Cryptography and Security — Final Exam

Serge Vaudenay

16.1.2023

- duration: 3h
- no documents allowed, except one 2-sided sheet of handwritten notes
- a pocket calculator is allowed
- communication devices are not allowed
- the exam invigilators will **not** answer any technical question during the exam
- readability and style of writing will be part of the grade
- answers should not be written with a pencil

1 Symmetric-Key RSA

As people scare about quantum computers being able to factor RSA moduli, some people proposed to continue to use RSA by keeping the public key secret and end up with an encryption scheme with a symmetric key $K = (N, e, d)$. For simplicity, we consider plain RSA only. The purpose of the exercise is to make a quantum key recovery attack with chosen plaintext.

Q.1 Fully describe the symmetric-key RSA scheme between a sender Alice and a receiver Bob (key structure, message domains, key generation, encryption, decryption).

Q.2 Describe the game for key recovery with chosen plaintext with the symmetric scheme of the previous question.

Q.3 In the case where e is small (e.g. $e = 65\,535$) and known by the adversary, propose an efficient (quantum) key-recovery attack with one known plaintext.

Q.4 Propose an efficient (quantum) key-recovery attack with two chosen plaintexts, (e is not known any more and can be large as well).

2 Hash-Based Signature

We consider a one-way hash function F from a set E to itself. We further consider a collision-resistant hash function H mapping an arbitrary message m to a digest $H(m)$ belonging to a given hash space. We analyze some digital signature schemes based on F and H.

Q.1 We recall the Lamport scheme with parameter n.

- Key generation: pick $2n$ random $\mathsf{sk}_{i,b} \in E$ for $i = 1, \ldots, n$ and $b \in \{0, 1\}$, compute $\mathsf{pk}_{i,b} = F(\mathsf{sk}_{i,b})$. We set $\mathsf{sk} = (\mathsf{sk}_{i,b})_{i=1,\ldots,n;b=0,1}$ and $\mathsf{pk} = (\mathsf{pk}_{i,b})_{i=1,\ldots,n;b=0,1}$.
- Hash space: $H(m) \in \{0, 1\}^n$.
- Signature: $\sigma = (\mathsf{sk}_{i,H(m)_i})_{i=1,\ldots,n}$.
- Verification: check $F(\sigma_i) = \mathsf{pk}_{i,H(m)_i}$ for $i = 1, \ldots, n$.

The scheme should be used to sign a single message but we investigate what happens if we sign several.

Q.1a Assume the adversary knows two signed messages (m_1, σ_1) and (m_2, σ_2) such that $H(m_1)$ and $H(m_2)$ differ on exactly d bit positions. Given a random m, what is the probability that the adversary can forge a signature for m?

Q.1b If m, m_1, m_2 are random, what are the expected value of d and the probability to forge?

Q.1c Propose a key-recovery chosen message attack using $\mathcal{O}(\log n)$ chosen messages, similar complexity, and success probability 1.

Q.2 We recall the FORS scheme with parameters k and t.

- Key generation: pick kt random $\mathsf{sk}_{i,j} \in E$ for $i = 1, \ldots, k$ and $j = 1, \ldots, t$, compute $\mathsf{pk}_{i,j} = F(\mathsf{sk}_{i,j})$. We set $\mathsf{sk} = (\mathsf{sk}_{i,j})_{i=1,\ldots,k;j=1,\ldots,t}$ and $\mathsf{pk} = (\mathsf{pk}_{i,j})_{i=1,\ldots,k;j=1,\ldots,t}$.
- Hash space: $H(m) \in \{1, \ldots, t\}^k$.
- Signature: set $\sigma = (\mathsf{sk}_{i,H(m)_i})_{i=1,\ldots,k}$.
- Verification: check $F(\sigma_i) = \mathsf{pk}_{i,H(m)_i}$ for $i = 1, \ldots, k$.

This scheme is meant to be used to sign a few messages.

Q.2a After the signature of n random messages, what is, for each i, the expected number of indices j for which $\mathsf{sk}_{i,j}$ is revealed?

Q.2b Compute roughly the probability to be able to forge the signature of a random message after n random messages have been signed.

Q.2c Application: $k = 33$, $t = 2^6$. How many random messages can we sign without this probability becoming larger than $\frac{1}{2}$?

