

# **RSA Digital Signature Standards**

**Burt Kaliski, RSA Laboratories  
RSA Conference 2000**

# Outline

**I. Background**

**II. Forgery and provable security**

**III. Example signature schemes**

**IV. Standards strategy**

# Part I: Background

# General Model

- **A signature scheme consists of three (or more) related operations:**
  - *key pair generation* produces a public/private key pair
  - *signature operation* produces a signature for a message with a private key
  - *verification operation* checks a signature with a public key
- **In a scheme with *message recovery*, verification operation recovers message from signature**
- **In a scheme with *appendix*, both message and signature must be transmitted**

# Trapdoor One-Way Functions

- A *one-way function*  $f(x)$  is easy to compute but hard to invert:
  - easy:  $x \rightarrow f(x)$
  - hard:  $f(x) \rightarrow x$
- A *trapdoor one-way function* has trapdoor information  $f^{-1}$  that makes it easy to invert:
  - easy:  $f(x), f^{-1} \rightarrow x = f^{-1}(f(x))$
- Many but not all signature schemes are based on trapdoor OWFs

# RSA Trapdoor OWF

- The RSA function is

$$f(x) = x^e \bmod n$$

where  $n = pq$ ,  $p$  and  $q$  are large random primes, and  $e$  is relatively prime to  $p-1$  and  $q-1$

- This function is conjectured to be a trapdoor OWF
- Trapdoor is

$$f^{-1}(x) = x^d \bmod n$$

where  $d = e^{-1} \bmod \text{lcm}(p-1, q-1)$

# Signatures with a Trapdoor OWF

- **Signature operation:**

$$s = \sigma(M) = f^{-1}(\mu(M))$$

– where  $\mu$  maps from message strings to  $f^{-1}$  inputs

- may be randomized
  - invertible for signatures with message recovery
- **Verification operation (with appendix):**

$$f(s) =? \mu(M)$$

- if randomized,  $f(s) \in? \mu(M)$

- **Verification operation (with message recovery):**

$$M = \mu^{-1}(f(s))$$

# Mapping Properties

- Mapping should have similar properties to a hash function:
  - one-way: for random  $m$ , hard to find  $M$  s.t.  $\mu(M) = m$
  - collision-resistant: hard to find  $M_1, M_2$  s.t.  $\mu(M_1) = \mu(M_2)$
- For message recovery, a “redundancy” function
- May also identify underlying algorithms
  - e.g., algorithm ID for underlying hash function
- Should also interact well with trapdoor function
  - ideally, mapping should appear “random”



# Multiplicative Properties of RSA

- **RSA function is a *multiplicative homomorphism*:**  
for all  $x, y$ ,

$$f(xy \bmod n) = f(x) f(y) \bmod n$$

$$f^{-1}(xy \bmod n) = f^{-1}(x) f^{-1}(y) \bmod n$$

- **More generally:**

$$f^{-1}\left(\prod x_i \bmod n\right) = \prod (f^{-1}(x_i)) \bmod n$$

- **Property is exploited in most forgery attacks on RSA signatures, but also enhances recent security proofs**

# Part II: Forgery and Provable Security

# Signature Forgery

- A *forgery* is a signature computed without the signer's private key
- Forgery attacks may involve interaction with the signer: a *chosen-message* attack
- Forgery may produce a signature for a specified message, or the message may be output with its signature (*existential forgery*)

# Multiplicative Forgery

- Based on the multiplicative properties of the RSA function, if

$$\mu(M) = \prod \mu(M_j)^{\alpha_j} \text{ mod } n$$

then

$$\sigma(M) = \prod \sigma(M_j)^{\alpha_j} \text{ mod } n$$

- Signature for  $M$  can thus be forged given the signatures for  $M_1, \dots, M_l$ , under a chosen-message attack

# Small Primes Method

- **Suppose  $\mu(M)$  and  $\mu(M_1), \dots, \mu(M_l)$  can be factored into small primes**
  - **Desmedt-Odlyzko (1986); Rivest (1991 in PKCS #1)**
- **Then the exponents  $\alpha_i$  can be determined by relationships among the prime factorizations**
- **Requires many messages if  $\mu$  maps to large integers, but effective if  $\mu$  maps to small integers**
- **Limited applicability to example schemes**

# Recent Generalization

- Consider  $\mu(M), \mu(M_1), \dots, \mu(M_l) \bmod n$ , and also allow a fixed factor
  - Coron-Naccache-Stern (1999)
- Effective if  $\mu$  maps to small integers mod  $n$  times a fixed factor
- Broader applicability to example schemes:
  - ISO 9796-2 [CNS99]
  - ISO 9796-1 [Coppersmith-Halevi-Jutla (1999)]
  - recovery of private key for Rabin-Williams variants [Joye-Quisquater (1999)]

# Integer Relations Method

- What if the equation

$$\mu(M) = f(t) \prod \mu(M_i)^{\alpha_i}$$

could be solved without factoring?

