Mastermath
Spring 2015
Exam Cryptology Course
Tuesday, 09 June 2015

Name :

Student number :

Exercise	1	2	3	4	5	6	total
points							

Notes: Please hand in this sheet at the end of the exam. You may keep the sheets with the exercises.
This exam consists of 6 exercises. You have from 14:00 – 17:00 to solve them. You can reach 100 points.
Make sure to justify your answers in detail and to give clear arguments. Document all steps, in particular of algorithms; it is not sufficient to state the correct result without the explanation. If the problem requires usage of a particular algorithm other solutions will not be accepted even if they give the correct result.
All answers must be submitted on paper provided by the university; should you require more sheets ask the proctor. State your name on every sheet.
Do not write in red or with a pencil.
You are allowed to use any books and notes, e.g. your homework. You are not allowed to use the textbooks of your colleagues.
You are allowed to use a calculator without networking abilities. Usage of laptops and cell phones is forbidden.

1. This exercise is about code-based cryptography.

   (a) The binary Hamming code $\mathcal{H}_4(2)$ has parity check matrix

$$H = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

   and parameters $[15, 11, 3]$.
   Correct the word $(0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1)$. 4 points

2. This exercise is about attacks on code-based cryptography.

   Lee and Brickell's algorithm finds low-weight codewords. Assume for concreteness that the code contains a word of weight t and assume for simplicity that there is only one word c of weight t.

   The outer loop randomly permutes the columns of the parity-check matrix H and turns the rightmost $n-k$ columns into an $(n-k)\times(n-k)$ identity matrix (if these columns are not linearly independent another permutation is tried).

   The inner loop picks p of the remaining k columns and computes the sum of these p columns, resulting in a column vector of length $n-k$. The algorithm succeeds if the resulting vector has weight $t-p$.

   (a) Explain how to obtain the word c of weight t from the steps described above, i.e., assume that you have found p columns so that their sum has weight $t-p$. 4 points

   (b) Compute the probability that one choice of column permutation distributes the positions of c in such a way that p of the ones land in the k positions on the left and $t-p$ of them land in the $n-k$ positions on the right. Ignore the likelihood of obtaining an identity matrix on the right. 8 points

   (c) Compute the probability of success of one round of the inner loop, i.e., the probability of picking the correct p columns to get the weight $t-p$ vector, given that the outer loop has permuted the columns to end up with a split suitable to find c this way. 4 points

3. This exercise is about the NTRU encryption system. Remember that all computations take place in $R = \mathbb{Z}[x]/(x^N - 1)$ and are done modulo p or modulo q. The secret key is $f(x) \in R$, $f \cdot f_p = 1$ in R/p, $f \cdot f_q = 1$ in R/q, $h = f_q \cdot g$ in R/q, and $c = p\phi h + m$ in R/q for random ϕ and message m.

(a) Let $p = 2, d_f = d_\phi = d_g = 2$ and $N = 13$. Compute how large q has to be so that decryption is guaranteed to be correct, i.e., so that taking the coefficients of $a = f \cdot c$ in R/q as elements in $[-(q-1)/2, (q-1)/2]$ produces the correct message. $\boxed{\text{8 points}}$

(b) Let $N = 5, p = 2$, and $f(x) = x^4 - x + 1$. Compute the inverse f_p of f in R/p and compute $f \cdot f_p$ in R/p to verify that the result is indeed 1.
Hint: this needs a XGCD computation. Make sure to document the steps or state how you did this computation. Do *not* simply state the result or just a verification of the result. $\boxed{\text{6 points}}$

4. This exercise is about differential cryptanalysis of the same toy cipher from the lectures. Using key $(k_1, k_2, k_3, k_4, k_5) \in (\{0,1\}^{16})^5$ it encrypts a plaintext $P = P_1 || \ldots || P_{16} \in \{0,1\}^{16}$ as follows. Let S be the current state, we start with $S = P$. Rounds $i = 1, 2, 3$ perform key mixing

$$S \leftarrow S \oplus k_i,$$

substitution using a Sbox (Table 2)

$$S \leftarrow Sbox(S_1 \ldots S_4) || \ldots || Sbox(S_{12} \ldots S_{16}),$$

and finally applies permutation π_P (Table 1) on the state bits:

$$S \leftarrow S_{\pi_P(1)} || \ldots || S_{\pi_P(16)} = S_1 || S_5 || S_9 || \ldots || S_{12} || S_{16}.$$

