HOMEWORK 2 – CRYPTOGRAPHY

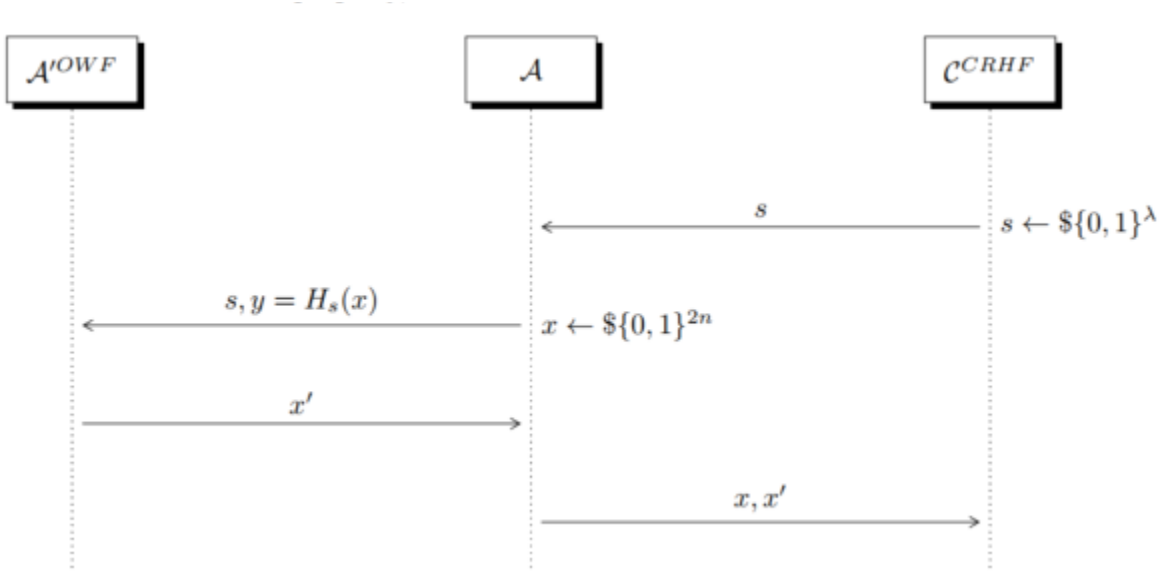
1 Hashing

(a) (i) **Solution:**

The property we want to prove is the following:

H is CRHF $\sqsupset$ H is OWF

To show this property, we have to make a reduction:



When does A lose?
Since CRHF game needs the final couple (x; x'), where x != x', if A $'^{OWF}$ returns
$x' = x$ the CRHF game doesn't work.

This **BAD** event happens when:

$$\mathcal{P}[x = x'] = Col(X, X') = \sum_{x} \mathcal{P}[X = x \wedge X' = x] = \sum_{x} \mathcal{P}[X = x]\mathcal{P}[X' = x] = \frac{1}{2^{2n}}$$

