Advanced Cryptography — Final Exam
Solution

Serge Vaudenay

26.6.2018

- duration: 3h
- any document allowed
- 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

The exam grade follows a linear scale in which each question has the same weight.

1 Ciphertext Collision in Semantically Secure Cryptosystems

We consider a public-key cryptosystem $(\mathsf{Gen}, \mathcal{M}, \mathsf{Enc}, \mathsf{Dec})$. We assume perfect correctness, i.e. for all s and all $x \in \mathcal{M}$, if $(K_p, K_s) \leftarrow \mathsf{Gen}(1^s)$ then

$$\Pr[\mathsf{Dec}_{K_s}(\mathsf{Enc}_{K_p}(x)) = x] = 1$$

Given a probabilistic polynomial-time adversary $\mathcal{A}$, we consider the following game:

Game $\Gamma_{\mathcal{A}}(s)$:
$(K_p, K_s) \leftarrow \mathsf{Gen}(1^s)$
$X \leftarrow \mathcal{A}(K_p)$
$Y_0 \leftarrow \mathsf{Enc}_{K_p}(X)$
$Y_1 \leftarrow \mathsf{Enc}_{K_p}(X)$
return $1_{Y_0=Y_1}$

Q.1 Prove that if the cryptosystem is IND-CPA secure, then $\Pr[\Gamma_{\mathcal{A}}(s) \rightarrow 1]$ is negligible.
Hint: construct an IND-CPA adversary with advantage related to $\Pr[\Gamma_{\mathcal{A}}(s) \rightarrow 1]$.

We define an IND-CPA *adversary as follows:*
Algorithm $\mathcal{B}(s)$*:*
receive K_p
run $X_1 \leftarrow \mathcal{A}(K_p)$
pick $X_0 \in \mathcal{M}$ *such that* $X_0 \neq X_1$
send X_0, X_1*, receive* Y
$Y' \leftarrow \mathsf{Enc}_{K_p}(X_1)$
return $1_{Y=Y'}$

If Y *is the encryption of* X_1*, the* IND-CPA *game outputs* 1 *with probability* $\Pr[\Gamma_{\mathcal{A}}(s) \rightarrow 1]$*. If* Y *is the encryption of* X_0*, we cannot have* $Y = Y'$*, so the game outputs* 1 *with probability zero. Hence, the advantage of* $\mathcal{B}$ *is exactly* $\Pr[\Gamma_{\mathcal{A}}(s) \rightarrow 1]$*. Due to* IND-CPA *security, this is negligible.*

2 Non-Malleability in Adaptive Security

This exercise is inspired from Bellare-Desai-Pointcheval-Rogaway, Relations Among Notions of Security for Public-Key Encryption Schemes, *CRYPTO 1998, LNCS vol. 1462, Springer.*

We consider a public-key cryptosystem $(\mathsf{Gen}, \mathcal{M}, \mathsf{Enc}, \mathsf{Dec})$. We assume perfect correctness, i.e. for all s and all $x \in \mathcal{M}$, if $(K_p, K_s) \leftarrow \mathsf{Gen}(1^s)$ then

$$\Pr[\mathsf{Dec}_{K_s}(\mathsf{Enc}_{K_p}(x)) = x] = 1$$

Given an adversary in two parts $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, a bit $b \in \{0, 1\}$, and the security parameter s, we define the $\mathsf{IND\text{-}CCA}$ game as follows:

Game $\mathsf{IND\text{-}CCA}^b_{\mathcal{A}}(s)$
$(K_p, K_s) \leftarrow \mathsf{Gen}(1^s)$
$(X_0, X_1, \sigma) \leftarrow \mathcal{A}_1^{\mathcal{O}_1(\cdot)}(K_p)$ $\quad \triangleright$ σ is a "state" for $\mathcal{A}_1$ to transmit data to $\mathcal{A}_2$
$Y \leftarrow \mathsf{Enc}_{K_p}(X_b)$
$b' \leftarrow \mathcal{A}_2^{\mathcal{O}_2(\cdot)}(\sigma, Y)$
return b'

where the oracles $\mathcal{O}_1$ and $\mathcal{O}_2$ are defined as follows:

Oracle $\mathcal{O}_1(y)$:
return $\mathsf{Dec}_{K_s}(y)$
Oracle $\mathcal{O}_2(y)$:
if $y = Y$ **then**
abort the game
end if
return $\mathsf{Dec}_{K_s}(y)$

We define the advantage

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{A}}(s) = \Pr[\mathsf{IND\text{-}CCA}^1_{\mathcal{A}}(s) \to 1] - \Pr[\mathsf{IND\text{-}CCA}^0_{\mathcal{A}}(s) \to 1]$$

We say that the cryptosystem is $\mathsf{IND\text{-}CCA}$ secure if for all probabilistic polynomial time (PPT) adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{A}}(s)$ is negligible.

Q.1 The definition of $\mathsf{IND\text{-}CCA}$ security which was given in the course (Def.5.5 on p.55–56 in the lecture notes, or slide p.404) was based on an interactive game between an adversary and a challenger. Prove that the two styles of definition for $\mathsf{IND\text{-}CCA}$ security are equivalent. (Carefully construct $(\mathcal{A}_1, \mathcal{A}_2)$ from an interactive adversary and an interactive adversary from $(\mathcal{A}_1, \mathcal{A}_2)$.)

In the interactive-style definition, the interactive adversary $\mathcal{A}'$ receives a public key, makes decryption queries, submit two plaintexts, get a ciphertext, makes new decryption queries, and produces a bit. We define $\mathcal{A}'$ from $(\mathcal{A}_1, \mathcal{A}_2)$ as follows:

Algorithm $\mathcal{A}'$

wait until K_p is received

simulate $\mathcal{A}_1(K_p)$; any query to $\mathcal{O}_1$ by this simulation is done by making decryption queries to the challenger

the simulation ends by producing (X_0, X_1, σ)

submit (X_0, X_1) to the challenger and get Y in return

simulate $\mathcal{A}_2(\sigma, Y)$; any query to $\mathcal{O}_2$ by this simulation is done by making decryption queries to the challenger

the simulation ends by producing b'

return b'

Clearly, the IND-CCA *game with $(\mathcal{A}_1, \mathcal{A}_2)$ is perfectly simulated by the interactive game with $\mathcal{A}'$. Hence, the advantages match.*

Conversely, given an interactive adversary $\mathcal{A}'$, we define $(\mathcal{A}_1, \mathcal{A}_2)$ as follows:

Algorithm $\mathcal{A}_1(K_p)$

simulate $\mathcal{A}'$ who starts by receiving K_p; any decryption query defines a query to $\mathcal{O}_1$ and the simulated answer to query is made from the answer to the oracle

at some point, $\mathcal{A}'$ issues (X_0, X_1), we let σ be the state of the simulation

return (X_0, X_1, σ)

Algorithm $\mathcal{A}_2(\sigma, Y)$

resume the simulation of $\mathcal{A}'$ from state σ, by starting from the reception of Y; any decryption query defines a query to $\mathcal{O}_2$ and the simulated answer to query is made from the answer to the oracle

the simulation ends by releasing a bit b'

return b'

Again, the simulation is perfect. Hence, the advantages match.

Q.2 Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an IND-CCA adversary. We define another IND-CCA adversary as follows:

Algorithm $\mathcal{B}_1^{\mathcal{O}_1(\cdot)}(K_p)$

```
simulate A_1^{O_1(·)}(K_p) → (X_0, X_1, σ)
if X_0 = X_1 then
    set σ' ← (σ, 1)
    pick an arbitrary X such that X ≠ X_1
    return (X, X_1, σ')
else
    set σ' ← (σ, 0)
    return (X_0, X_1, σ')
```

end if

Algorithm $\mathcal{B}_2^{\mathcal{O}_2(\cdot)}(\sigma', Y)$

parse $\sigma' = (\sigma, c)$

if $c = 1$ **then**

  return 0

else

  simulate $\mathcal{A}_2^{\mathcal{O}_2(\cdot)}(\sigma, Y) \to b'$

  return b'

end if

Prove that

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}CCA}}(s) = \mathsf{Adv}_{\mathcal{B}}^{\mathsf{IND\text{-}CCA}}(s)$$

Deduce that we can always assume $X_0 \neq X_1$ in an $\mathsf{IND\text{-}CCA}$ adversary.

