{-1} \mod \varphi(N)$. (1p)
3. Set $\mathbf{pk} = (N,e)$ and $\mathbf{sk} = d$. (1p) *(note: the role of e and d is interchangeable, as long as* **Enc** *takes as input the public value, and* **Dec** *the secret one).*

Enc$(\mathbf{pk},m) \to c$, is the encryption algorithm. It takes as input the public key e and a message $m \in \mathbb{Z}_{\varphi(N)}$ and outputs the ciphertext $c = m^e \mod N$. (2p)

Dec$(\mathbf{sk},c) \to m$ is the decryption algorithm. It takes as input the secret key d, and the ciphertext c; it outputs the message $m = c^d \mod N$. (2p)

4. This exercise is about the Extended Euclidean Algorithm (and modular inversion).

(a) Use the Extended Euclidean Algorithm to show that 25 and 28 are relatively prime and to find $x,y \in \mathbb{Z}$ such that $25x + 28y = 1$. **(6p)**

By running the Euclidean Algorithm on $a = 28$ and $b = 25$ one obtains the Greatest Common Divisor (GCD) between the two numbers as follows: (2p)

$$\begin{array}{ccrr||ccrr}
28 & = & 25(1) & +\,3 & a & = & b\,(q_1) & +\,r_1 \\
25 & = & 3(8) & +\,1 & b & = & r_1\,(q_2) & +\,r_2 \\
3 & = & 1(3) & +\,0 & r_1 & = & r_2\,(q_3) & +\,r_3
\end{array}$$

The GCD of 28 and 25 is the reminder of the second-last row: $r_2 = 1$. Therefore, the two numbers are relatively prime. (2p)

In order to find x and y such that $25x + 28y = 1$ (the linear combinators of Bézout's identity) we need to use the Extended Euclidean Algorithm. We start from the second-last row in the above table, and read the equations "backwards" in order to obtain $r_2 = 1$ (the GCD) written in terms of a and b (the two given numbers).

$$\begin{array}{ccccccccc}
25 & = & 3(8) & +1 & \Leftrightarrow & 1 & = & 25 - 8(3) \\
28 & = & 25(1) & +3 & \Leftrightarrow & 3 & = & 28 - 25(1)
\end{array}$$

Substituting the 3 in the first (right-hand-side) equation with the second equation we get:

$$1 = 25 - 8(3) = 25 - 8(28 - 25) = 25(9) + 28(-8).$$

Thus, $x = 9$ and $y = -8$. (2p)

(b) Are there many integer solutions x, y to the equation $25x + 28y = 1$? Write them explicitly. **(5p)**

Yes, (1p) there are infinitely many (countable number of) solutions to the given equation. More precisely, for any integer $t \in \mathbb{Z}$, we can set $x_t = (9 + 28t)$ (2p) and $y_t = (-8 - 25t)$, (2p), then all the solutions are described by:

$$25 \cdot (9 + 28t) + 28 \cdot (-8 - 25t) = 1.$$

(c) Compute the inverse of 25 in $\mathbb{Z}_{28}^*$ and the inverse of 28 in $\mathbb{Z}_{25}^*$. **(2p)**

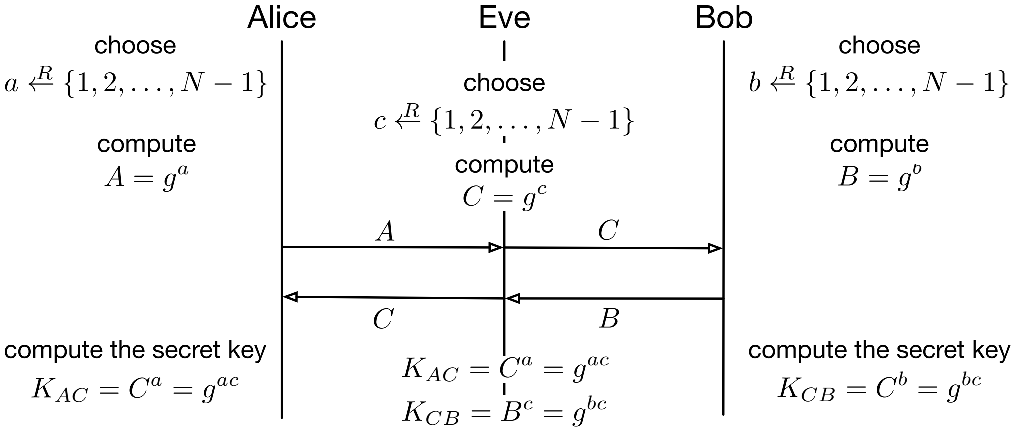
Reading the equation $25 \cdot 9 + 28 \cdot (-8) = 1$ modulus 28 we can see that the inverse of 25 in $\mathbb{Z}_{28}^*$ is 9. (1p) Similarly, reading the previous equation $\mod 25$ we get: $28(-8) = 1 \mod 25$ that is $28^{-1} = -8 = 17 \mod 25$. (1p)