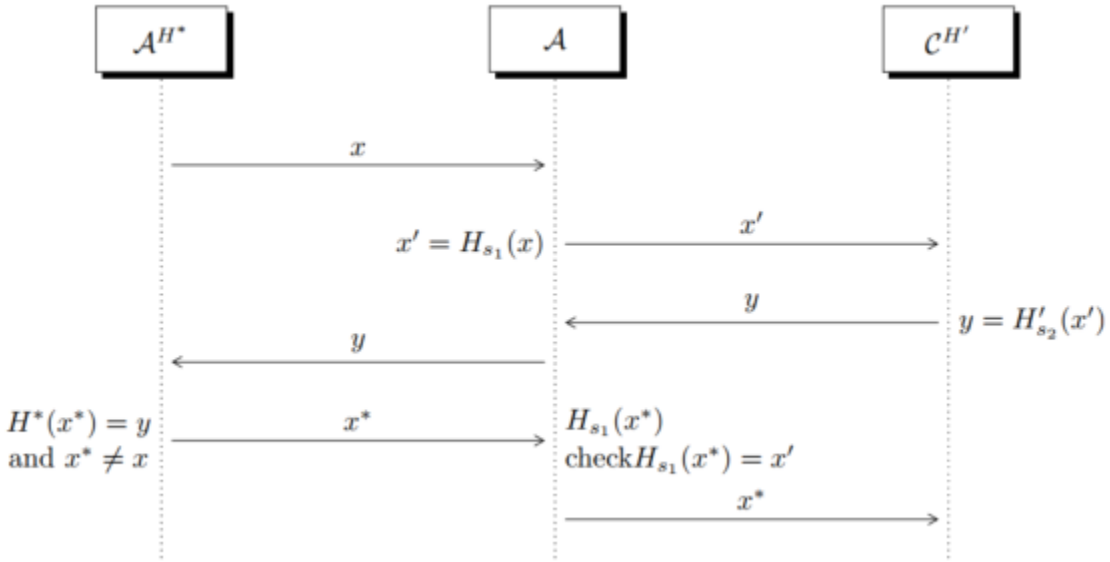
(ii) **Solution:**

If the function is not compressing ($h : n \rightarrow n$) and it is collision resistant ,we can take the best hash function possible (a bijective one). In this case the function will obviously be CRHF since it is impossible to find $x \neq x'$ such that $h(x) = h(x')$.
However it will not be one-way since there will be a unique correspondence between an element in the domain and an element in the codomain.
So when given a 1-to-1 mapping should be easy to go from an element to another.

(b) **Solution:**

We are given $H^*_{s1;s2}(x) = H'_{s2}(H_{s1}(x))$ with $H^* : 4n \rightarrow n$.

Let's suppose that there $\exists A^{H^*}$ which is able to find a collision in H^*.
Now let's consider the following Game:



The "BAD event" is the one in which A^{H^*} outputs a collision for H_{s1}, meaning that $H_{s1}(x) = H_{s1}(x')$.

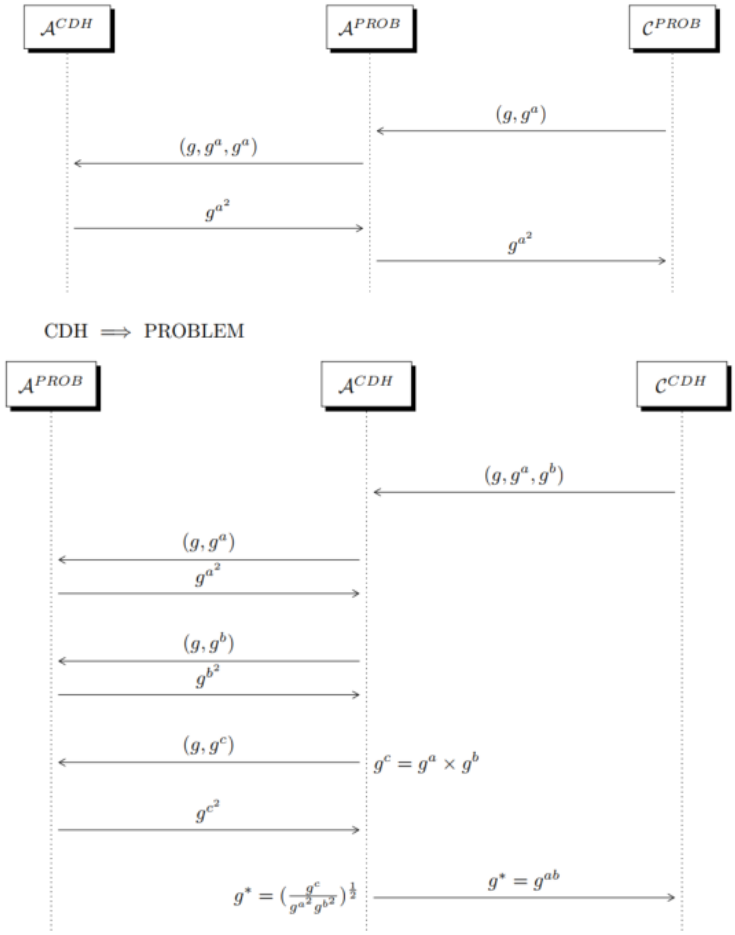
In this case, the second part of the reduction doesn't work. But $Pr[BAD]$ is negligible since H_{s1} is collision resistant by definition.

But now we have that $H^*(x') = H^*(x)$ since x' was a collision for H^* and this must be a collision also for H'_{s2} which was a CRHF for hypothesis.