Let E be the event that $X_0 = X_1$ with adversary $\mathcal{A}$. We have

$$\Pr[\mathsf{IND\text{-}CCA}^1_{\mathcal{A}}(s) \to 1|E] = \Pr[\mathsf{IND\text{-}CCA}^0_{\mathcal{A}}(s) \to 1|E]$$

thus

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}CCA}}(s) = \Pr[\mathsf{IND\text{-}CCA}^1_{\mathcal{A}}(s) \to 1, \neg E] - \Pr[\mathsf{IND\text{-}CCA}^0_{\mathcal{A}}(s) \to 1, \neg E]$$

We have

$$\Pr[\mathsf{IND\text{-}CCA}^b_{\mathcal{B}}(s) \to 1] = \Pr[\mathsf{IND\text{-}CCA}^b_{\mathcal{A}}(s) \to 1, \neg E]$$

hence

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}CCA}}(s) = \mathsf{Adv}_{\mathcal{B}}^{\mathsf{IND\text{-}CCA}}(s)$$

We now define the $\mathsf{NM\text{-}CCA}$ game (for non-malleability) as follows:

Game $\mathsf{NM\text{-}CCA}^b_{\mathcal{A}}(s)$

$(K_p, K_s) \leftarrow \mathsf{Gen}(1^s)$

$(M, \sigma) \leftarrow \mathcal{A}_1^{\mathcal{O}_1(\cdot)}(K_p)$ ▷ σ is a "state" which allows $\mathcal{A}_1$ to transmit data to $\mathcal{A}_2$

$X_0 \leftarrow M$ ▷ M is a sampling algorithm defined by $\mathcal{A}_1$

$X_1 \leftarrow M$ ▷ we sample two independent plaintexts using M

$Y \leftarrow \mathsf{Enc}_{K_p}(X_1)$

$(R, Y'_1, \ldots, Y'_n) \leftarrow \mathcal{A}_2^{\mathcal{O}_2(\cdot)}(\sigma, Y)$ ▷ R is a poly. algo. returning a boolean

$X'_i \leftarrow \mathsf{Dec}_{K_s}(Y'_i)$, $i = 1, \ldots, n$

if $Y \notin \{Y'_1, \ldots, Y'_n\}$ and $\perp \notin \{X'_1, \ldots, X'_n\}$ and $R(X_b, X'_1, \ldots, X'_n)$ **then**

```
    return 1
else
    return 0
end if
```

We use the same oracles $\mathcal{O}_1$ and $\mathcal{O}_2$ as for $\mathsf{IND\text{-}CCA}$. We define

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{NM\text{-}CCA}}(s) = \Pr[\mathsf{NM\text{-}CCA}_{\mathcal{A}}^{1}(s) \to 1] - \Pr[\mathsf{NM\text{-}CCA}_{\mathcal{A}}^{0}(s) \to 1]$$

We say that the cryptosystem is $\mathsf{NM\text{-}CCA}$ secure if for all probabilistic polynomial time (PPT) adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{NM\text{-}CCA}}(s)$ is negligible.

The goal of this exercise is to show the equivalence between $\mathsf{NM\text{-}CCA}$ security and $\mathsf{IND\text{-}CCA}$ security.

Q.3 We assume that $\mathcal{M}$ has a group structure (additively denoted), with at least two different elements 0 and 1, 0 being neutral. Assume that there is a polynomial algorithm Inc such that for all s,

$$\Pr\left[\mathsf{Dec}_{K_s}(\mathsf{Inc}_{K_p}(\mathsf{Enc}_{K_p}(X))) = X + 1\right] = 1$$

for $(K_p, K_s) \leftarrow \mathsf{Gen}(1^s)$. By constructing an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, prove that the cryptosystem is not $\mathsf{NM\text{-}CCA}$ secure.