- Effective for weak  $\mu$ :
  - ISO 9796-1 with *three* chosen messages [Grieu (1999)]

# Reduction Proofs

- A *reduction proof* shows that inverting the function  $f$  “reduces” to signature forgery: given a forgery algorithm  $F$ , one can construct an inversion algorithm  $I$
- “Provable security”:
  - inversion hard  $\rightarrow$  forgery hard
- “Tight” proof closely relates hardness of problems



# Random Oracle Model

- In the *random oracle* model, certain functions are considered “black boxes”: forgery algorithm cannot look inside
  - e.g., hash functions
- Model enables reduction proofs for generic forgery algorithms — inversion algorithm embeds input to be inverted in oracle outputs
- Multiplicative property can enhance the proof

# Part III: Example Signature Schemes

# Overview

- **Several popular approaches to RSA signatures**
- **Approaches differ primarily in the mapping  $\mu$**
- **Some differences also in key generation**
- **Some also support Rabin-Williams (even exponent) signatures**
  
- **There are many other signature schemes based on factoring (e.g., Fiat-Shamir, GQ, Micali, GQ2); focus here is on those involving the RSA function**

# Schemes with Appendix

- **Basic scheme**
- **ANSI X9.31**
- **PKCS #1 v1.5**
- **Bellare-Rogaway FDH**
- **Bellare-Rogaway PSS**

# Basic Scheme

- $\mu(M) = \text{Hash}(M)$
- Pedagogical design
- Insecure against multiplicative forgery for typical hash sizes
- (Hopefully) not widely deployed

# ANSI X9.31

(Digital Signatures Using Reversible Public-Key  
Cryptography for the Financial Services Industry, 1998)

- $\mu(M) = 6b \text{ } bb \dots bb \text{ } ba \parallel \text{Hash}(M) \parallel 3x \text{ } cc$   
where  $x = 3$  for SHA-1, 1 for RIPEMD-160
- Ad hoc design
- Resistant to multiplicative forgery
  - some moduli are more at risk, but still out of range
- Widely standardized
  - IEEE P1363, ISO/IEC 14888-3
  - US NIST FIPS 186-1
- ANSI X9.31 requires “strong primes”

# PKCS #1 v1.5

(RSA Encryption Standard, 1991)

- $\mu(M) = 00\ 01\ ff\ \dots\ ff\ 00 \parallel \text{HashAlgID} \parallel \text{Hash}(M)$
- Ad hoc design
- Resistant to multiplicative forgery
  - moduli near  $2^k$  are more at risk, but still out of range
- Widely deployed
  - SSL certificates
  - S/MIME
- To be included in IEEE P1363a; PKCS #1 v2.0 continues to support it

# ANSI X9.31 vs. PKCS #1 v1.5

- **Both are deterministic**
- **Both include a hash function identifier**
- **Both are ad hoc designs**
  - both resist [CNS99]/[CHJ99] attacks
- **Both support RSA and RW primitives**
  - see IEEE P1363a contribution on PKCS #1 signatures for discussion
- **No patents have been reported to IEEE P1363 or ANSI X9.31 for these mappings**



# Bellare-Rogaway FDH

(Full Domain Hashing, ACM CCCS '93)

- $\mu(M) = 00 \parallel \text{Full-Length-Hash}(m)$
- Provably secure design
- To be included in IEEE P1363a

# Bellare-Rogaway PSS

(Probabilistic Signature Scheme, Eurocrypt '96)

- $\mu(M) = 00 \parallel H \parallel G(H) \oplus [\textit{salt} \parallel 00 \dots 00]$

where  $H = \text{Hash}(\textit{salt}, M)$ ,  $\textit{salt}$  is random, and  $G$  is a mask generation function

- Provably secure design
- To be included in IEEE P1363a; ANSI X9.31 to be revised to include it

*Note:* The format above is as specified in PKCS #1 v2.1 d1, and is subject to change.

