Introduction

Electronic transactions and activities taken place over Internet need to be protected against all kinds of interference, accidental or malicious. The general task of the information technology security is to provide this protection. In order to secure the information transmission, various systems/schemes like cryptographic systems, digital signature schemes etc. are implemented.

Due to drastic and continuous increasing trend in the number of business transactions and financial activities in electronic formats, deploying a system for legally binding electronic contracts becomes essential. Digital signature schemes play a vital role in providing such binding. Again, it is very easy to modify and manipulate electronic documents in various ways. Electronic documents duly signed by digital signature should resist such manipulations.

Beside their design and extra properties, signature schemes are expected to be secure. Formalizing security notions for signature schemes allows one to assess the system's security by providing a security proof, thereby giving a guarantee of trust.

This report documents my presentation in the seminar on notions of security. Using the notions of security, I present the security proof for the short message variant of Gennaro-Halevi-Rabin (GHR) signature scheme. This report is organized as follows: first, a brief introduction on digital signature, reductionist security and notions of security is provided. Then the mathematical assumption on strong RSA problem is discussed, followed by a brief description on the GHR signature scheme. Finally, the reduction of SRSA problem, believed to be intractable, to breaking the GHR signature scheme in the strongest sense is presented.

Digital Signature

Just like a handwritten signature on a document, the digital/electronic signature allows to relate an individual to a specific digital/electronic file. The following are important requirements for digital signatures:

- The signature must be tightly attached to the signed document.
- It should be easy to sign for the legitimate signer, easy to verify the signature for the recipient, and hard to sign for a forger.
- The signer should not be able to deny that he signed the document.
- Sometimes it is important that a signed document can only be used once for its legitimate purpose, not several times (say, a cheque).

In order to obtain most/all of the requirements of digital signature, a signature scheme is required. A signature scheme must have a key generator that produces the pair of keys ($\boldsymbol{sk}$, $\boldsymbol{pk}$), the signature signing function $\boldsymbol{S}$, and the signature verification

function ***V***. Here ***sk*** is the signer's secret key and ***pk*** is the key for verification. In a symmetric system, a same secret key ***k*** is used for signing and verification functions, i.e., ***k*** = ***sk*** = ***pk***. In an asymmetric system, ***sk*** is kept private by the signer and ***pk*** may be publicly published to everybody.

To sign a message ***m***, the signature ***s*** is generated using ***s*** = $S_{sk}(m)$. Here the signature ***s*** is another message. For any signature (i.e., message) ***z***, $V_{pk}(m, z)$ is either *true* or *false*, namely

$$V_{pk}(m,z) = \begin{cases} true & if\ z = V_{pk}(m) \\ false & otherwise \end{cases}$$

A message authentication scheme (or cryptographic transmission scheme), private key or public key, is considered to be a good choice for implementing the signature scheme. The decryption is used as signing function and the encryption is used for verification:

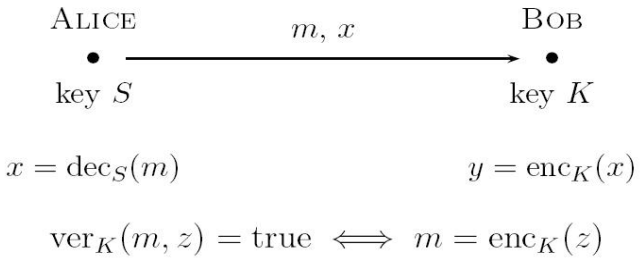

Figure 1: Implementation of a signature scheme using a cryptographic scheme

A public key encryption system like RSA or ElGamal can be used. Then Alice's key ***S*** is her secret key, used for decryption in the communication mode, and Bob verifies Alice's signature using her public key ***K***.

Reductionist Security

Once a cryptographic system or a signature scheme is described/established, how can its security be proved? The common practice in the last decade was trying to exhibit an attack. If there was an attack, the system was insecure. But if there was no such attack, it was quite difficult to certainly label it as secured. It was seen that a system that was considered secure for a long period of time, had broken later.