2 Number Theory

(a) **Solution:**

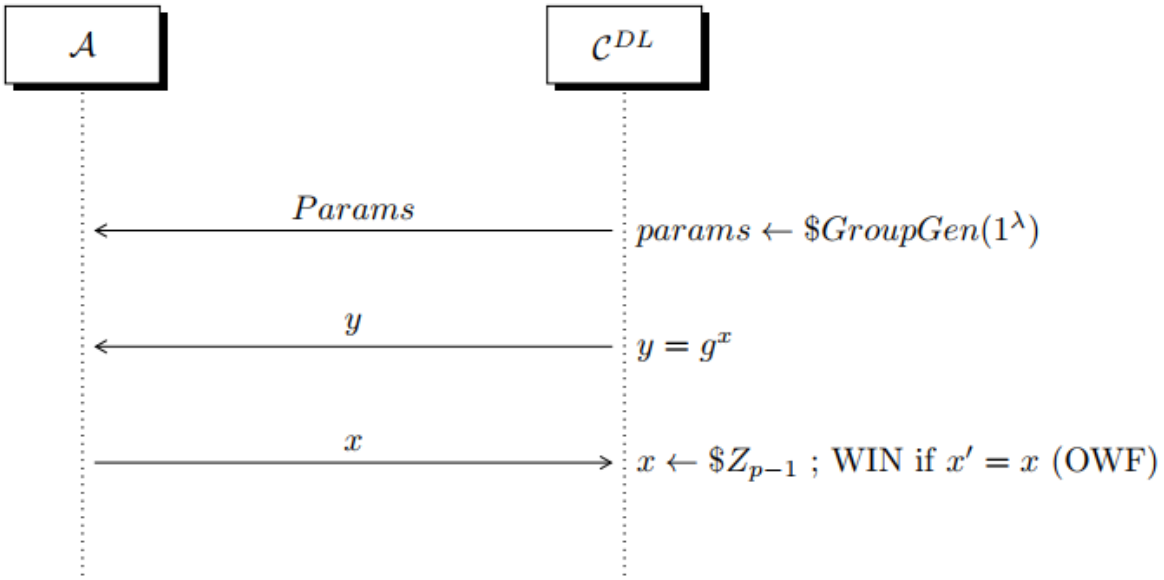
Let's call $(g; g^a)$ compute $g^{a\hat{}2}$ as PROBLEM.
Now we want to prove that PROBLEM $\Leftrightarrow$ CDH
PROBLEM -> CDH:
Let's suppose there $\exists A^{CDH}$ which is able to break CDH with non negligible probability.



CDH $\Longrightarrow$ PROBLEM

(b) **Solution:**

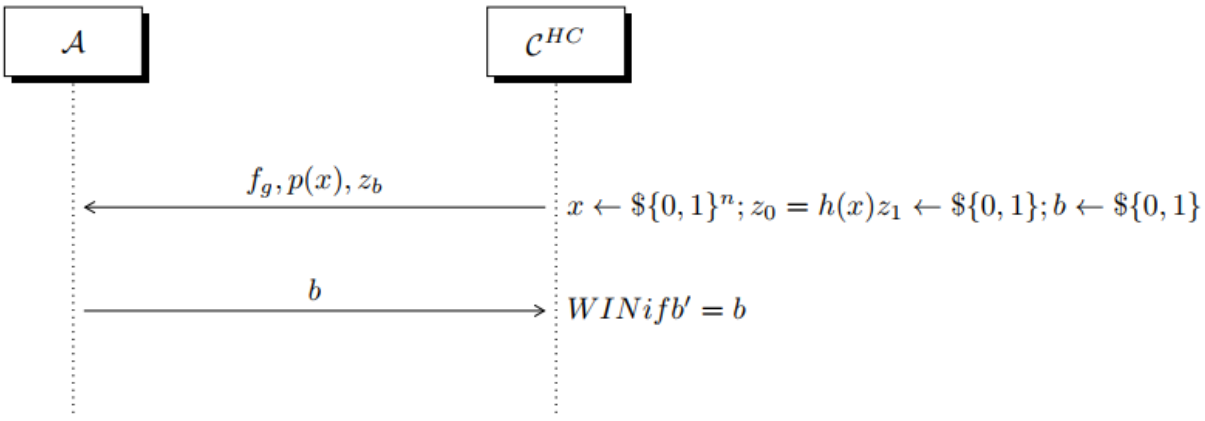
$f_{g;p}(x) := g^x \bmod p$ is one-way under the Discrete Logarithm assumptions.
We have:



DL is secure only if $DLOG_g(y) = log g^y$, and in that case DL is also an OWF.
A polynomial time function defines as $h(x) : \{0,1\}^n \to \{0,1\}$ is hard-core for the function $f_{g,p}$ IFF

$$(f_{g,p}(x), h(x)) \approx_c (f_{g,p}(x), b), b \leftarrow \$\{0,1\}$$

So to show that these two are distinguishable, we consider the following Game:



(c) **Solution:**

Let's consider that $x_i = xmodp_i, y_i = ymodp_i$.

x solves $x^2 = ymodN$ IFF $\begin{cases} x_1^2 \equiv y_1 modp_1 \\ x_5^2 \equiv ymodp_5 \end{cases}$

We will call this S. As first step, we will prove $n => n$:
$x^2 = ymodN \rightarrow \exists K \in Z$ such that $y = KN + x^2 = Kp_i...p_5 + x^2 \rightarrow$
$\begin{cases} y_1 \equiv x_1^2 modp_1 \\ y_5 \equiv x_5^2 modp_5 \end{cases}$ And as second step: n$\Leftarrow n$:
By CRT we know that if $x_1...x_5$ are solutions of S, then there doesn't exist a x such that $x \equiv x_i modp_i$ for i=1,...,5.

$S \longleftrightarrow \begin{cases} y_1 - x^2 \equiv 0 modp_i \\ y_5 - x^2 \equiv 0 modp_5 \end{cases}$

$y \equiv x^2 modN$ and y is the unique solution of $\begin{cases} y \equiv y_1 modp_1 \\ y \equiv y_5 modp_5 \end{cases}$

Now we will show that we can obtain the least significant bit of x from $f_{g,p}(x) = g^x modp$, so we test if g^x is a quadratic residue mod p.

$$(g^x)^{\frac{p-1}{2}} \equiv 1 modp$$

- if x is even $h(x) = 0$
  if $g^x = g^{2k} \rightarrow (g^{p-1})^k \equiv 1 modp$
- if x is odd $h(x) = 1$
  if $g^x = g^{2k+1} \rightarrow g^{(2k+1)\frac{p-1}{2}} \neq 1 modp$

With this, we have shown that we can obtain the least significant bit, so $h(x)$ is not hard-core.

3 Public-Key Encryption

(a) **Solution:**

I will prove this using contradiction. Let's assume that there ∃ an adversary $B = (B1; B2)$ who $\mathscr{E}$-breaks the IND-CPA security of E. Now let's construct an adversary $A = (Rand; Rev; Query; Dist)$ for the ROR-SCDA game with T=1 as follows:

- Rand always generates uniform random bits.
- Rev is the identity function.

• Query gets p_k and runs B1 on p_k. Then B1 outputs a message m and
some state. Query makes a single RoR query where Mesg outputs
the fixed message m. Query outputs the resulting ciphertext c as
well as the state.

• Dist outputs B2(c, state).
Since there is no communication between Mesg and Rand, it holds
that $c = 0$. Therefore we have that $H_\infty(pk)$ **-** $min(p; c) = H_\infty(pk)$.
Also, the randomness output by Rand is a l-source and the pair (m,r)
is a l-source conditioned on ρ. Therefore, the adversary A satisfies
the attack profile $(H_\infty (pk); l; l) = \Pi$.

(b) (i) **Solution:**
The goal here would be to demonstrate that if Π is $CCA1$-> Π’
is also CCA1. In order to do this let’s observe the following reduction scheme:
Let’s suppose there $\exists A^{\Pi 0'}$which is able to break the CCA1 security
of Π’

$A^{\Pi'}$ sends ciphertext composed of t elements. A takes every single
element and sends to C to get the plaintext of each one.
Then he recombines the plaintext and sends back the single plaintext to $A^{\Pi'}$.
At the start of the challenge, $A^{\Pi'}$ sends two messages: m_0, m_1 of t
bits to A. A sends to C the first t-1 bytes of m_0 and receives the
corresponding t **-** 1 ciphertexts and then A sends to C the challenge

as: $m_0[t]$ and 1, receiving the ciphertext $c^{*'}$ of one of the two. At this
point, A recombines all of the t-1 ciphertext plus the last one received,
$c^{*'}$, and sends back to A^{Π}. Now A just forwards the response. The probability will be