(d) How many elements are in $\mathbb{Z}_{28}^*$? **(2p)**
$|\mathbb{Z}_{28}^*| = \varphi(\mathbf{28}) = \varphi(7) \cdot \varphi(2^2) = (7 - 1) \cdot 2^{2-1}(2 - 1) = \mathbf{12}$. (2p)

(e) *Bonus question: Write down all the elements in* $\mathbb{Z}_{28}^*$. *(3 bonus points)*
$\mathbb{Z}_{28}^*$ is the group generated by all the invertible elements of $\mathbb{Z}_{28}$, i.e., all the positive numbers $a \in \{1, 2, \dots, 28\}$ that are coprime with 28 (since in this case the $\text{GCD}(a, 28) = 1$ and we can compute the inverses as shown in the points above of the exercise). (1p)
Explicitly, we have $\mathbb{Z}_{28}^* = \{1, 3, 5, 9, 11, 13, 15, 17, 19, 23, 25, 27\}$. (2p)

5. Describe the Man-in-the-Middle attack against the Diffie-Hellman key-exchange protocol. **(8p)**

The Man-in-the-Middle attack is an attack where the adversary (called Eve in the picture below[1]) lies in-between Alice and Bob. Eve can see and modify the messages in the conversation between Alice and Bob. (1p)



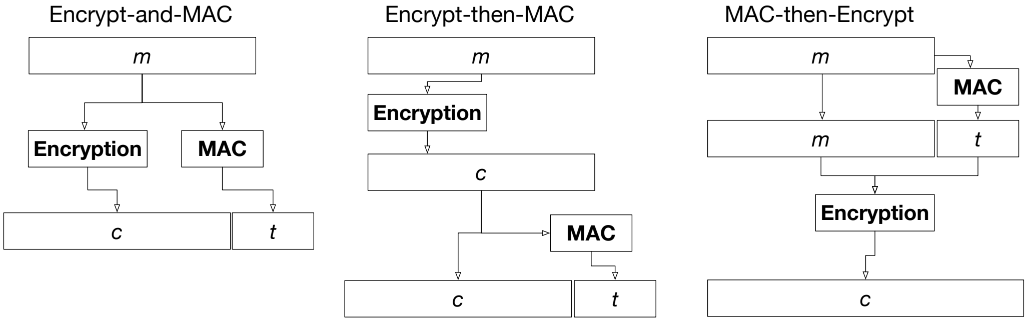
(5p)

At the end of the (tampered) key exchange protocol, Eve managed to create a shared secret key with Alice ($K_{AC} = g^{ac}$), while Alice believes that K_{AC} is her secret key shared with Bob. (1p) At the same time, $K_{CB} = g^{bc}$ is the shared secret key between Eve and Bob, who, however, thinks K_{CB} is the shared secret key between him and Alice. (1p)

[1] In the picture, we assume to be working in a cyclic group $\mathbb{G} =< g >$ of order $N - 1$, for example $\mathbb{Z}_p^*$.

Data Integrity (20p)

6. In order to achieve authenticated encryption, i.e., authenticity (and integrity) of ciphertexts one can combine a MAC (Message Authentication Code) with an Encryption scheme in the following three ways:



(a) Using the functions $\mathbf{Enc}(\mathbf{pk}, \cdot)$ and $\mathbf{MAC}(\mathbf{sk}', \cdot)$, write the output of the three schemes above. **(6p)**
(Hint: your answer is a composition of the above functions. It shall look like: $\mathbf{Enc}\big(\mathbf{pk}, \mathbf{MAC}(\mathbf{sk}', m)\big)$. *You can denote the concatenation of two strings with the symbol* $||$*.)*

Encrypt-and-MAC: $\mathbf{Enc}(\mathbf{pk}, m) \,||\, \mathbf{MAC}(\mathbf{sk}', m)$. (2p)
Encrypt-then-MAC: $\mathbf{Enc}(\mathbf{pk}, m) \,||\, \mathbf{MAC}(\mathbf{sk}', c)$. (2p)
MAC-then-Encrypt: $\mathbf{Enc}\big(\mathbf{pk}, m \,||\, \mathbf{MAC}(\mathbf{sk}', m)\big)$. (2p)

(b) Write an Authenticated-Decryption algorithm for the Encrypt-and-MAC scheme. Note that the output of $\mathbf{ADec}\big((\mathbf{sk}, \mathbf{sk}'), c \,||\, t\big)$ is $\perp$ if we cannot guarantee the integrity of the message, otherwise the output is the correct plaintext. **(4p)**