Nowadays a different approach is used. The idea is to provide security proof that no attack exists under some mathematical assumptions. But the assumption has to be reasonable. If there is any attack found, it indicates that the assumption is wrong.

There is a good number of mathematical problems those have been drawing the attention of many scholars, professionals, analysts and researchers for a long time. Yet there is no probabilistic polynomial time solution for these problems and hence these problems are presumed as hard. The mathematical assumptions based on these problems are more appropriate and carry more sense than just exhibiting attacks. A relationship is established between the mathematical problems and the systems/schemes described by means of reduction.

To prove $\boldsymbol{A} \Leftarrow \boldsymbol{B}$, i.e., problem $\boldsymbol{A}$ is reducible to problem $\boldsymbol{B}$; it is needed to show an algorithm (with polynomial resources) that solves $\boldsymbol{A}$ with access to an oracle that solves $\boldsymbol{B}$.

There is no straightforward and direct formulation of security proof. There are only reductions. The security based on reduction is termed as the provable security or reductionist security.

To get a security proof, one needs to

1. Formally define the security notion to achieve,
2. Make precise mathematical assumptions,
3. Design a system (a signature scheme) and describe its operational modes,
4. Exhibit a reduction from the assumed underlying problem to breaking the scheme in the sense of the defined security notion.

Notions of Security

A security notion (or level) is entirely defined by pairing an adversarial goal with an adversarial model. Depending on the context in which a given signature scheme (or cryptosystem) is used, one may formally define a security notion for the system,

- by telling what goal an adversary would attempt to reach (the adversarial goal), and
- what means or information are made available to the attacker (the adversarial or attack model).

Here some of the adversarial goals as well as adversarial models related to digital signatures are briefly described.

Adversarial Goals

Unbreakability: The attacker recovers the secret key $\boldsymbol{sk}$ from the public key $\boldsymbol{pk}$ (or an equivalent key if any). This goal is denoted UB. It is implicitly appeared with public-key signature scheme (or cryptography).

Universal Unforgeability: The attacker, without necessarily having recovered ***sk***, can produce a valid signature ***s*** of any message ***m*** in the message space. It is noted UUF.

Existential Unforgeability: The attacker creates a message ***m*** and a valid signature ***s*** of it (likely no control over the message). This is denoted EUF.

Adversarial Models

Key-Only Attacks: The adversary only has access to the public key ***pk***. This is denoted KOA. This is an unavoidable scenario in public-key signature scheme (or cryptography).

Known Message Attacks: An adversary has access to signatures for a set of known messages. It is noted KMA.

Chosen Message Attacks: Here the adversary is allowed to use the signer as an oracle (full access), and may request the signature of any message of his choice (multiple requests of the same message are allowed). It is denoted CMA.

Putting the adversarial goal on the y-axis and adversarial model on the x-axis, the security notions are obtained. The intersecting points represent security notions. For example, UB-KOA, UB-KMA, EUF-CMA etc are security notions. If there are u adversarial goals and v adversarial models, there will be $u \times v$ security notions.