Round 4 applies key mixing with round key k_4, substitution using the sbox and finally applies another key mixing with round key k_5. After round 4, the cipher outputs the current state S as the ciphertext C.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\pi_P(i)$	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

Table 1: State bit permutation

In contrast to the lecture notes, we use the following SBox:

in	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
out	0	3	5	8	6	9	C	7	D	A	E	4	1	F	B	2

Note most significant bit is left most bit and using hexadecimal notation.
So 'C' represents number 12 or '1100' in binary.

Table 2: Sbox

This SBox has the following Difference Distribution Table (Table 3:

		out															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	0																
	1																
	2																
	3			0	2	4	2	2	0	2	2	0	0	0	0	0	0
	4			0	0	0	4	4	0	0	2	2	0	2	0	0	2
	5			4	0	2	2	0	0	0	2	0	2	2	0	0	2
	6			0	2	2	0	2	0	2	0	0	2	2	0	0	2
	7			0	0	0	2	0	2	0	0	0	0	2	0	2	4
in	8			0	0	0	2	2	4	0	2	0	2	2	2	0	0
	9			0	0	0	0	2	0	2	2	2	0	2	0	4	0
	A			2	0	0	0	0	2	4	0	0	2	0	4	2	0
	B			2	2	4	2	2	0	0	0	0	0	0	2	0	2
	C			2	4	0	0	0	0	0	0	2	2	2	0	2	0
	D			2	2	2	0	0	2	2	2	0	0	2	0	0	2
	E			0	0	2	0	0	2	2	0	0	2	0	4	0	0
	F			4	0	0	0	2	2	2	2	4	0	0	0	0	0

Table 3: Sbox difference distribution table

(a) Complete the DDT. You only have to write down the missing numbers in a table. $\boxed{\text{4 points}}$

(b) Consider the boomerang with input plaintext difference

$$\Delta P = (0000\ 1111\ 0000\ 0000)$$

and output ciphertext difference

$$\Delta C = (0000\ 1110\ 0000\ 0000),$$

then a quartet $(P^{(1)}, P^{(2)}, P^{(3)}, P^{(4)})$ satisfies this boomerang if

$$P^{(1)} \oplus P^{(2)} = \Delta P, \quad P^{(3)} \oplus P^{(4)} = \Delta P, \quad \text{and}$$

$$C^{(1)} \oplus C^{(3)} = \Delta P, \quad C^{(2)} \oplus C^{(4)} = \Delta C.$$

Compute the total success probability of finding such quartets over all round 1 & 2 differentials with the given ΔP and all round 3 & 4 differentials with the given ΔC. (Hint: in round 2 each Sbox has either input difference 0 or 4 (0100), so every active round 2 Sbox contributes a term $2 \times (4/16)^2 + 4 \times (2/16)^2$. Likewise, in round 3 each active Sbox has output difference 4.) $\boxed{\text{8 points}}$

(c) Consider all 3-round differentials that have only 1 active Sbox in round 1 and only 1 active Sbox in round 3. Prove that all such 3-round differentials are impossible differentials. 8 points

5. This exercise is about hash-based signatures.

The HORS (Hash to Obtain Random Subset) signature scheme is an example of a few-time signature scheme. It has integer parameters k, t, and ℓ, uses a hash function $H : \{0,1\}^* \to \{0,1\}^{k \cdot \log_2 t}$ and a one-way function $f : \{0,1\}^\ell \to \{0,1\}^\ell$. For simplicity assume that H is surjective.

To generate the key pair a user picks t strings $s_i \in \{0,1\}^\ell$ and computes $v_i = f(s_i)$ for $0 \leq i < t$. The public key is $P = (v_0, v_1, \ldots, v_{t-1})$; the secret key is $S = (s_0, s_1, \ldots, s_{t-1})$.

To sign a message $m \in \{0,1\}^*$ compute $H(m) = (h_0, h_1, \ldots, h_{k-1})$, where each $h_i \in \{0, 1, 2, \ldots, t-1\}$. The signature on m is $\sigma = (s_{h_0}, s_{h_1}, s_{h_2}, \ldots, s_{h_{k-1}})$.