(The precision of the proof is important.)

HINT: use M sampling in a set of two different plaintexts and R defined by $R(X, X') = 1_{X'=X+1}$.

Algorithm $\mathcal{A}_1^{\mathcal{O}_1(\cdot)}(K_p)$

pick $z, z' \in \mathcal{M}$ *such that* $z \neq z'$

define M *sampling in* $\{z, z'\}$ *with uniform distribution*

return (M, K_p) $\quad \triangleright$ *we set* $\sigma = K_p$

Algorithm $\mathcal{A}_2^{\mathcal{O}_2(\cdot)}(K_p, Y)$

$Y' \leftarrow \mathsf{Inc}_{K_p}(Y)$

define R *by* $R(X, X') = 1_{X'=X+1}$

return (R, Y')

Since $0 \neq 1$ *in* $\mathcal{M}$, *we have that* $\mathsf{Dec}_{K_s}(Y') = \mathsf{Dec}_{K_s}(Y) + 1 \neq \mathsf{Dec}_{K_s}(Y)$ *so* $Y' \neq Y$. *We further have* $\mathsf{Dec}_{K_s}(Y') \neq \perp$. *So, the outcome of the game is*

$$\mathsf{NM\text{-}CCA}_{\mathcal{A}}^{b}(s) = R(X_b, \mathsf{Dec}_{K_s}(Y')) = R(X_b, X_1 + 1) = 1_{X_1+1=X_b+1} = 1_{X_1=X_b}$$

thanks to the group property.

In the NM-CCA *game, if* M *picks two identical plaintexts* $X_0 = X_1$, *then the outcome of the game is always 1 no matter what is b. If* $X_0 \neq X_1$, *the outcome of the game is* $1_{b=1}$. *Hence*

$$\Pr[\mathsf{NM\text{-}CCA}_{\mathcal{A}}^{1}(s) \rightarrow 1] = 1$$

and

$$\Pr[\mathsf{NM\text{-}CCA}_{\mathcal{A}}^{0}(s) \rightarrow 1] = \frac{1}{2}$$

Therefore, we have

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{NM\text{-}CCA}}(s) = \frac{1}{2}$$

Q.4 Given an NM-CCA adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we construct an IND-CCA adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ as follows:

Algorithm $\mathcal{B}_1^{\mathcal{O}_1(\cdot)}(K_p)$

simulate $\mathcal{A}_1^{\mathcal{O}_1(\cdot)}(K_p) \rightarrow (M, \sigma)$

sample $z_0 \leftarrow M$

sample $z_1 \leftarrow M$

set $\sigma' \leftarrow (z_0, z_1, \sigma)$

return (z_0, z_1, σ')

Algorithm $\mathcal{B}_2^{\mathcal{O}_2(\cdot)}(\sigma', Y)$

parse $\sigma' = (z_0, z_1, \sigma)$

simulate $\mathcal{A}_2^{\mathcal{O}_2(\cdot)}(\sigma, Y) \rightarrow (R, Y'_1, \ldots, Y'_n)$

for $i = 1, \ldots, n$ **do**

  if $Y = Y'_i$ **then return** 0

  $X'_i \leftarrow \mathcal{O}_2(Y'_i)$

if $X'_i = \perp$ **then return** 0
end for
compute $b' \leftarrow R(z_1, X'_1, \ldots, X'_n)$
return b'

Prove that

$$\mathsf{Adv}_{\mathcal{B}}^{\mathsf{IND\text{-}CCA}}(s) = \mathsf{Adv}_{\mathcal{A}}^{\mathsf{NM\text{-}CCA}}(s)$$

Deduce that $\mathsf{IND\text{-}CCA}$ security implies $\mathsf{NM\text{-}CCA}$ security.