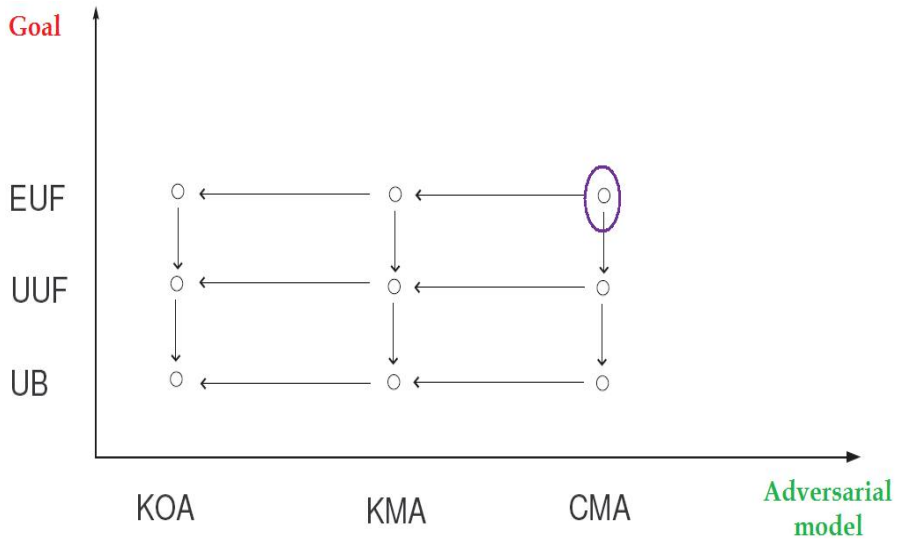

Figure 2: The notions of security for signature scheme

The security notions are explained with an example with RSA signature scheme. In 1978, Rivest-Shamir-Adleman proposed this scheme. In the RSA one-way trapdoor permutation,

- Alice sets $n = p \cdot q$ where p and q are large random primes. She picks a public exponent e co-prime to $\varphi(n) = (p - 1) \cdot (q - 1)$. She computes $d = e^{-1} \bmod \varphi(n)$ and publishes (n, e) as public key. She keeps d as her private key.

- To sign a message $m \in Z_n^*$, Alice compute the signature $s = D(m) = m^d \bmod n$ and sends (m, s) to Bob,

- To verify the pair (m, s), Bob computes $m' = E(s) = s^e \bmod n$ and accepts the signed message if $m' = m$.

Assume Bob picks some random s'' and computes $m'' = E(s'')$. Then (m'', s'') is a valid pair since $s'' = D(m'')$ is a valid signature of m''. However, Bob can generate signatures for messages he does not control. We already know that this capability is known as existential forgery. Existential forgery is a weak form of forgery, and practical applications of digital signatures might live with it. Therefore, RSA signature scheme is existential forgeable even in key-only attacks. Finally, it can be said that RSA signature scheme is not EUF-KOA (Existentially Unforgeable under Key-Only Attacks), and hence not EUF-KMA and not EUF-CMA.

Again, RSA is a morphism. For any two elements $m_1, m_2 \in Z_n^*$,

$$(m_1 \cdot m_2)^d = m_1^d \cdot m_2^d \mod n,$$

meaning that $D(m_1 \cdot m_2) = D(m_1) \cdot D(m_2)$. Now suppose Bob wants Alice's signature for some specific message m.

- He picks a random $m_1 \in Z_n^*$ and computes $m_2 = m / m_1 \bmod n$, i.e. Bob finds m_1, m_2 such that $m_1 \cdot m_2 = m \pmod n$,
- Assume Bob asks Alice to sign m_1 and m_2 and receives $D(m_1)$ and $D(m_2)$,
- Bob computes $s = D(m) = D(m_1) \cdot D(m_2) \bmod n$ on his own and get a valid pair (m, s) for his message m.

This is called a universal forgery under a chosen message attack. It is much worse than existential forgery, because the attacker obtains the signature of any message he wants with probability one, without asking the signer to sign it. Therefore, RSA signature scheme is not UUF-CMA (Universally Unforgeable under Chosen Message Attacks).

From the plot, it can be said that UB-KOA is the weakest/lower security notion. On the other hand, EUF-CMA is the highest/upper. In EUF-CMA, the attacker is provided the maximum information/resources and asked to do the weakest task.

Because EUF-CMA is the upper security level, it is desirable to prove security with respect to this notion. Formally, an signature scheme is said to be (q, τ, ε)-secure if for any adversary A with running time upper-bounded by τ,

$$\mathrm{Succ}^{\text{EUF-CMA}}(A) = \Pr\left[(sk, pk) \leftarrow G(1^k), (m^*, s^*) \leftarrow A^{S(sk,\cdot)}(pk), V(pk, m^*, s^*) = 1\right] < \varepsilon,$$

where the probability is taken over all random choices. Here, τ means operations (more specifically the running time) of the scheme and ε means the success rate of the scheme.