The Authenticated-Decryption algorithm is defined as follows.
$\mathbf{ADec}\big((\mathbf{sk}, \mathbf{sk}'), c \,||\, t\big)$:
(1) decrypt $\mathbf{Dec}(\mathbf{sk}, c) \rightarrow m$, (1p)
(3) check if $\mathbf{MAC}(\mathbf{sk}', m) == t$. (1p)
If the last check fails, it means that $\mathbf{MAC}(\mathbf{sk}', m) \neq t$, i.e., there is no plaintext integrity and the algorithm should return $\perp$. (1p) Otherwise **ADec** returns $m \leftarrow \mathbf{Dec}(\mathbf{sk}, c)$. (1p)

7. One fundamental property of hash functions is *collision resistance.*

(a) Give the definition of a collision resistant hash function. **(3p)**

A hash function $H : \{0,1\}^N \rightarrow \{0,1\}^n$ (with $N > n$) is said to be collision resistant if it is *hard* to find two inputs that hash to the same output (1p). Formally, we say that it is computationally infeasible to find $x, y \in \{0,1\}^*$ such that $x \neq y$ and $H(x) = H(y)$. (2p)

(b) What security class do collision resistant hash function belong to?
(e.g. unconditional security, computational security...) **(2p)**

Collision resistance is a computational security definition (2p), indeed since the image space of H (the digest) is smaller than its input space, there will always exist distinct strings on which H has the same image. Security relies on the fact that it *computationally infeasible* to *determine* such strings.

(c) Prove the birthday paradox: for large enough N (where N is the size of the output of the hash function) after $k^2 = 2N\log(2)$ trials the probability of finding a collision is at least $1/2$. **(5p)**
(Hints: start by computing the probability of the complementary event, i.e., that after k trials one has found no collision. Towards the end, you can do the substitution $k^2 \sim k(k-1)$, to simplify computations. Additional useful bounds and formulas: $\log(1-x) < -x$, $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$*.)*

We want to show that $P = \mathsf{Prob}(\text{ finding a collision in } k \text{ trials }) > \frac{1}{2}$ for *not so large* values of k. We being by computing the probability of the complementary event, i.e., $1-P = \mathsf{Prob}(\text{ no collision in } k \text{ trials })$, that is the probability that at every trial we get distinct messages, thus:

$$\begin{aligned} 1-P &= \mathsf{Prob}(\text{ no collision in } k \text{ trials }) \\ &= \left(\tfrac{N-1}{N}\right)\cdot\left(\tfrac{N-2}{N}\right)\cdots\left(\tfrac{N-(k-1)}{N}\right) \end{aligned}$$

By taking the logarithm at both sides of the equation we get:

$$\begin{aligned} \log(1-P) &= \log\left(\tfrac{N-1}{N}\right) + \log\left(\tfrac{N-2}{N}\right) + \cdots + \log\left(\tfrac{N-(k-1)}{N}\right) \\ &= \log\left(1-\tfrac{1}{N}\right) + \log\left(1-\tfrac{2}{N}\right) + \cdots + \log\left(1-\tfrac{k-1}{N}\right) \\ &< -\tfrac{1}{N} - \tfrac{2}{N} + \cdots - \tfrac{k-1}{N} \quad \text{from the bound given in the hint} \\ &= -\textstyle\sum_{i=1}^{k-1} \tfrac{i}{N} \\ &\leq -\tfrac{1}{N}\left(\tfrac{k(k-1)}{2}\right) \quad \text{from the Gaussian sum given in the hint} \\ &\leq -\tfrac{1}{N}\tfrac{k^2}{2} \quad \text{from the subsitution given in the hint} \end{aligned}$$

Now if we set $k^2 = 2N\log(2)$, as given in the question, we get:

$$\begin{aligned} \log(1-P) &\leq -\tfrac{1}{N}\left(\tfrac{2N\log(2)}{2}\right) \\ &\leq -\log(2) = \log(2^{-1}) = \log(\tfrac{1}{2})\,. \end{aligned}$$

Taking the exponential on both sides of the inequality we get: $1-P \leq \frac{1}{2}$, thus $P \geq \frac{1}{2}$. QED. (5p)

Advanced Topics in Cryptography (30p)

8. Describe in your own words (or give the definition of)

(a) The structure of a Σ (sigma) protocol. **(6p)**

A Σ-protocol has (usually) the following structure:

i. The prover generates a random looking value called *commitment* (or witness) and sends it to the verifier. (2p)

ii. The verifier replies with a (random) challenge to the prover. (2p)

iii. The prover performs some computations, based on the challenge and the secret (connected to the statement). The resulting value is sent as a response to the verifier. (2p)