We first observe that since we check that $Y \neq Y'_i$, there is no problem to query $\mathcal{O}_2(Y'_i)$. By denoting $X_1 = z_b$ and $X_0 = z_{1-b}$, we can see that the $\mathsf{IND\text{-}CCA}_{\mathcal{B}}^b$ game is a perfect simulation of the $\mathsf{NM\text{-}CCA}_{\mathcal{A}}^b$ game (with some steps moved from the core game or adversary and decryption replaced by $\mathcal{O}_2$). Hence

$$\mathsf{IND\text{-}CCA}_{\mathcal{B}}^b = \mathsf{NM\text{-}CCA}_{\mathcal{A}}^b$$

thus

$$\mathsf{Adv}_{\mathcal{B}}^{\mathsf{IND\text{-}CCA}}(s) = \mathsf{Adv}_{\mathcal{A}}^{\mathsf{NM\text{-}CCA}}(s)$$

Q.5 We assume that $\mathcal{M}$ has at least four elements.
Given an $\mathsf{IND\text{-}CCA}$ adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we construct an $\mathsf{NM\text{-}CCA}$ adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ as follows:

Algorithm $\mathcal{B}_1^{\mathcal{O}_1(\cdot)}(K_p)$
simulate $\mathcal{A}_1^{\mathcal{O}_1(\cdot)}(K_p) \rightarrow (z_0, z_1, \sigma)$
define M sampling in $\{z_0, z_1\}$ with uniform distribution
set $\sigma' \leftarrow (\sigma, K_p, z_0, z_1)$
return (M, σ')

Algorithm $\mathcal{B}_2^{\mathcal{O}_2(\cdot)}(\sigma', Y)$
parse $\sigma' = (\sigma, K_p, z_0, z_1)$
take an injective function T on $\mathcal{M}$ such that $T(z_0) \notin \{z_0, z_1\}$ and $T(z_1) \notin \{z_0, z_1\}$
simulate $\mathcal{A}_2^{\mathcal{O}_2(\cdot)}(\sigma, Y) \rightarrow b'$
$Y' \leftarrow \mathsf{Enc}_{K_p}(T(z_{b'}))$
define $R(X, X') = 1_{T(X)=X'}$
return (R, Y')

Prove that

$$\mathsf{Adv}_{\mathcal{B}}^{\mathsf{NM\text{-}CCA}}(s) = \frac{1}{2}\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}CCA}}(s)$$

Deduce that $\mathsf{NM\text{-}CCA}$ security implies $\mathsf{IND\text{-}CCA}$ security.
HINT$_1$: assume without loss of generality that $z_0 \neq z_1$
HINT$_2$: compute $\Pr[X_0 = z_{b'}]$, $\Pr[X_1 = z_{b'} | X_1 = z_1]$, and $\Pr[X_1 = z_{b'} | X_1 = z_0]$.

Using Q.2, we can always transform the adversary to obtain $z_0 \neq z_1$. So, we assume $z_0 \neq z_1$ without loss of generality.
Due to correctness, we note that no decryption abort, so the outcome is

$$\mathsf{NM\text{-}CCA}^b_{\mathcal{B}}(s) = 1_{R(X_b,\mathsf{Dec}_{K_s}(Y'))=1, Y\neq Y'} = 1_{R(X_b,T(z_{b'}))=1,Y\neq Y'} = 1_{T(X_b)=T(z_{b'}),Y\neq Y'}$$

where Y is an encryption of X_1 and Y' is an encryption of $T(z_{b'})$. Given the assumptions on T, we always have $X_1 \neq T(z_{b'})$. Due to the correctness of decryption, we deduce that we always have $Y \neq Y'$. Due to injectivity, we deduce

$$\Pr[\mathsf{NM\text{-}CCA}^b_{\mathcal{B}}(s) = 1] = \Pr[X_b = z_{b'}]$$

Hence,

$$\mathsf{Adv}^{\mathsf{NM\text{-}CCA}}_{\mathcal{B}}(s) = \Pr[X_1 = z_{b'}] - \Pr[X_0 = z_{b'}]$$

We have

$$\mathsf{Adv}^{\mathsf{NM\text{-}CCA}}_{\mathcal{B}}(s) = \Pr[X_1 = z_{b'}] - \Pr[X_0 = z_{b'}]$$

