Report on Present State of ESIGN Signature Evaluation (full evaluation)

January 28, 2002

Yasuyuki Sakai, Member

Public-Key Cryptography Subcommittee

ESIGN signature

(One of the signature schemes included in the guidelines for electronic signature law)

- **Category: Signature**
- **Security basis:**

  It is difficult to solve $n=p^2q$ type factoring problems;

  AER (approximate e-th root) problems are difficult
- **Provable security:**

  ESIGN, which is included in the guidelines for electronic signature law, uses the assumption that AER (approximate e-th root) problems are difficult. At the present time there is no proof with this assumption that existential forgeries are infeasible under adaptive chosen message attacks, even if a random oracle hash function is used.
- **Implementability characteristics :**

  ESIGN's signature generation is faster than that of RSA signatures.
- **SW implementation information :**

  Key generation: 610ms, signature generation: 1.04ms, signature verification: 0.70ms

  (|n|=1152, e=1024, SHA-1 used, Celeron 800MHz, included in self-evaluation)

ESIGN signature versions

ESIGN has been submitted or presented on seven occasions (the main ones are listed below).

<table>
<tr><th></th><th>Recommended parameter</th><th>Encode</th><th>Provable security</th></tr>
<tr><td>Guidelines for electronic signature law</td><td>$|n| \geq 1024$, $e \geq 8$</td><td rowspan="2">EMSA</td><td rowspan="2">None (now evaluating to determine whether there is an efficient attack method)</td></tr>
<tr><td>CRYPTREC2001</td><td>$|n| \geq 1152$, $e \geq 1024$</td></tr>
<tr><td>CRYPTREC2000</td><td>$|n| \geq 960$, $e \geq 8$</td><td colspan="2">No specifications (primitive proposal only)</td></tr>
<tr><td>IEEE P1363a</td><td>No specifications (P1363 policy)</td><td rowspan="2">EMSA5</td><td rowspan="2">Yes ($n=p^2q$ type factoring assumption, approximate e-th root assumption, random oracle model, existential unforgeability under adaptive chosen message attacks)</td></tr>
<tr><td>NESSIE
(changes planned as shown on right)</td><td>$|n| \geq 1152$, $e \geq 1024$</td></tr>
</table>

Full evaluation policy

- ESIGN is being evaluated from the following perspectives
  - Primitive evaluation:

    Difficulty of AER (approximate e-th root) problems
  - Scheme evaluation:

    Recommended parameters in guidelines for electronic signature law :

    $$\text{Security of } |n| \geq 1024,\ e \geq 8$$

    Security of recommended parameters at other standardizing organizations
  - Other issues

Evaluation comments: AER (approximate e-th root) problems

- If e=2:

  AER problems can be solved using the method of Brickell et al [Crypto95] or the method of Vallee et al [Eurocrypt88]

  (use LLL algorithm to solve for lower-degree modular polynomials), as well as Coppersmith's method [Eurocrypt96], which is a refinement of the above.
- If e=3:

  The above methods can be extended to apply to cases where e=3.
- If $e \geq 4$:

  The claim that there are no known efficient solutions to AER problems is satisfactory.

Evaluation comments: Encoding (1/2)

ESIGN using EMSA Encoding (no provable security)
(This ESIGN version is included in the guidelines for electronic signature law)

- An external evaluator claims to have discovered a new attack method (forgery; the legitimacy of this claim has not been verified). If the evaluator's claim is correct and the attack is used when the hash function output is 160 bits (SHA-1), then forgeries would be successful with a non-negligible probability in cases such as the following.

  (1) |n|=1024 and e=4

  (2) |n|=2048 and e=7

  (3) |n|=2048 and e=8

- The new attack discovered by this evaluator does not threaten security when e=1024 (according to the evaluator).

Evaluation comments: Encoding (2/2)

ESIGN using EMSA5 Encoding (provable security)
(IEEE P1363a, NESSIE)

Under the assumption that AER (approximate e-th root) problems are difficult, satisfactory proof is offered that existential forgeries are infeasible (security in the strongest sense) under adaptive chosen message attacks when a random oracle is used.