Public Key Encryption Algorithms and the Random Oracle

Debdeep Mukhopadhyay
IIT Kharagpur

Provable Security

- Schemes are efficient but not provably secured:
  - example standard RSA
- Schemes are provably secured are not efficient.
- Exceptions exist, like the El Gamal encryption which is efficient and secured at the same time. We need more constructions which are provably secured, but also efficient.
  - Random Oracles gives one such approach.

What is a Random Oracle?

- Imagine it as a large book of random numbers.
- when you turn to any page (query on any input), a random output is returned.
  - if you subsequently turn into the same page (that is query again on the same point) the same number is returned.
- but if different points are accessed, random numbers are returned.

The RO Methodology

- First, a scheme is proved secured in the ideal world: that is assuming that ROs exist. Standard assumptions are also made.
- Secondly, the RO is replaced by a real hash function, that is if a party is to query a hash for say x, it computes it itself.

Proof Techniques

- If the adversary A, has not queried for some point x, then H(x) is completely random.
- We will construct a reduction, showing that if A is able to break the encryption scheme using the RO, then it can be used to break a standard cryptographic assumption.

Proof Techniques

- The reduction may choose values for the output of H (the RO) and return to A.
  - this is called “programmability”
- The reduction sees all the queries that A makes to the RO

Soundness of the proof system

- Problem arises when we fix the RO by a hash function.
- However, using a scheme that is proved in the RO model is better than no proof.
- However if at the slight cost of efficiency, we have a cryptosystem with a proof in the standard model (like that of the ElGamal encryption), then that is preferred.

A concrete example

- Consider a RSA based scheme,
  - public key: [N,e]
  - secret key: d
  - Plaintext: m$\in\{0,1\}^{l(n)}$
  - **Enc:** <[r^emod N, m ^ H(r)]>

The Construction

If the RSA problem is hard and H is modeled as a RO, the construction has IND secured encryptions under CPA.

Let A be a PPT, and define:

$\varepsilon(n)=\Pr[\mathrm{PubK}^{eav}_{A,\Pi}(n)=1]$

Define $\mathrm{Pub}^{eav}_{A,\Pi}(n)$:

1. A random function H is chosen.
2. Generate <N,e,d>. A is given p_k =<N,e> and may query H(.). Eventually A outputs two messages, $m_0, m_1 \leftarrow \{0,1\}^{l(n)}$
3. A random bit $b \leftarrow \{0,1\}$ and a random $r \leftarrow Z_n^*$ are chosen. A is given the ciphertext, $<[r^e \bmod N, H(r) \oplus m_b]>$. The adversary can still query H(.).
4. Finally, A outputs b'. $\mathrm{Pub}^{eav}_{A,\Pi}$ returns 1, if b=b'. Else 0 is returned.

The proof

- Define Query to be the event that at any point A queries r to the RO (where r is the value used to generate the challenge, c).

$$\therefore \Pr[\text{success}] = \Pr[\text{success} \wedge \overline{\text{Query}}] + \Pr[\text{success} \wedge \text{Query}]$$
$$\leq \Pr[\text{success} | \overline{\text{Query}}] + \Pr[\text{Query}]$$

Claim 1: $\Pr[\text{success} | \overline{\text{Query}}] \leq \frac{1}{2}$

Claim 2: Pr[Query] is negligible

Claim 1 follows from the fact that if A does not query for r, then H(r) is random, and so A has no way to understand whether m_0 or m_1 was encrypted.

Claim 2

- Construct a reduction D, which takes as input $c_1 = r^e \bmod N$ and has to output r (that is break RSA).
- It generates randomly $c_2 \in \{0,1\}^{l(n)}$ and sends to A.
- **A makes some queries to H, r_i. D observes the queries and checks if $r_i^e \bmod N = c_1$. Whenever there is a match, thus RSA is broken. So, the Pr[Query] must be negligible, under the standard RSA assumption.**

$E(x) = T(r) \| G(r) \oplus x$ is RO-IND-CPA for trapdoor T.
Suppose this is not true. That is we have an adversary $A=(A_0, A_1)$ with significant advantage ε.
Remember A_0 is used to generate the plaintexts m_0 and m_1. A_1 is then handed the challenge c, which is the ciphertext corresponding to a randomly chosen message.
Both A_0 and A_1 can make queries to the random oracle G.
Using these algorithms we intend to invert T, the trap-door function without knowing the trap-door.

If A_0 asks a query for r (used to generate the challenge), return r (thus we have inverted the trap-door). Else A_0 terminates, and A_1 starts.

Instead of feeding A_1 the challenge ciphertext, it is asked T(r)||z, where $z=\{0,1\}^{|x|}$, is a random string.
It is checked whether A_1 makes a query at r, by checking if T(r)=y.
Define A_k : Event that A_1 asks a query at r. If it does not then it has no advantage in guessing which plaintext was encrypted.

$$\therefore 1/2 + \varepsilon < \Pr[\text{A succeeds}|A_k]\Pr[A_k] + \Pr[A \text{ succeeds}|\overline{A_k}]\Pr[\overline{A_k}].$$
$$< \Pr[A_k] + 1/2$$
$$\therefore \Pr[A_k] > \varepsilon$$

Thus we can invert the trapdoor T with significant probability, thus we arrive at a contradiction.

E(x)=T(r)||G(r)$\oplus$x||H(rx) is secure against chosen ciphertext attack.
We proof in the same lines. Consider a successful adversary A=(A_0,A_1)
with probability of success > 1/2 + ε. We shall construct an algorithm
N, using A which inverts the trapdoor T without knowing the secret.

In addition to G, now both the algorithms also access $D^{G,H}$, the decryption
oracle.
If a query to G is made such that T(r)=y, then return r, else a random string.
If a query to H is made such that T(r)=y, then return r, else a random string.
If a query is asked at a||w||b to $D^{G,H}$, checks whether there is already
a query at r of G and ru of H st. a=T(r), w=G(r)$\oplus$u, then return u, else "invalid".

Define A_k: Event that A makes an oracle call at G(r) or H(ru)
Define L_k: Event that $D^{G,H}$ is asked query for a||w||b, where
b=$H(T^{-1}(a) \| w \oplus G(T^{-1}(a)),$ but never asks its H oracle on
$T^{-1}(a) \| w \oplus G(T^{-1}(a)).$
$\therefore 1/2 + \varepsilon < \Pr[\text{A succeeds}|L_k]\Pr[L_k] + \Pr[\text{A succeeds}|\neg L_k \wedge A_k]\Pr[\neg L_k \wedge A_k] +$
$\Pr[\text{A succeeds}|\neg L_k \wedge \neg A_k]\Pr[\neg L_k \wedge \neg A_k]$
$1/2 + \varepsilon < \Pr[L_k] + \Pr[A_k] + 1/2$
$\therefore \varepsilon < n2^{-k} + \Pr[A_k]$
$\therefore \Pr[A_k] > \varepsilon - n2^{-k}$.*Contradiction.*