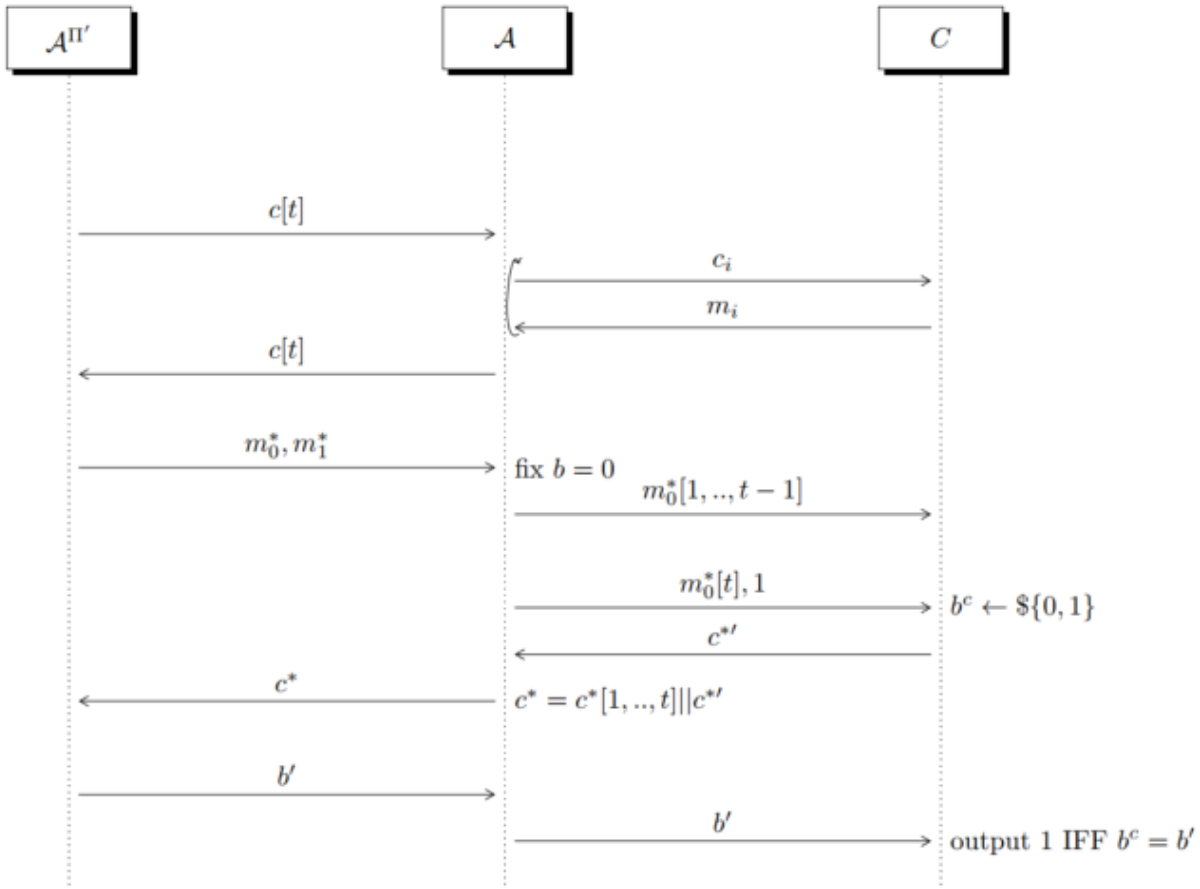
$$|P[A^{\Pi} = 0|b^c = 0] - P[A^{\Pi} = 0|b^c = 1]| = \frac{1}{2} + negl(\lambda) - \frac{1}{2} > negl(\lambda)$$

(ii) **Solution:**

Π CCA2 $\implies$ $\Pi'\neg$CCA2

Let's consider the following PKE Scheme:

- $Enc(P_k, m[t]) = Enc(P_k, m_1)||...||Enc(P_k, m_t)$
- $Dec(S_k, c[t]) = Dec(S_k, c_1)||...||Dec(S_k, c_t)$



Since in CCA2 we can make decryption queries after the challenge,
we can create a $c' != c^*$ just by inverting the first two bits of c^* ($c* = c^*_1||c^*_2||...||c^*_t$ now $c' = c^*_2||c^*_1||...||c^*_t$).

Now when we receive the decrypted message, we can simply switch the first two bits again and discover which of the two challenge messages was encrypted.

(c) **Solution:**
Based on assumptions that x is random and $h = g^x$, let's assume that
we have received a ciphertext (c_1; c_2) and that we know the secret

key
x.