The notation $A^{S(sk,\cdot)}$ means that the adversary has access to a signing oracle throughout the game, but at most q times. The message m^* output by A was never requested to the signing oracle.

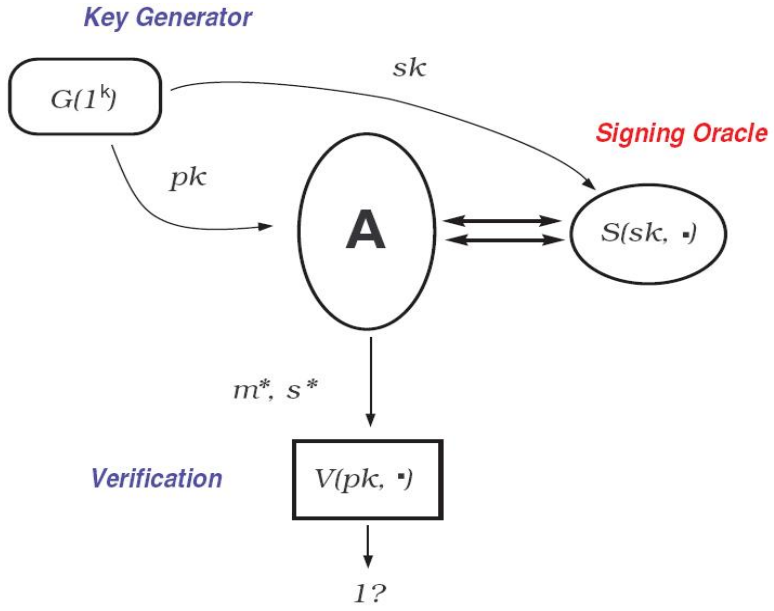

Figure 3: The EUF-CMA game playing environment

Mathematical Assumptions

Public-key design allows constructing systems by assembling and connecting smaller cryptographic or atomic primitives together. For example: one-way functions, hash functions, arithmetic operations etc. Cryptographic primitives are connected to plenty of (supposedly) intractable problems:

- Strong RSA (SRSA) is hard,
- Discrete log is hard,
- Diffie-Hellman is hard,
- Factoring is hard,

Here the term 'Hard' means that there is no probabilistic polynomial time algorithm can solve the problem with non-negligible probability. The problem, which is associated with a probabilistic polynomial time algorithm that solves the problem, is not termed as 'Hard'.

Strong RSA Problem

Let $n = p \cdot q$ be a safe RSA modulus and $z \in Z^*_n$. Find x and e such that

$$z = x^e \bmod n \qquad \text{with } (x, e) \neq (z, 1).$$

An algorithm R is said to (τ_R, ε_R)-solve the SRSA problem if in at most τ_R operations,

$$\Pr\left[n \leftarrow RSA(1^k),\ z \leftarrow Z_n^*,\ (x,e) \leftarrow R(n,z),\ z = x^e \bmod n\right] \geq \varepsilon_R$$

where the probability is taken over R's random tapes and the distribution of (n, z).

Strong RSA Assumption

The mathematical assumption based on SRSA problem (i.e., Strong RSA assumption) states that if an algorithm R is polynomially time bounded, the success of R is negligible. If k is the security parameter of the safe RSA modulus, then the SRSA assumption can be expressed for any (τ_R, ε_R)-solver as

$$\tau_R \leq \text{poly}(k) \quad \Rightarrow \quad \varepsilon_R = \text{negl}(k)$$

It was introduced by Barić and Pfitzmann in 1999 as 'Strong RSA Conjecture', which states that:

Given a randomly chosen RSA modulus n, and a random element $z \in Z^*_n$, it is infeasible to find a pair (e, x) with $e > 1$ such that $x^e = z \pmod n$.