# FDH vs. PSS

- **FDH is deterministic, PSS is probabilistic**
- **Both provably secure**
  - same paradigm as **Optimal Asymmetric Encryption Padding (OAEP)**
- **PSS has tighter security proof, is less dependent on security of hash function**
- **PSS-R variant supports message recovery, partial message recovery**
- **PSS is patent pending (but generously licensed)**

# Schemes with Message Recovery

- **Basic scheme**
- **ISO/IEC 9796-1**
- **ISO/IEC 9796-2**
- **Bellare-Rogaway PSS-R**

# Basic Scheme

- $\mu(M) = M$
- Another pedagogical design (“textbook RSA”)
- Insecure against various forgeries, including existential forgery ( $M = f(\sigma)$ )
- Again, hopefully not widely deployed

# ISO/IEC 9796-1

(Digital Signature Scheme Giving Message Recovery, 1991)

- $$\begin{aligned} \mu(M) = & s^*(m_{l-1}) s'(m_{l-2}) m_{l-1} m_{l-2} \\ & s(m_{l-3}) s(m_{l-4}) m_{l-3} m_{l-4} \dots \\ & s(m_3) s(m_2) m_3 m_2 \\ & s(m_1) s(m_0) m_0 6 \end{aligned}$$

where  $m_i$  is the  $i$ th nibble of  $M$  and  $s^*$ ,  $s'$  and  $s$  are fixed permutations

- Ad hoc design with significant rationale
- *Not* resistant to multiplicative forgery [CHJ99], [Grieu 1999]
  - may still be appropriate if applied to a hash value

# ISO/IEC 9796-2

(Digital Signature Scheme Giving Message Recovery —  
Mechanisms Using a Hash Function, 1997)

- $\mu(M) = 4b\ bb\ bb\ \dots\ bb\ ba \parallel M \parallel \text{Hash}(M) \parallel bc$   
or  $6a \parallel M' \parallel \text{Hash}(M) \parallel bc$

where  $M'$  is part of the message

- this assumes modulus length is multiple of 8
- general format allows hash algorithm ID
- **Ad hoc design**
  - hash provides some structure
- **Not resistant to multiplicative forgery if hash value is 64 bits or less [CNS99]**
  - may still be appropriate for larger hash values

# Bellare-Rogaway PSS-R

(Probabilistic Signature Scheme with Recovery, 1996)

- $\mu(M) = 00 \parallel H \parallel G(H) \oplus [\textit{salt} \parallel 00 \dots 01 \parallel M]$

where  $H = \text{Hash}(\textit{salt}, M)$ ,  $\textit{salt}$  is random, and  $G$  is a mask generation function

- Provably secure design
- To be included in IEEE P1363a; ISO/IEC 9796-2 to be revised to include it

*Note:* The format above is as specified in IEEE P1363a D1, and is subject to change.



# Part IV: Standards Strategy

# Standards vs. Theory vs. Practice

- **ANSI X9.31 is widely standardized**
- **PSS is widely considered secure**
- **PKCS #1 v1.5 is widely deployed**
  
- **How to harmonize?**
- **(Related question for signature schemes with message recovery)**

# Challenges

- **Infrastructure changes take time**
  - particularly on the user side
- **ANSI X9.31 is more than just another encoding method, also specifies “strong primes”**
  - a controversial topic
- **Many communities involved**
  - formal standards bodies, IETF, browser vendors, certificate authorities

# Prudent Security

- **What if a weakness were found in ANSI X9.31 or PKCS #1 v1.5 signatures?**
  - no proof of security, though designs are well motivated, supported by analysis
  - would be surprising — but so were vulnerabilities in ISO/IEC 9796-1,-2
- **PSS embodies “best practices,” prudent to improve over time**

# Proposed Strategy

- **Short term (1-2 years): Support both PKCS #1 v1.5 and ANSI X9.31 signatures for interoperability**
  - e.g., in IETF profiles, FIPS validation
    - NIST intends to allow PKCS #1 v1.5 in FIPS 186-2 for an 18-month transition period
- **Long term (2-5 years): Move toward PSS**
  - not necessarily, but perhaps optionally with “strong primes”
  - upgrade in due course — e.g., with AES algorithm, new hash functions

# Standards Work

- **IEEE P1363a will include PSS, PSS-R**
  - also FDH, PKCS #1 v1.5 signatures
- **PKCS #1 v2.1 d1 includes it**
- **ANSI X9.31 will be revised to include PSS**
- **ISO/IEC 9796-2 will be revised to include PSS-R**
- **Coordination is underway**

# Conclusions

- **Several signature schemes based on RSA algorithm**
  - **varying attributes: standards, theory, practice**
- **Recent forgery results on certain schemes, security proofs on others**
- **PSS a prudent choice for long-term security, harmonization of standards**