CS276 Lecture 16-A

Prof. David Wagner
Scribe: Wei Xu (xuw@cs)

March 17, 2004

1 FDH is secure in random oracle model

The first thing for today is to prove the security of full domain hash in random oracle model. To make our lives easier, we consider FDH-RSA. Recall that in full domain hash:

$S(m) = H(m)^d \bmod n$, $H : 0, 1* \to \mathbb{Z}/n\,\mathbb{Z}$

"Full domain" means that the output function is between 0 and $n-1$.

Recall what a secure signature scheme is. The adversary got a oracle, and after a number of queries to the signing oracle, he does a forgery, which is a new message with valid signature.

1.1 A Lemma

To get some intuition, we first prove a lemma, in which the adversary does not make any query to signing oracle S.

Lemma 1 *If RSA is (t, ϵ) -trapdoor permutation, then full domain hash FDH-RSA is $(\frac{t}{c}, 0, t\epsilon)$-secure signature scheme.*

To proof this lemma, we assume, by the way of contradiction, that there is a forger A. We consider A as a black box with public key as an input. It also has two wires connecting to a random oracle H. With high probability, A outputs a pair (m, τ)m, in which m is a new message and $H(m) = \tau^3 \pmod{d}$.

A is illustrated in fig.1

We want to show that if any forger A exists, we can construct an inverter $B$of the trapdoor permutation. So B is given a public key, n, as well as input y, and want to output x s.t. $x^3 = y \pmod{n}$. (as illustrated in fig.**??**)

Hint 1: Assume that A always query H for a message m before it outputs m.

Hint 2: You can run A with different oracle.

Here is a (bad) idea: Let H always returns y on A's query. When A outputs a forgery, it outputs one of the messages it queried to H. The $H(m)$ will always be y, and we output τ, which is the cube root of y. In this way, we are tricking A to find cube root for us.

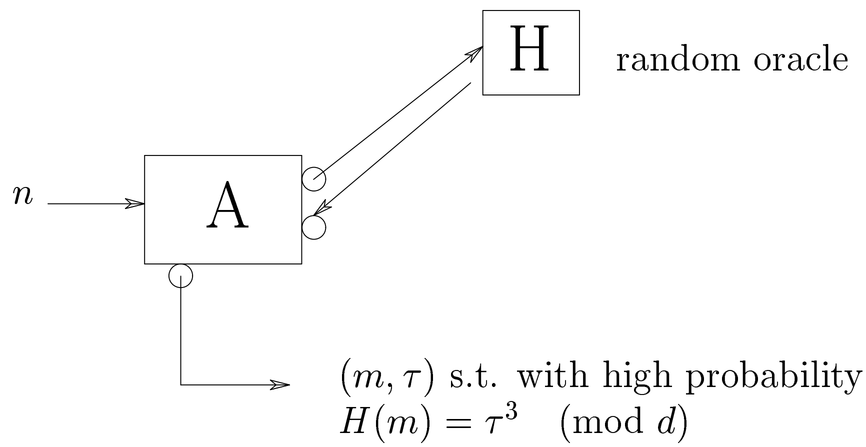

Figure 1: Adversary A

This is wrong because H is not a random oracle. A has high probability of forging, but the probability is taken over the choice of H. We have to be lucky to choose the correct H.

Here is a trick to patch it up.

Only for one of the queries A makes to H, H will return y. Otherwise, it returns what $H(m)$ is supposed to be. Construct B as shown in fig.**??**.

Now look at what B does. Consider oracle H'. The distribution of its output is same as a random oracle. The value y is uniform on $\mathbb{Z}/n\mathbb{Z}$, and independent of everything else that A sees. A is still interacting with a random oracle. A's decision of which of the t messages to forge is independent of j. If A forges with a probability $t\epsilon$, A only helps when it outputs the pair (m, τ), i.e. j-th message, which has probability of $t\epsilon/t = \epsilon$.

In this lemma, we did not take into account the chosen message attack. We will prove the real theorem in which A is given the signing oracle S.

1.2 Theorem

Theorem 1 (security of FDH) *If RSA is (t, ϵ)-trapdoor permutation, then full domain hash FDH-RSA is $(\frac{t}{c}, q, t\epsilon)$-secure signature scheme.*

To prove this theorem, we assume, by the way of contradiction, that there is an adversary A that forges, shown in fig.**??**. Notice that it is different from 1 in that A has access to a additional oracle S.