GHR Signature Scheme

Gennaro-Halevi-Rabin (GHR) signature scheme is based on the hash-and-sign paradigm. It was presented in 1999 by Rosario Gennaro, Shai Halevi and Tal Rabin. In hash-and-sign schemes, the message to be signed is hashed using a 'cryptographic hash function' and the result is signed using a 'standard signature scheme', such as RSA. The GHR signature scheme resembles the standard RSA signature algorithm.

In this seminar, only the short message variant of the GHR signature scheme is considered, where the maximum message length is 30 bits.

The GHR scheme works as follows:

[1]. Generate a safe RSA modulus $n = p \cdot q$ with $p = 2p' + 1$, $q = 2q' + 1$, where p, q, p' and q' are distinct prime numbers. Randomly select $z \in Z^*_n$.

Let H: $\{0, 1\}^l \mapsto Primes \geq 3$ and $\neq p', q'$ be a collision-free hash function ($l = 30$). Publish (n, z) as public key. Keep (p, q) as private key.

[2]. To sign a message $m \in \{0, 1\}^l$, compute $s = z^{1/H(m)} \bmod n$.

[3]. Given (m, s), check whether $s^{H(m)} = z \bmod n$.

Reduction Proof

The reduction, which is used for the security proof is SRSA $\leq$ EUF-CMA(GHR).

It means under the Strong RSA assumption, the GHR signature scheme is Existentially Unforgeable to a Chosen Message Attack in the standard model.

In order to proof that, SRSA $\leq$ EUF-CMA(GHR), it is needed to show that breaking EUF-CMA(GHR) allows to solve SRSA, i.e., that an adversary breaking GHR can be used as a black box tool to answer SRSA requests with non-negligible probability.

The reduction R will behave as follows.

- ✓ R is given $n \leftarrow \text{RSA}(1^k)$ and $z \in Z^*_n$, as well as an attacker A that $(q, \tau_A, \varepsilon_A)$-solves EUF-CMA(GHR),
- ✓ R simulates G and transmits pk to A,
- ✓ R receives signature queries from A: R will have to simulate a signing oracle with respect to pk at most q times,
- ✓ A outputs a forgery $(m^\wedge, s^\wedge)$ for GHR with probability ε_A,
- ✓ R outputs non-trivial (x, e) such that $z = x^e \bmod n$.
- ✓ R will provide a perfect simulation and (τ_R, ε_R)-solve SRSA with

$$\varepsilon_R \geq \frac{\varepsilon_A}{2^l} \text{ and } \tau_R \leq \tau_A + poly\left(2^l, k\right)$$

Simulation of Oracles

The reduction has to simulate the attacker's environment in a way that preserves (or does not alter too much) the distribution of all random variables which interact with it.

First the perfect simulation of key generation oracle ***G***, signing oracle ***S*** and verification oracle ***V*** is provided according to the working principles.