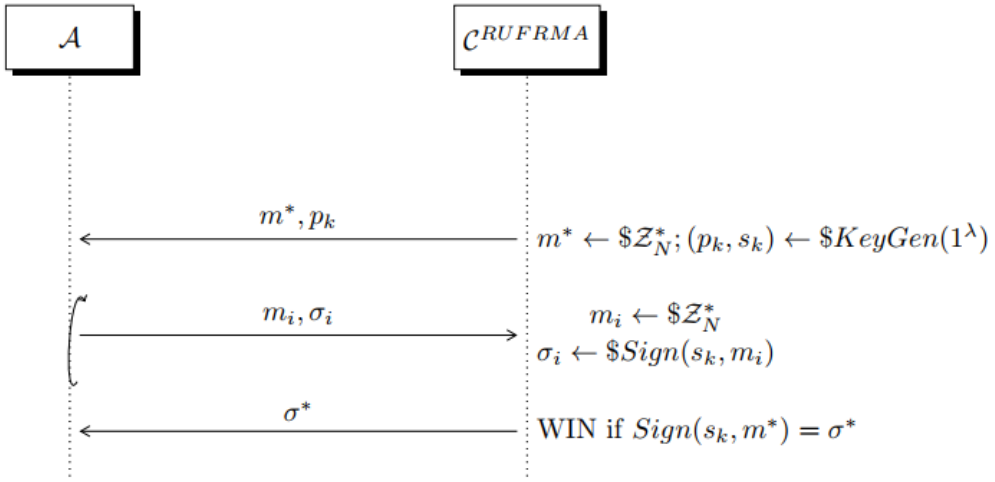
We shall compute $Dec(c_1) = c^x_1 = (g^y)^x = (g^r)^y = h^y$. Thus, if $Dec(c_1) = c_2$, then $c_2 = h^y$ and we decrypt to m=0. In this situation, we
know that either b=0 was encrypted and the decryption was correct, or
b=1 was encrypted and $\mathscr{E}$ was chosen such that $g^{\mathscr{E}} = h^y$ i.e. $\mathscr{E} = xy$. In
this case, the decryption was incorrect. But this will only happen with a
probability $1/Q$ which is negligible in n.

If $Dec(c1)$ $!= c2$, we can decrypt to m=1. This decryption is always correct.

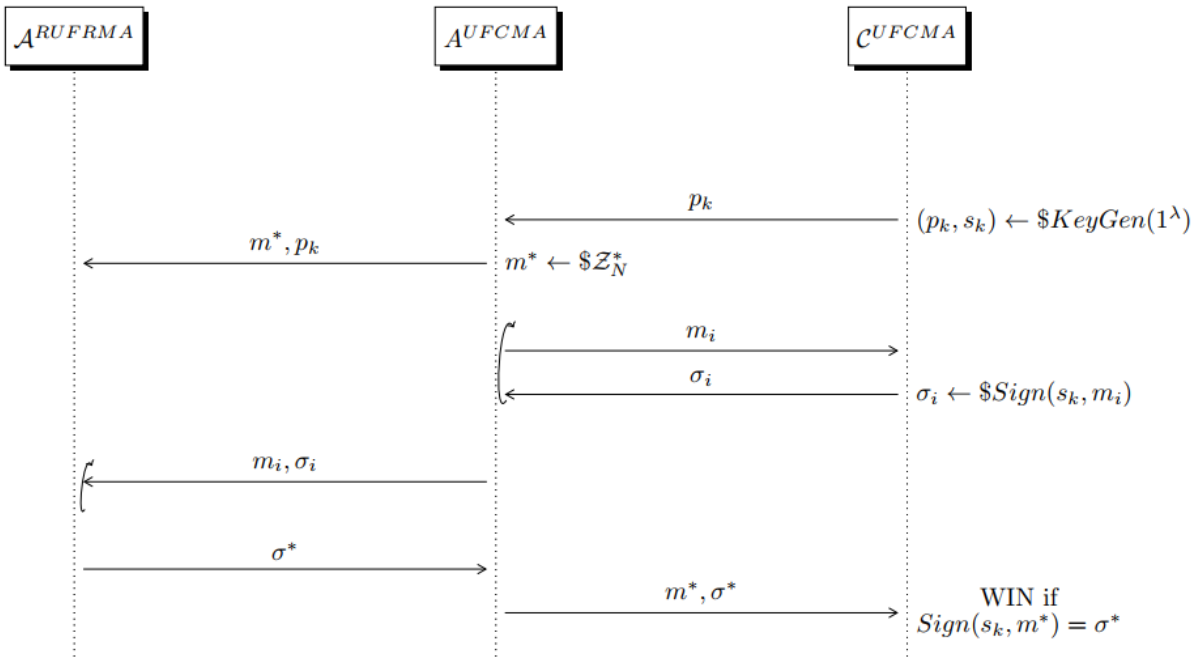
4 Signature Schemes

(a) **Solution:**

Let's consider the following Game:



UFCMA⇒ RUFRMA So we can use an A^{RUFRMA} to break UFCMA:



UFCMA$\not\Longleftarrow$ RUFRMA We can consider the previous Game but with an UFCMA distinguisher, but we can't break in this way RUFRMA, so because distinguisher receives random messages by the challenger and he can't choose the messages. For example, we can consider a case in which the challenger puts in σ_i, the i-th bit of s_k each time that m_i ends with bit 0. In UFCMA, we can't send messages that end with 0 and in RUFRMA we can't choose messages used to create various σ_i, so we can't know all s_k.

$$Pr[m_i[lastbit] = 0; \forall i \in 0, ..., n-1] = \frac{1}{2^n} \in negl(\lambda)$$

(b) **Solution:**

The above signature scheme doesn't satisfy UFCMA because an adversary can forge (m^*, σ^*) as is shown below:

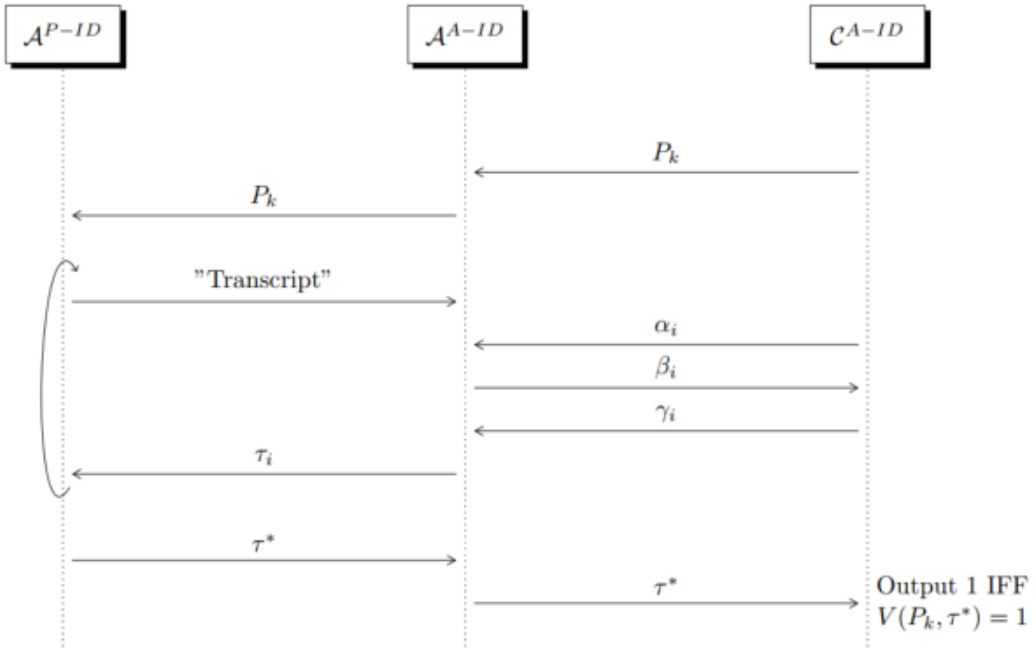
- Choose $\sigma^* \in Z_N^*$
- Compute $m^* = (\sigma^*)^2 mod N$

The above assumption instead satisfies RUFRMA because we can't compute m^* in this case because it is given to us randomly at the beginning.

