Anish Banerjee, Shankh Gupta **Problem Set - 4**
COL759: Cryptography November 2023

Problem : Collision Resistant Hashing based on number-theoretic assumptions

Solution:

(a) Let $\mathcal{A}$ be an adversary that can break the collision resistance property of the Hash function $H : \mathbb{G}^\lambda \times \mathbb{Z}_q^\lambda \to \mathbb{G}$

$$H((g_1, g_2, \ldots, g_\lambda), (\alpha_1, \alpha_2, \ldots, \alpha_\lambda)) = \prod_{i=1}^{\lambda} g_i^{\alpha_i}$$

with non-negligible probability ϵ. We will use $\mathcal{A}$ to build an adversary $\mathcal{B}$ which breaks DLOG assumption. Consider the reduction:

Reduction for (a)

- The RSA Challenger sends the tuple (q, g, g^α) to $\mathcal{B}$
- $\mathcal{B}$ samples $i^* \leftarrow [\lambda]$ and $\beta_1, \beta_2, \ldots, \beta_{\lambda-1} \leftarrow \mathbb{Z}_q$.
- $\mathcal{B}$ sends the key $(g^{\beta_1}, g^{\beta_2} \ldots g^\alpha, \ldots g^{\beta_{\lambda-1}})$ where g^α is the i^*th element of the tuple to $\mathcal{A}$
- $\mathcal{A}$ responds with a collision $(x_1, x_2, \ldots, x_{i^*}, \ldots, x_{\lambda-1}), (y_1, y_2, \ldots, y_{i^*}, \ldots y_{\lambda-1})$ such that

$$\left(\prod_{i=1}^{\lambda-1} g^{\beta_i x_i}\right) g^{\alpha x_{i^*}} = \left(\prod_{i=1}^{\lambda-1} g^{\beta_i y_i}\right) g^{\alpha y_{i^*}} \tag{1}$$

- Using the collision, $\mathcal{B}$ computes α as follows:
  Since $g^m = g^n \implies m = n$ for the generator g, we have

$$\left(\sum_{i=1}^{\lambda-1} \beta_i x_i\right) + \alpha x_{i^*} = \left(\sum_{i=1}^{\lambda-1} \beta_i y_i\right) + \alpha y_{i^*} \tag{2}$$

$$\alpha = (y_{i^*} - x_{i^*})^{-1} \sum_{i=1}^{\lambda-1} \beta_i (x_i - y_i) \tag{3}$$

Assuming $y_{i^*} \neq x_{i^*}$ (explained below). Note that all the operations in eqn (2) and (3) are in $\mathbb{Z}_q$

Figure 1: Reduction for (a)

As mentioned above, the reduction works only when $y_{i^*} \neq x_{i^*}$. Observe that at least two elements of the tuples $(x_1, x_2, \ldots, x_{i^*}, \ldots, x_{\lambda-1}), (y_1, y_2, \ldots, y_{i^*}, \ldots y_{\lambda-1})$ must be distinct (if only one element is different, then equation (2) makes them equal). Since the index i^* is chosen randomly, probability that i^* matches with the distinct elements is

$$\Pr[\text{Match}] \geq \frac{2}{\lambda}$$

Hence the winning probability of the reduction is $\Pr[\text{Win}] \geq \frac{2\epsilon}{\lambda}$

NOTES:

- Since g is the generator of the group, any random element can be written in the form g^β where β is sampled randomly. So, the key received by $\mathcal{A}$ is randomly generated.
- It is essential to sample i^* randomly. If we chose it to be fixed, then there may exist $\mathcal{A}$ which always outputs collision tuples matching at that specific index.

(b) Let us define an efficient adversary $\mathcal{A}$ which can break the CRHF security of $h_{N,e,z}$. In other words, given input (N, e, z), adversary $\mathcal{A}$ can output a collision (x_1, y_1) and (x_2, y_2), with $(x_1, y_1) \neq (x_2, y_2)$, such that $h_{N,e,z}(x_1, y_1) = h_{N,e,z}(x_2, y_2)$. We will show that if there exists such an adversary $\mathcal{A}$, then it is possible to construct a reduction algorithm $\mathcal{B}$ which uses $\mathcal{A}$ to break the RSA problem.
The algorithm $\mathcal{B}$ interacts with $\mathcal{A}$ and a challenger $\mathcal{C}$ as follows :

Reduction for (b)

- The challenger $\mathcal{C}$ runs the setup to obtain the key N, e and a random integer $z \leftarrow \mathbb{Z}_N^*$. It forwards (N, e, z) to reduction $\mathcal{B}$ which in turn forwards this to the adversary $\mathcal{A}$.
- The adversary then sends its collision as (x_1, y_1) and (x_2, y_2) to the reduction $\mathcal{B}$.
- $\mathcal{B}$ uses the output of adversary $\mathcal{A}$ to compute the e^{th} root of z, which it then forwards to the challenger and breaks the RSA assumption.

Figure 2: Reduction for (b)

$\mathcal{B}$, given a collision, computes the e^{th} root of z as follows:

- $\mathcal{A}$ forwards its collision (x_1, y_1) and (x_2, y_2) to $\mathcal{B}$ such that

$$x_1^e \cdot z^{y_1} = x_2^e \cdot z^{y_2}$$

  Observe that if $y_1 = y_2$ then $x_1^e = x_2^e$. This would result in x_1 being equal to x_2, since e is co-prime to $\phi(N)$ and the e^{th} root (i.e. the inverse of e) should be unique. Therefore it is not possible for adversary to output a collision of the form $(x_1, y), (x_2, y), x_1 \neq x_2$. Hence we can safely assume $y_2 > y_1$ without loss of generality.
- The reduction then computes the inverse of x_2 and z. Since we have $x_1^e \cdot z^{y_1} = x_2^e \cdot z^{y_2}$, we can multiply both sides by $(x_2^{-1})^e$ and $(z^{-1})^{y_1}$ to get

$$(x_1 x_2^{-1})^e = (z)^{y_2 - y_1}$$

  Let $a = x_1 x_2^{-1}$ and $b = y_2 - y_1$. So the reduction has computed $a \in \mathbb{Z}_N^*$ and $b \in \mathbb{Z}_e^*$, such that

$$a^e = z^b$$

  .
- $\mathcal{B}$ now needs to compute the b^{th} root of a. Observe that since b and e are co-prime, by the Extended Euclid's algorithm we can find m and n such that:

$$mb + ne = 1$$

  Now,

$$z^1 = z^{mb+ne} = a^{me} z^{ne} = (a^m z^n)^e$$

  Thus the e^{th} root of z is $a^m z^n$.

NOTES:

- Since x_2 and z both belong to $\mathbb{Z}_N^*$, so we can compute using Extended Euclid's Algorithm s, t, u and v such that

$$sx_2 + tN = 1$$

  and

$$uz + vN = 1$$

  Using these we can efficiently compute the inverse of both x_2 and z.
- $b \in \mathbb{Z}_e^*$. This is because both y_1 and y_2 belongs to $\mathbb{Z}_e$ and assuming $y_2 > y_1$ we can say that $b < e$. Also, it is not possible for y_1 to be equal to y_2 (as explained above). Hence $0 < b < e$, which implies it belongs to $\mathbb{Z}_e^*$.