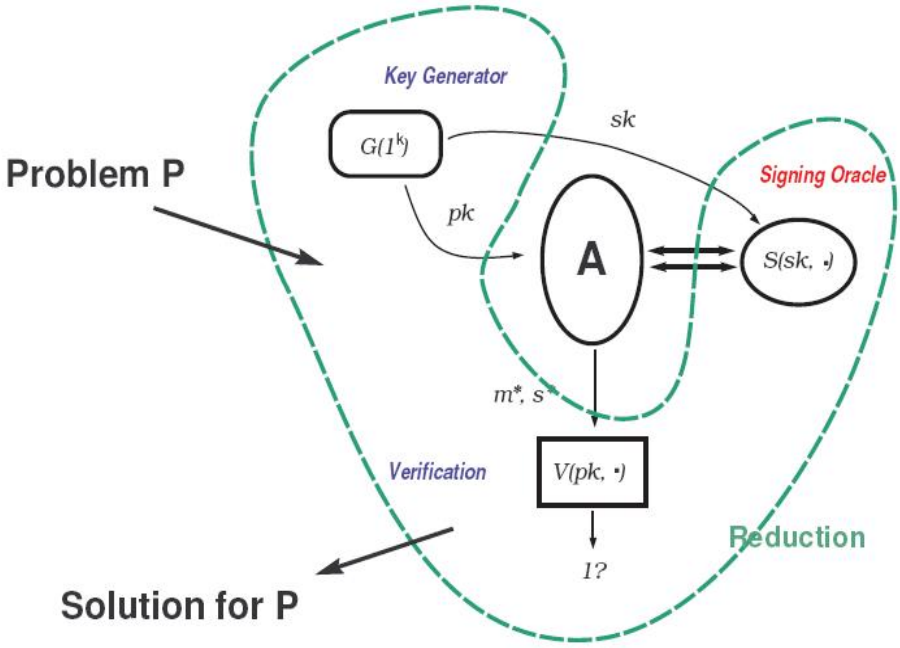

Figure 4: Simulating the attacker's environment

Simulation of G

For each and every message $m_i \in \{0, 1\}^l$, compute $H(m_i)$. Set $E = \prod H(m_i)$.

Compute $y = z^E \bmod n$ and send the GHR public key (n, y) to A.

Since $n \leftarrow \text{RSA}(1^k)$ (external to R) and $z \in Z^*_n$ (external to R) are random choices, and $z \mapsto z^E$ is one-to-one {as E and $\varphi(n)$ are co-prime, $f(z) = z^E \bmod n$ is a bijection}, (n, y) is perfectly indistinguishable from a random GHR public key ($n \leftarrow \text{RSA}(1^k)$, $y \in Z^*_n$).

Therefore, the simulation of G is perfect.

Simulation of S

When A requests the signature of a message m_i, send $s_i = z^{E/H(m_i)} \bmod n$.

Knowing z and E, it is easy to extract a $H(m_i)$-th root of y for any m_i. A's queries can be answered with perfectly valid signatures.

Therefore, the simulation of S is perfect.

Simulation of V

The signature s_i is verified using $s_i^{H(m_i)} = z^E \bmod n$. The simulation of V is trivial.

Therefore, the simulation of the attacker's environment is perfect:

$$\Pr[A \text{ forges}] \geq \varepsilon_A$$

Now, the forgery output by A with probability ε_A will be $(m^\wedge, s^\wedge)$ where $m^\wedge$ is from the given message space and $s^\wedge = z^{E/H(m^\wedge)} \bmod n$. But it is mentioned earlier that with known z and E, R could have computed the forgery. Besides, the forgery must help R to get good solution for (x, e). As the forgery is not new and provides no clue to the solution for (x, e), it is not possible for R to come up with positive outcome. Hence, this is not a forgery and also not exploitable.

Again, an alternative simulation of key generation oracle G, signing oracle S and verification oracle V is provided with a slight change in the working principles.

Simulation of G

Choose $i \in \{1, 2, \ldots \ldots, 2^l\}$ uniformly at random.

For each message $m_j \in \{0, 1\}^l$, compute $\mathbf{H}(m_j)$. Set $E = \prod_{j \neq i} H(m_j)$.

Compute $y = z^E \bmod n$ and send the GHR public key (n, y) to A.

Since $n \leftarrow \text{RSA}(1^k)$ (external to R) and $z \in Z^*_n$ (external to R) are random choices, and $z \mapsto z^E$ is one-to-one {as E and $\varphi(n)$ are co-prime, $f(z) = z^E \bmod n$ is a bijection}, (n, y) is perfectly indistinguishable from a random GHR public key $(n \leftarrow \text{RSA}(1^k), y \in Z^*_n)$.

Therefore, the simulation of G is still a perfect one.

Simulation of S

When A requests the signature of a message m_j:

- If $j \neq i$, send $s_i = z^{E/\mathbf{H}(m_j)} \bmod n$.
- If $j = i$, abort the simulation experiment

A's queries can be answered with perfectly valid signatures except when the query message is m_i.