To verify the signature, compute $H(m) = (h_0, h_1, \ldots, h_{k-1})$ and $(f(s_{h_0}), f(s_{h_1}), f(s_{h_2}), \ldots, f(s_{h_{k-1}}))$ and verify that $f(s_{h_i}) = v_{h_i}$ for $0 \leq i < t$.

(a) Let $\ell = 80$, $t = 2^5$, and $k = 3$. How large (in bits) are the public and secret keys? How large is a signature? How many different signatures can the signer generate for a fixed key pair as $H(m)$ varies? Ignore that s-values could collide. 5 points

(b) The same public key can be used for $r + 1$ signatures if H is r-subset-resilient, meaning that given r signatures and thus r vectors $\sigma_j = (s_{h_{j,0}}, s_{h_{j,1}}, s_{h_{j,2}}, \ldots, s_{h_{j,k-1}})$, $1 \leq j \leq r$ the probability that $H(m')$ consists entirely of components in $\{h_{j,i} | 0 \leq i < k, 1 \leq j \leq r\}$ is negligible.

Even for $r = 1$, i.e. after seeing just one typical signature, an attacker has an advantage at creating a fake signature. What are the options beyond exact collisions in H? 2 points

(c) Let $\ell = 80$, $t = 2^5$, and $k = 3$. Let m be a message so that $H(m) = (h_0, h_1, h_2)$ satisfies that $h_i \neq h_j$ for $i \neq j$. You get to specify messages that Alice signs. You may not ask Alice to sign m.

State the smallest number of HORS signatures you need to request from Alice in order to construct a signature on m? How many calls to H does this require on average? You should assume that H and f do not have additional weaknesses beyond having too small parameters. Explain how you could use under 1000 evaluations of H if you are allowed to ask for two signatures. 9 points

6. This exercise is about the cryptanalysis of the broken cryptographic hash function MD5. In brief, MD5 uses a compression function `Compress` that takes as input a chaining value $CV_{in} = (A, B, C, D) \in (\mathbb{Z}/2^{32}\mathbb{Z})^4$ and a message block

$M = (m_0, \dots, m_{15}) \in (\mathbb{Z}/2^3 2\mathbb{Z})^{16}$. It initializes $(Q_0, Q_{-1}, Q_{-2}, Q_{-3}) = (B, C, D, A)$ and computes 64 steps $i = 0, \dots, 63$:

$$F_i = BF_i(Q_i, Q_{i-1}, Q_{i-2}); \quad T_i = Q_{i-3} + F_i + AC_i + W_i;$$

$$R_i = RL(T_i, RC_t); \quad Q_{i+1} = Q_i + R_i,$$

where BF_i is a boolean function, AC_i is an addition constant, W_i is message word $m_{\pi(i)}$ and $RL(\cdot, n)$ is bitwise cyclic left rotation by n bit positions (see Table 4). It outputs an update chaining value CV_{out}:

$$CV_{out} = CV_{in} + (Q_{61}, Q_{64}, Q_{63}, Q_{62}).$$

i	0	1	2	3	4	5	6	7
RC_{32+i}	4	11	16	23	4	11	16	23
W_{32+i}	m_5	m_8	m_{11}	m_{14}	m_1	m_4	m_7	m_{10}
RC_{40+i}	4	11	16	23	4	11	16	23
W_{40+i}	m_{13}	m_0	m_3	m_6	m_9	m_{12}	m_{15}	m_2
RC_{48+i}	6	10	15	21	6	10	15	21
W_{48+i}	m_0	m_7	m_{14}	m_5	m_{12}	m_3	m_{10}	m_1
RC_{56+i}	6	10	15	21	6	10	15	21
W_{56+i}	m_8	m_{15}	m_6	m_{13}	m_4	m_{11}	m_2	m_9

$$BF_{32}(x, y, z) = \cdots = BF_{47}(x, y, z) = x \oplus y \oplus z$$

$$BF_{48}(x, y, z) = \cdots = BF_{63}(x, y, z) = y \oplus (x \vee \overline{z})$$

Table 4: MD5 Round 3 & 4 boolean functions, rotation constants and message word permutations.