Since b' only depends on X_1, X_0 is independent from $z_{b'}$ so $\Pr[X_0 = z_{b'}] = \frac{1}{2}$ (because $z_0 \neq z_1$). Similarly, we have $\Pr[X_1 = z_{b'}|X_1 = z_c] = \Pr[b' = c|X_1 = z_c]$ for $c \in \{0,1\}$. Thus, we have

$$\Pr[X_1 = z_{b'}|X_1 = z_1] = \Pr[\mathsf{IND\text{-}CCA}^1_{\mathcal{A}}(s) = 1]$$
$$\Pr[X_1 = z_{b'}|X_1 = z_0] = 1 - \Pr[\mathsf{IND\text{-}CCA}^0_{\mathcal{A}}(s) = 1]$$

Since $\Pr[X_1 = z_0] = \Pr[X_1 = z_1] = \frac{1}{2}$,

$$\Pr[X_1 = z_{b'}] = \frac{\Pr[\mathsf{IND\text{-}CCA}^1_{\mathcal{A}}(s) = 1] + 1 - \Pr[\mathsf{IND\text{-}CCA}^0_{\mathcal{A}}(s) = 1]}{2}$$

Therefore

$$\mathsf{Adv}^{\mathsf{NM\text{-}CCA}}_{\mathcal{B}}(s) = \frac{1}{2}\left(\Pr[\mathsf{IND\text{-}CCA}^1_{\mathcal{A}}(s) = 1] - \Pr[\mathsf{IND\text{-}CCA}^0_{\mathcal{A}}(s) = 1]\right)$$

Therefore

$$\mathsf{Adv}^{\mathsf{NM\text{-}CCA}}_{\mathcal{B}}(s) = \frac{1}{2}\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{A}}(s)$$

3 Unruh Transform from Σ to NIZK

> *This exercise is inspired from Unruh,* Non-Interactive Zero-Knowledge Proofs in the Quantum Random Oracle Model*, EUROCRYPT 2015, LNCS vol. 9057, Springer.*

We consider a Σ protocol (P, V) for a relation R. We let E be the set of challenges. Given some parameters t and $m \geq 2$, we define the following non-interactive zero-knowledge proof (NIZK), with input (x, w) such that $R(x, w)$ holds:

Algorithm $\mathsf{Proof}(x, w)$:

for $i = 1$ to t **do**
    pick a sequence of fresh coins ρ_i
    set $a_i \leftarrow P(x, w; \rho_i)$
    for $j = 1$ to m **do**
        pick $e_{i,j} \in E - \{e_{i,1}, \ldots, e_{i,j-1}\}$ at random
        set $z_{i,j} \leftarrow P(x, w, e_{i,j}; \rho_i)$
        set $h_{i,j} \leftarrow G(z_{i,j})$
    end for
end for
set $h \leftarrow H(x, (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m})_{i=1,\ldots,t})$
set $(J_1, \ldots, J_t) \leftarrow h$
set $z_i = z_{i,J_i}$ for $i = 1, \ldots, t$
set $\pi = (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m}, z_i)_{i=1,\ldots,t}$
return π

This algorithm uses two random oracles G and H. Oracle H is assumed to return a t-tuple of integers between 1 and m. We use the following verification algorithm (with some missing step):

Algorithm $\mathsf{Verify}(x, \pi)$:

parse $\pi = (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m}, z_i)_{i=1,\ldots,t}$
set $h \leftarrow H(x, (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m})_{i=1,\ldots,t})$
set $(J_1, \ldots, J_t) \leftarrow h$
verify $\cdots$
verify $V(x, a_i, e_{i,J_i}, z_i)$ for $i = 1, \ldots, t$
verify $h_{i,J_i} = G(z_i)$ for $i = 1, \ldots, t$
return 1 if all verifications passed

Q.1 By taking the verification with the missing step, give an algorithm to forge a proof given x but without the knowledge of w.
Which step should be added to have a sound proof?