(b) The three main properties of a Secret Sharing Scheme, with threshold t. **(6p)**

i. $(t+1)$**-correctness:** (1p) any $t+1$ parties together can compute the secret s. (1p)

ii. **Privacy:** (1p) no single party alone learns anything about the secret s. (1p)

iii. **t-unconditional security:** (1p) any subset of t parties cannot recover the secret s, no matter how much computational power the parties have. (1p)

9. Consider the Mignotte's Secret Sharing Scheme with the values $m_1 = 5, m_2 = 6, m_3 = 7, m_4 = 11, m_5 = 13$.

(a) Imagine you are the Dealer and you want to share the value $s = 150$. Explain how you would compute the shares for each of the 5 parties, and compute the values s_i, for $i = 1, 2, 3, 4, 5$. **(6p)**

In order to compute the shares using Mignotte's Secret Sharing Scheme, the Dealer has to compute: $s_i = s \mod m_i$ for all the parties $i \in \{1, 2, 3, 4, 5\}$. (1p) Explicitly, we get:

$$\begin{aligned} s_1 &= 150 \mod 5 = 0, \text{(1p)} \\ s_2 &= 150 \mod 6 = 0, \text{(1p)} \\ s_3 &= 150 \mod 7 = 3, \text{(1p)} \\ s_4 &= 150 \mod 11 = 7, \text{(1p)} \\ s_5 &= 150 \mod 13 = 7. \text{(1p)} \end{aligned}$$

(b) Now assume you hold the three shares $s_1 = 0$, $s_2 = 4$, $s_5 = 5$. You know that the shares have been generated using Mignotte's Secret Sharing Scheme with $m_1 = 5, m_2 = 6, m_3 = 7, m_4 = 11, m_5 = 13$. Can you reconstruct the secret value s? If so, compute it, otherwise explain why not. **(12p)**

Yes, it is possible to reconstruct s from the information given. (1p) The secret s is the solution (unique modulus $m_1 \cdot m_2 \cdot m_5 = 5*6*13 = 390$) of the following linear system of congruences:

$$\begin{cases} x = 0 \mod 5 \\ x = 4 \mod 6 \\ x = 5 \mod 13 \end{cases} \qquad \text{(2p)}$$

I proceed by finding a solution to the first two equations in the above system:

$$\begin{cases} x = 0 \mod 5 \\ x = 4 \mod 6 \end{cases} \qquad \begin{array}{c} \text{Bézout Identity} \\ 5(5) + 6(-4) = 1 \quad \text{(2p)} \\ \text{Found using the EEA} \end{array}$$

The partial solution is $x = 4 \cdot (25) + 0 \cdot (-24) = 100 \equiv 10 \mod 30$. (2p) Now, I combine this result with the last equation of the linear system:

$$\begin{cases} x = 10 \mod 30 \\ x = 5 \mod 13 \end{cases} \qquad \begin{array}{c} \text{Bézout Identity} \\ 13(7) + 30(-3) = 1 \quad \text{(2p)} \\ \text{Found using the EEA} \end{array}$$

The final result is $x = 10 \cdot 13 \cdot 7 + 5 \cdot 30 \cdot (-3) = 460 \equiv 70 \mod 390$. (3p) We observe that since $70 < 143 = m_4 * m_5$, some combinations of two shares would enable to compute s without the need of a third one. In our case, applying the CRT on s_2, s_5 solely, we get $s = 70$.

(c) *Bonus question: For what values of the threshold t, the given m_i can be used? (2 bonus points)*

By construction it must hold that $1 \leq t \leq n-1 = 4$, where $n = 5$ is the number of parties involved in the scheme. We have to check for what values of $t \in \{1, \cdots, 4\}$, both the following conditions hold:

$$\begin{array}{ll} (1) & \gcd(m_i, m_j) = 1 \text{ for all } i, j \in \{1, \cdots, 5\} \text{ with } i \neq j \\ (2) & m_{5-t+1} \cdots m_t < m_1 \cdots m_{t+1} \end{array}$$

We can easily see that (1) is satisfied for all the possible choices of $i, j \in \{1, \cdots, 5\}$ with $i \neq j$. For (2) we need to check for the different values for t:

- $t = 1$: $13 = m_5 < m_1 m_2 = 30$, so this value of t can be used.
- $t = 2$: $143 = m_4 m_5 < m_1 m_2 m_3 = 210$, so this value of t can be used.
- $t = 3$: $1001 < 2310$, so this value of t can be used.
- $t = 4$: $6006 < 30030$, so this value of t can be used.