(a) Fill in the missing values in the following partial Sufficient Condition Tables for the boolean functions of round 3 & 4:

BF_{32-47} XYZ	$g = \{0\}$ $c_{BF_{32},XYZ,g}$	$g = \{+1\}$ $c_{BF_{32},XYZ,g}$	$g = \{-1\}$ $c_{BF_{32},XYZ,g}$
`...`	`...`	n/a	n/a
`..+`	n/a	`^.+`	`!.+`
`..-`			

BF_{48-63} XYZ	$g = \{0\}$ $c_{BF_{48},XYZ,g}$	$g = \{+1\}$ $c_{BF_{48},XYZ,g}$	$g = \{-1\}$ $c_{BF_{48},XYZ,g}$
`...`	`...`	n/a	n/a
`-..`	`-.0`	`-11`	`-01`
`+..`			
`++.`			

4 points

(b) Determine a partial differential path for MD5 over steps 48 up to 63 using $\delta m_{11} = 2^{11}$ (and $\delta m_i = 0$ for $i \neq 11$) such that $\delta Q_{45} = \dots \delta Q_{61} = 0$ and $\delta Q_{62} = \delta Q_{63} = \delta Q_{64} = 2^{21}$. Specify ΔQ_i for $i = 45, \dots, 64$ and for non-trivial steps $i = 61, 62, 63$ specify $\Delta F_i, \delta T_i$ and δR_i. 4 points

(c) Determine a partial differential path for MD5 over steps 32 up to 47 using $\delta m_{11} = 2^{11}$ (and $\delta m_i = 0$ for $i \neq 11$) such that $\delta Q_{32} = \dots \delta Q_{48} = 0$. Specify ΔQ_i for $i = 29, \dots, 49$ and for non-trivial steps $i = 32, 33, 34$ specify $\Delta F_i, \delta T_i$ and δR_i. 4 points

(d) As treated in the lecture notes, it is possible given any CV_{in}, CV'_{in} to compute a full differential path over steps $0, \dots, 63$ that completes above found partial differential path over steps $32, \dots, 63$. Finding a solution (M, M') for that full differential path results in

$$CV_{out} = \texttt{Compress}(CV_{in}, M), \quad CV'_{out} = \texttt{Compress}(CV'_{in}, M'),$$

with

$$\delta CV_{out} = \delta CV_{in} + (0, 2^{21}, 2^{21}, 2^{21}).$$

This in fact works for any $\delta m_{11} = 2^b$ with $b = 0, \dots, 31$ and $\delta Q_{62} = \delta Q_{63} = \delta Q_{63} = RL(2^b, 10)$. Prove that given any CV_i, CV'_i with $\delta CV_i = (0, x, x, x)$ for some $x \in \mathbb{Z}/2^{32}\mathbb{Z}$, one can use a series of r of these near-collision attacks to obtain $\delta CV_{i+r} = (0, 0, 0, 0)$ with $r \leq 32$. 6 points

(e) To reduce the amount of near-collision attacks required, one can also consider the negated versions with $\delta m_{11} = -2^b$. As thereby one can add $\pm(0, 2^a, 2^a, 2^a)$ for any $a = 0, \dots, 31$, one can use a binary signed digit representation of x.

Describe a procedure that given any x computes a series of r tuples $(\delta m_{11}, \delta Q_{61} = \delta Q_{62} = \delta Q_{63})$ that one can use to construct r near-collision attacks to reduce $\delta CV_i = (0, x, x, x)$ to zero, where r is minimal. That is, there exists no shorter series that also reduces δCV_i to zero. 4 points

(f) Write down an efficient algorithm based on the birthday paradox that given any CV_{i-1}, CV'_{i-1} computes blocks M_i, M'_i such that

$$\texttt{Compress}(CV_{i-1}, M_i) - \texttt{Compress}(CV'_{i-1}, M'_i) = (0, x, x, x),$$

for some $x \in \mathbb{Z}/2^{32}\mathbb{Z}$ and estimate its complexity. (Hint: rewrite the condition $\delta CV_i = (A, B, C, D) = (0, x, x, x)$ as $\delta A = 0$, $\delta(B - C) = 0$ and $\delta(B - D) = 0$). 8 points