Since i is chosen in $[1, 2^l]$ independently from the attacker's view,

$$\Pr[\text{perfect simulation}] = \Pr[m_i \notin \text{Queries}(A)] \geq \frac{2^l - q}{2^l} = 1 - \frac{q}{2^l}$$

Simulation of V

The signature s_i is verified using $s_i^{H(m_i)} = z^E \bmod n$. The simulation of V is the same and again trivial.

Now, assume that at the end of the game, A outputs a forgery ($m^\wedge$, $s^\wedge$) with $m^\wedge = m_i$ and $s^\wedge = s$. Then

$$s^{H(m_i)} = y = z^E \bmod n$$

As $H(m_i)$ and E are co-prime, the Bézout theorem says there must be a and b such that $a \cdot H(m_i) + b \cdot E = 1$. Using the Extended Euclidian Algorithm, the values of a and b can easily be computed. Now,

$$z = z^1 = z^{a \cdot H(m_i) + b \cdot E} = z^{a \cdot H(m_i)} \cdot z^{b \cdot E} = \left(z^a\right)^{H(m_i)} \cdot \left(z^E\right)^b = \left(z^a\right)^{H(m_i)} \cdot \left(s^{H(m_i)}\right)^b = \left(z^a \cdot s^b\right)^{H(m_i)}$$

Finally, R sets $x = z^a \cdot s^b$ and $e = H(m_i)$ and outputs a genuine solution (x, e).

Analysis

From the two different simulations, it is clear that when the simulation is perfect, A can never produce a valid forgery which will eventually be used by R for obtaining the solution pair (x, e). On the other hand, in case of the alternative simulation, the probability of a successful forgery depends on number of conditions (i.e., lucks). These include:

- A will not query the message m_i which is chosen at random during the simulation of G. If A queries message m_i, the system will abort and A is not expected to provide a forgery.
- The message in A's forgery ($m^\wedge$, $s^\wedge$) must be m_i i.e., $m^\wedge = m_i$.

Therefore, it can be stated that with the accurate implementation of GHR scheme, it is not possible for R to obtain the solution even with the help of A. A is assumed to be able to break the GHR scheme, but actually A is not capable of doing so. It is vivid that if A could break the scheme easily, solving the SRSA problem by R would be easy.

On the other hand, if we analyze the attack model available to A, we can verify that A has the GHR public key (n, y) and at most q message-signature pair (m_w, s_w), where $1 \leq w \leq q$.

Now, if A wants to put forward a forgery for a message $m^{\wedge}$; a signature can be obtained by A applying the formula $s^{\wedge} = z^{E/H(m^{\wedge})} \bmod n$. For doing that, A must need the value of z and E.

With the available information, i.e., values of (n, y); A can only get the values of z and E solving the equation $y = z^E \bmod n$, which is nothing but the SRSA problem (assuming that A has the specific hash function too). From the mathematical assumption as well as the simulations; it is clear that, solving SRSA is hard (even with the help of A that is capable of forging the scheme). Therefore, it can be said that since the SRSA problem is assumed hard, producing forgery on the scheme is also hard.

Finally, we have got,

(a) If A could forge on the scheme under chosen message attacks easily, solving the SRSA problem by R would be easy.
(b) The SRSA problem is presumed hard, producing forgery on the GHR signature scheme is also hard.

Hence, it is proved that SRSA $\leq$ EUF-CMA(GHR).

Conclusion

According to the requirements of getting a security proof, we have

- ✓ defined a security notions for signature scheme,
- ✓ made precise mathematical assumption (SRSA is hard),
- ✓ described the operation modes of GHR signature scheme and
- ✓ finally performed a reduction from the underlying problem of the mathematical assumption (SRSA problem) to existentially forging of the GHR signature scheme under chosen message attacks.

Therefore, it is evident that GHR signature scheme is secure.