Now we want to construct an algorithm B that inverts RSA. i.e. given public key n and input y, it output x s.t. $x^3 = y \pmod n$.

The construction used for proving the lemma above does not work here, because in order to run A, we need to simulate oracle S to answer A's queries, but it requires the private key.

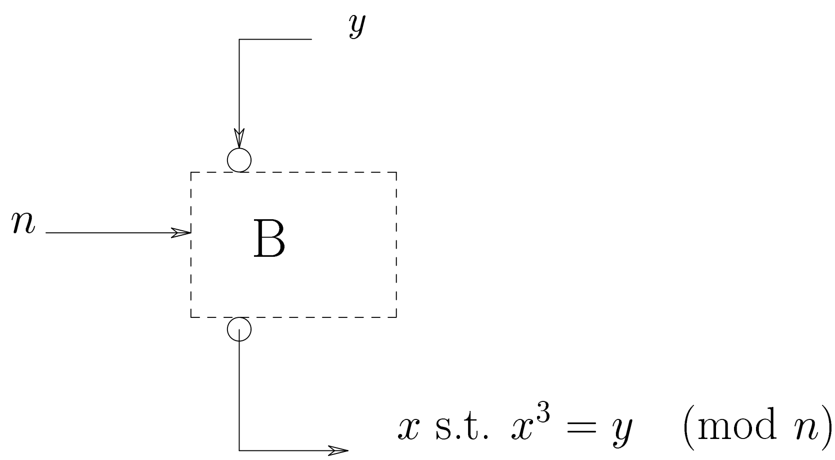

Figure 2: Adversary B

Without loss of generality, we assume that before the adversary output signature of m, it queries it to H'. So the hash of m is already determined.

Let H' return

$$\begin{cases} r_i^3 & \text{if } i \neq j \\ y & \text{if } i = j \end{cases}$$

And Let S' return

$$\begin{cases} r_i & \text{if } i \neq j \\ \text{FAIL} & \text{if } i = j \end{cases}$$

The construction of B is shown in fig.**??**.

H' is still a random oracle, with same distribution as a normal random. S' fails when A queries it with message m_i, so m_j cannot be a message queried to S' before. i.e. outputting the correct tag of m_j is a forgery. That is, τ is cube root of y only if A forges. Probability of inverting is at least $t\epsilon/t = \epsilon$.

This scheme (FDH) is quite easy to implement. However, from a security point of view, this is not quite good. The factor t here is really annoying. Assume that the adversary can have t= 2^64, and we choose RSA with $\epsilon = 2^{-40}$. The probability of the adversary breaking FDH is greater than 1, so the theorem tells you nothing!

[coron 2000] proved that you can get $O(q\epsilon)$, with RSA. Later, [coron 2002] proved that the bound is actually $\Omega(q\epsilon)$. $q\epsilon$ better than $t\epsilon$ because q is chosen by the system designer, while t is chosen by the adversary.