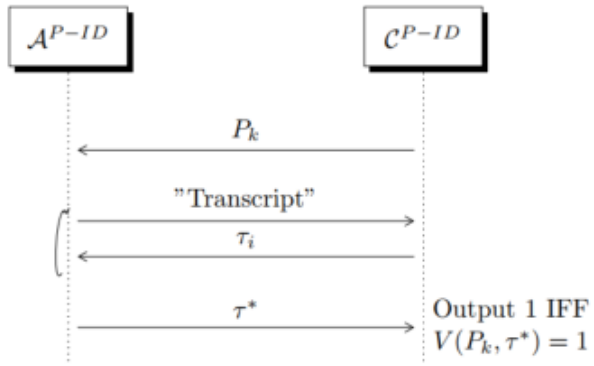
5 Actively Secure ID Schemes

(a) **Solution:**

We have to show that:
Active -> Passive
Let’s consider the following Game:

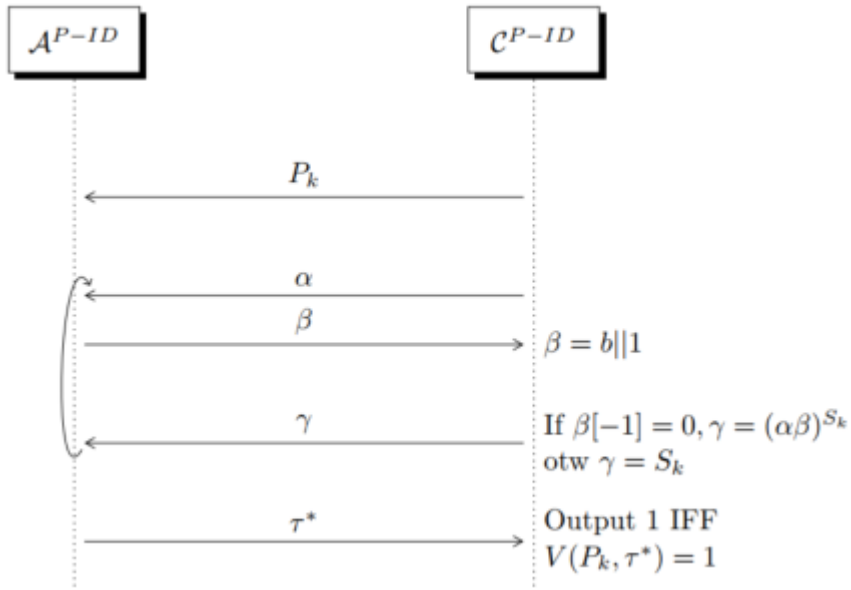


We can construct a Π_{BAD} such that Passive *!*-> Active, where if we play
as a dishonest Verifier we can leak the entire S_k.
For the passive Game:



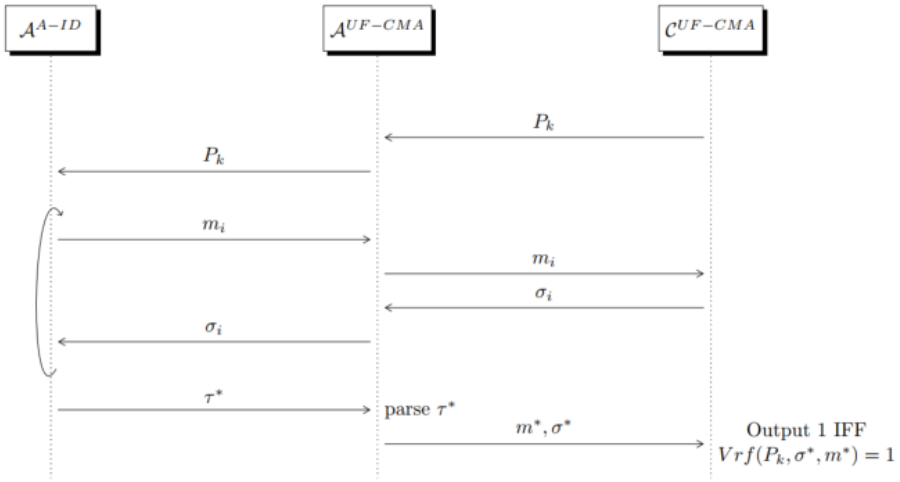
Now τi will be as follows, from the point of view of A:

- $\alpha \leftarrow U$
- $\beta = b||0$ where $b \leftarrow \$U$
- $\gamma = \beta[-1](S_k) + (1 - \beta[-1])(\alpha\beta)^{S_k}$



Now that we have S_k we can always create a valid τ^*.

(b) **Solution:**
Let's suppose there $\exists A^{A\text{-}ID}$ which is able to break the active security
of Π.



(c) **Solution:**
The HVZK property comes from the fact that the signature scheme is
used in the following way:

- The Verifier sends the message
- The Prover signs the message

doesn’t leak any information about the secret used for signing the message
(assuming *V* always as honest verifier). For the property of the signature
it will never reveal anything about the secret used to generate σ.