3 DLP in GGM

We define the Discrete Logarithm Problem (DLP) in the Generic Group Model (GGM) as follows. Given a prime number q, we define the following game Γ:

```
$\Gamma^{\mathcal{A}}$
1: pick $x \in \mathbf{Z}_q$ uniformly at random
2:  set $\mathsf{Mem} \leftarrow (1, x)$
3: run $\mathcal{A}^{\mathsf{OAdd},\mathsf{OCmp}}(q) \rightarrow y$
4: return $1_{x=y}$
```

```
Oracle $\mathsf{OAdd}(i, j)$:
5: $S \leftarrow \mathsf{Mem}[i] + \mathsf{Mem}[j] \bmod q$
6:  append $S$ to the list $\mathsf{Mem}$
7: return

Oracle $\mathsf{OCmp}(i)$:
8: return $1_{\mathsf{Mem}[i]=0}$
```

We use a memory Mem which is defined as a list of write-only registers. If Mem is of length n, the registers are $\mathsf{Mem}[1], \ldots, \mathsf{Mem}[n]$. In Step 2, Mem is initialized with length $n = 2$ so that $\mathsf{Mem}[1] = 1$ and $\mathsf{Mem}[2] = x$. In Step 6, the length of Mem is incremented: Appending a value S to Mem means defining a new register $\mathsf{Mem}[n+1]$ set to S. Essentially, this model allows the adversary $\mathcal{A}$ to do group operations over $\mathbf{Z}_q$ in a blind manner through the OAdd oracle. The adversary does not see the content of the memory Mem but knows the group order q. Additionally, the adversary can test is a register contains 0 through the OCmp oracle. The DLP is then defined in the usual manner. The advantage of $\mathcal{A}$ is

$$\mathsf{Adv}_{\mathcal{A}} = \Pr[\Gamma^{\mathcal{A}} \rightarrow 1]$$

The goal of the exercise is to show that DDH is hard in GGM.

Q.1 Let $a, b \in \mathbf{Z}_q$ be fixed. Construct an (as efficient as possible) adversary $\mathcal{A}$ so that at the end of the game, the last memory register contains $a + bx \bmod q$. Analyze its complexity.

Q.2 By using only OAdd and OCmp and given two integers i and j, show how $\mathcal{A}$ can efficiently determine whether $\mathsf{Mem}[i] = \mathsf{Mem}[j]$ or not. Analyze its complexity.

Q.3 Propose an adversary $\mathcal{A}$ of advantage 1 of minimal complexity to solve DLP. Carefully discuss if it fits the GGM model.

Q.4 If $\mathcal{A}$ never queries OCmp, prove that $\mathsf{Adv}_{\mathcal{A}}(\lambda) = \frac{1}{q}$.

Q.5 Prove by induction that a process which observes the queries made by $\mathcal{A}$ to the oracle can define 2-dimensional vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ such that for every i, we have $\mathsf{Mem}[i] = \boldsymbol{v}_i[1] + \boldsymbol{v}_i[2] \times x \bmod q$.

Q.6 Given $(a, b) \in \mathbf{Z}_q^2$ such that $(a, b) \neq (0, 0)$, prove that $\Pr[a + bx \bmod q = 0] \leq \frac{1}{q}$ over the random selection of $x \in \mathbf{Z}_q$.

Q.7 We define the alternate comparison procedure which is formalized as a subroutine of the adversary:

Subroutine $\mathsf{AltCmp}(i)$:

```
return $1_{v_i=(0,0)}$
```

where v_i is obtained from Q.5. We define $\mathcal{A}_t$, the adversary running exactly as $\mathcal{A}$ except that the t first queries to OCmp are made to AltCmp instead of Ocmp.

Prove that $\mathsf{Adv}_{\mathcal{A}_t} \leq \mathsf{Adv}_{\mathcal{A}_{t-1}} + \frac{1}{q}$.

HINT: define the event E that $\boldsymbol{v}_i \neq 0$ and $\mathsf{Mem}[i] = 0$, where i is the index which is queried to the t-th comparison oracle call.

Q.8 Deduce $\mathsf{Adv}_{\mathcal{A}}(\lambda) \leq \frac{n+1}{q}$ when $\mathcal{A}$ is limited to n oracle calls to OCmp.