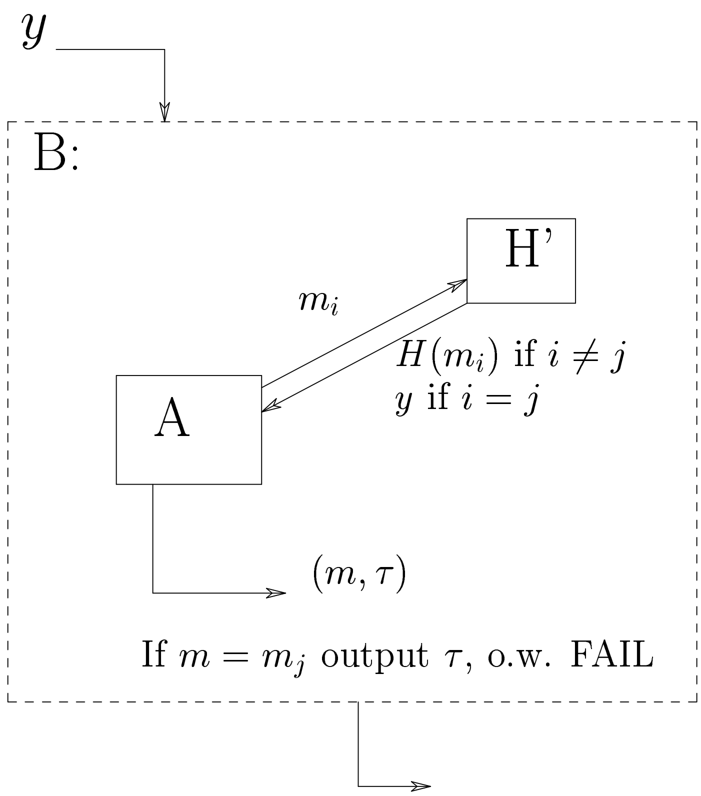

Figure 3: Adversary B

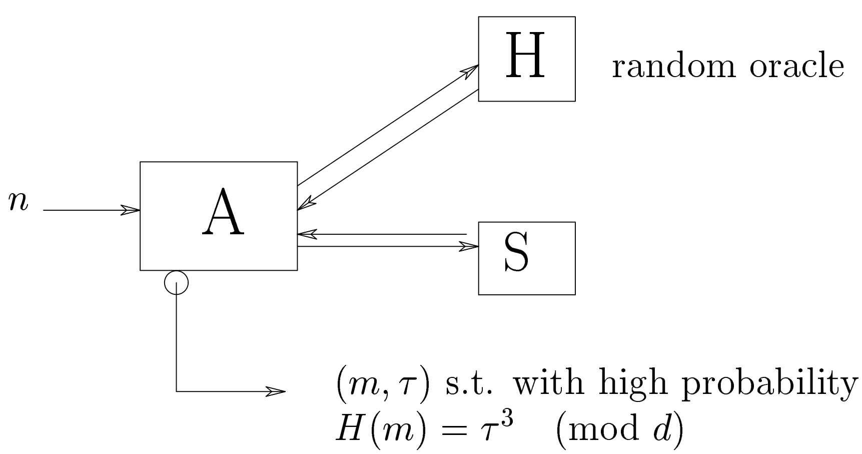

Figure 4: Adversary A

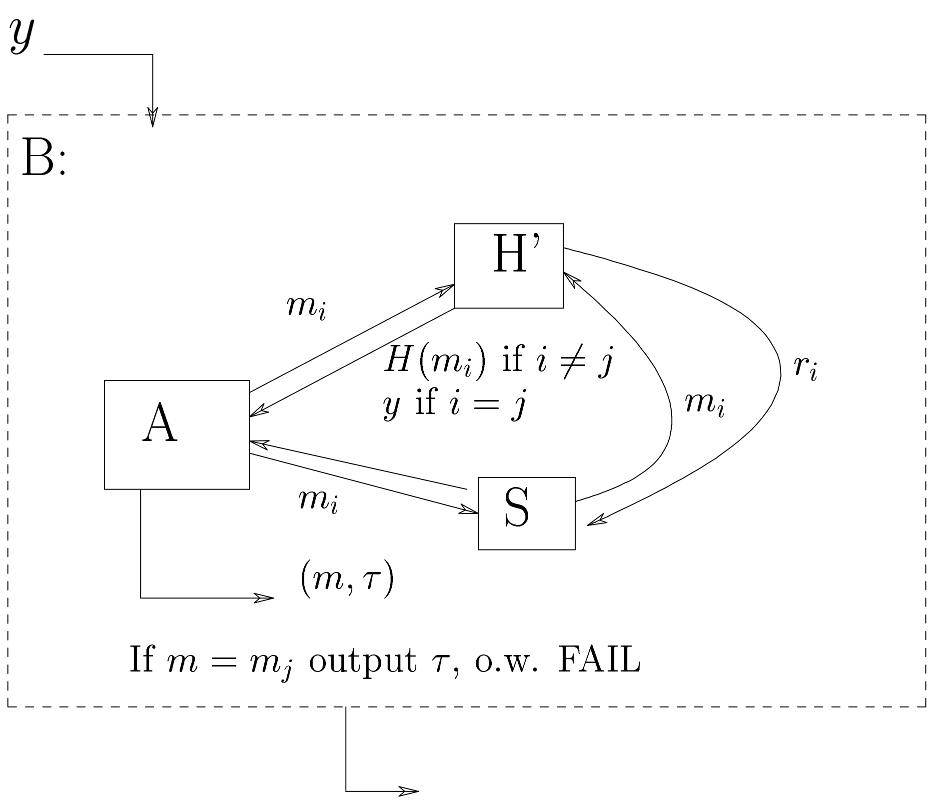

Figure 5: Adversary B