We use the simulator of the Σ protocol and all $e_{i,j}$ equal:

Algorithm $\mathsf{Forge}(x)$*:*

pick $e \in E$ at random
$(a, e, z) \leftarrow \mathcal{S}(x, e)$
set $a_i = a$ for $i = 1, \ldots, t$
set $e_{i,j} = e$ for $i = 1, \ldots, t,\ j = 1, \ldots, m$
set $z_{i,j} = z$ for $i = 1, \ldots, t,\ j = 1, \ldots, m$
set $h_{i,j} = G(z)$ for $i = 1, \ldots, t,\ j = 1, \ldots, m$
set $h \leftarrow H(x, (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m})_{i=1,\ldots,t})$
set $(J_1, \ldots, J_t) \leftarrow h$
set $z_i = z_{i,J_i}$ for $i = 1, \ldots, t$
set $\pi = (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m}, z_i)_{i=1,\ldots,t}$
return π

It is clear that the output π passes the verification with the missing step. The missing step is

for $i = 1$ *to* t ***do***
  verify that $e_{i,1}, \ldots, e_{i,m}$ are pairwise different
end for

Q.2 With the new verification step from the last question, give an algorithm with complexity $\mathcal{O}(m^t)$ to forge a valid π from x but without w.

We try to predict the index of the challenges which will be verified and use the simulator of the Σ protocol. We proceed as follows:

Algorithm $\mathsf{Forge}(x)$*:*

repeat
  for $i = 1$ *to* t ***do***
    pick $J_i \in \{1, \ldots, m\}$
    pick $e_{i,J_j} \in E$ *at random*
    $(a_i, e_{i,J_i}, z_i) \leftarrow \mathcal{S}(x, e_{i,J_i})$
    for $j = 1$ *to* m ***do***
      if $j \neq J_i$ ***then***
        pick $e_{i,j} \in E - \{e_{i,1}, \ldots, e_{i,j-1}, e_{i,J_i}\}$ *at random*
        set $z_{i,j}$ *at random*
        set $h_{i,j} \leftarrow G(z_{i,j})$
      end if
    end for
  end for
  set $h \leftarrow H(x, (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m})_{i=1,\ldots,t})$
until $(J_1, \ldots, J_t) = h$
set $z_i = z_{i,J_i}$ *for* $i = 1, \ldots, t$
set $\pi = (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m}, z_i)_{i=1,\ldots,t}$
return π

Since h randomly picks $J_1, \ldots, J_t$, each iteration succeeds with probability m^{-t}. Hence, we need $\mathcal{O}(m^t)$ iterations until we succeed.

Q.3 Construct a simulator in the random oracle model to show that the protocol is non-interactive zero-knowledge.

If finding a witness is easy, the problem is trivial: we just use the easy-to-find witness to simulate the proof as Proof*.*
If finding the witness is hard, the simulator works like in the previous question.

Algorithm Simulate(x)*:*

for $i = 1$ *to* t ***do***
  pick $J_i \in \{1, \ldots, m\}$
  pick $e_{i,J_j} \in E$ *at random*
  $(a_i, e_{i,J_i}, z_i) \leftarrow \mathcal{S}(x, e_{i,J_i})$
  set $h_{i,J_j} \leftarrow G(z_{i,J_j})$ ▷ *simulate G*
  for $j = 1$ *to* m ***do***
    if $j \neq J_i$ ***then***
      pick $e_{i,j} \in E - \{e_{i,1}, \ldots, e_{i,j-1}, e_{i,J_i}\}$ *at random*
      set $h_{i,j}$ *at random*
    end if
  end for
end for
set $h \leftarrow (J_1, \ldots, J_m)$
set $h = H(x, (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m})_{i=1,\ldots,t})$ ▷ *simulate H*
set $z_i = z_{i,J_i}$ *for* $i = 1, \ldots, t$
set $\pi = (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m}, z_i)_{i=1,\ldots,t}$
return π

We should show that a limited distinguisher receiving π and playing with the simulator of G and H cannot distinguish this π from a genuine one. For this, we should argue that it cannot find the correct $z_{i,j}$ for $j \neq J_i$, except with negligible probability (because he would, together with a_i, e_{i,J_i}, and z_{i,J_i}, be able to extract a witness with the Σ extractor, which is assumed to be hard). Without being able to query G with the right $z_{i,j}$, the value $h_{i,j}$ is free. Thus, the distinguisher cannot see if $h_{i,j}$ was randomly selected by the simulator without knowing $z_{i,j}$ or randomly selected by G.

Q.4 Let $P^*(x)$ be an algorithm taking x as input, interacting with G and H, and forging a valid π with probability p. Use the next questions to prove that there is an extractor who can run P^* once to extract a witness w for x with probability at least $p -$ negl.

Q.4a Transform P^* into an algorithm P' who either aborts or makes a valid π. It returns π with probability p, and a complexity similar to P^*.

The algorithm $P'(x)$ first runs $P^*(x)$ and obtain π. Then, it parses $\pi = (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m}, z_i)_{i=1,\ldots,t}$ and runs Verify(x, π)*. If verification fails, P' aborts. Otherwise, it returns π.*
Clearly, the probability of success is the same and the complexity is similar.
Note that P' always queries $H(x, (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m})_{i=1,\ldots,t}) = (J_1, \ldots, J_t)$. If also queries $G(z_i) = h_{i,J_i}$ for every i.

Q.4b Construct an extractor E on the previous P' such that by observing only one execution of P' with all queries to G and H, either P' aborts, or E finds a witness for x, or E aborts. But the probability that E aborts is bounded by $n_G n_H mtN^{-1} + n_H m^{-t}$, where n_G is the number of queries to G, n_H is the number of queries to H, and N is the size of the range of G.

Hint: say that a query q to H is good if it can be parsed in the form

$$q = x, (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m})_{i=1,\ldots,t}$$

Consider an extractor which aborts if any fresh query to G returns a value $h_{i,j}$ which is included in a previous good query q to H. Define another abort condition and extract a witness in remaining cases.

We consider an execution of P' with all its queries to G and H.
Let q be a fresh query to H by P'. We say that q is a good fresh query if q parses into some $q = x, (a_i, (e_{i,j}, h_{i,j})_{j=1,\ldots,m})_{i=1,\ldots,t}$ such that for every i, all $e_{i,j}$ are pairwise different. So, q defines a sequence of a_i, and arrays of $e_{i,j}$ and $h_{i,j}$.
Note that if P' succeeds to forge a valid π, there must be a good fresh query q which matches the content of π.
If any fresh query to G after the query q to H returns one of the values $h_{i,j}$ (there are mt of them), the extractor aborts. So, the probability to abort for this case is bounded by $n_G n_H mtN^{-1}$.
For each good fresh q, we define $J_q(i)$ as the set of j such that $h_{i,j}$ was returned by G at some point in the past. (Note that unless the extractor aborts, there won't be any future query to G returning $h_{i,j}$.) We let $J'_q(i)$ be the subset of $J_q(i)$ such that there exists one query $z_{i,j}$ to G which returned $h_{i,j}$ and satisfying the condition $V(x, a_i, e_{i,j}, z_{i,j})$. If there is any i such that $J'_q(i)$ has at least two elements, we can use the Σ extractor to get a witness for x.
Now, we consider the case where for all i, $J'_q(i)$ has at most one element. When H returns $(J_1, \ldots, J_t)$ to the fresh query q, if we have that $J_i \in J'_q(i)$ for all i, then we make the extractor abort. Clearly, the probability this happens is bounded by m^{-t}. Applying this to all queries to H, the probability to abort is bounded by nm^{-t}.
If the extractor does not abort and P' succeeds to make a valid π, we note that there is a good query q to H made by the verification. We take the fresher query equal to q. We also note that for all i, the verification in P' makes a query $G(z_i) = h_{i,J_i}$ for each i. So, z_i cannot be a fresh query and we must have $J_i \in J'_q(i)$ for all i. Hence, either E succeeded to extract a witness or it aborted on that fresh good query.