The RSA Trapdoor Permutation

Dan Boneh
Stanford University

Review: arithmetic mod composites

- Let $N = p \cdot q$ where p,q are prime
- Notation: $Z_N = \{0,1,2,\ldots,N-1\}$

  $(Z_N)^* = \{\text{invertible elements in } Z_N\}$
- Facts:
  - $x \in Z_N$ is in $(Z_N)^*$ $\Leftrightarrow$ $\gcd(x,N) = 1$
  - Number of elements in $(Z_N)^*$ is $\varphi(N) = (p-1)(q-1)$
- Euler's thm: $\forall\, x \in (Z_N)^* \;:\; x^{\varphi(N)} = 1$

Review: trapdoor permutations

Three algorithms: (G, F, F^{-1})

- G: outputs pk, sk

  pk defines a function $F(pk, \cdot): X \to X$

- F(pk, x): evaluates the function at x

- $F^{-1}(sk, y)$: inverts the function at y using sk

Secure trapdoor permutation (review):

the func. $F(pk, \cdot)$ is one-way without the trapdoor sk.

The RSA trapdoor permutation

- First published:
  - Scientific American, Aug. 1977.

    (after some censorship entanglements)

- Currently the "work horse" of Internet security:
  - Most Public Key Infrastructure (PKI) products.
  - SSL/TLS: Certificates and key-exchange.
  - Secure e-mail and file systems.

The RSA trapdoor permutation

- alg G: $N=pq.$ $\begin{cases} N \approx 1024 \text{ bits.} \quad p,q \approx 512 \text{ bits.} \\ e - \text{encryption exponent.} \quad \gcd(e, \varphi(N)) = 1 \,. \end{cases}$

- alg F: $\mathbf{RSA(M) = M^e} \in Z_N^*$ where $M \in Z_N^*$

- Trapdoor: d – decryption exponent.

  Where $e \cdot d = 1 \pmod{\varphi(N)}$

- alg F^{-1}: $RSA(M)^d = M^{ed} = M^{k\varphi(N)+1} = (M^{\varphi(N)})^k \cdot M = M$

- (n,e,t,ε)-RSA Assumption: For all t-time algs. A:

$$\Pr\left[A(N,e,x) = x^{1/e} \ (N) \ : \ \begin{matrix} p,q \xleftarrow{R} \text{n-bit primes,} \\ N \leftarrow pq, \quad x \xleftarrow{R} Z_N^* \end{matrix} \right] < \varepsilon$$

Textbook RSA is insecure

- Textbook RSA encryption:
  - public key: **(N,e)** Encrypt: $C = M^e \pmod{N}$
  - private key: **d** Decrypt: $C^d = M \pmod{N}$

    $(M \in Z_N^*)$

- Completely insecure cryptosystem:
  - Does not satisfy basic definitions of security.
  - Many attacks exist.

- The RSA trapdoor permutation is not a cryptosystem !

A simple attack on textbook RSA

- Session-key K is 64 bits. View $K \in \{0,\dots,2^{64}\}$
  Eavesdropper sees: $C = K^e \pmod N$.
- Suppose $K = K_1 \cdot K_2$ where $K_1, K_2 < 2^{34}$. (prob. ≈20%)
  Then: $C/K_1^e = K_2^e \pmod N$
- Build table: $C/1^e, C/2^e, C/3^e, \dots, C/2^{34e}$. time: 2^{34}

  For $K_2 = 0,\dots, 2^{34}$ test if K_2^e is in table. time: $2^{34} \cdot 34$
- Attack time: $\approx 2^{40} \ll 2^{64}$

RSA pub-key encryption (ISO std)

- (E_s, D_s): symmetric encryption scheme, AE-secure
  H: $Z_N \rightarrow K$ where K is key space of (E_s, D_s)

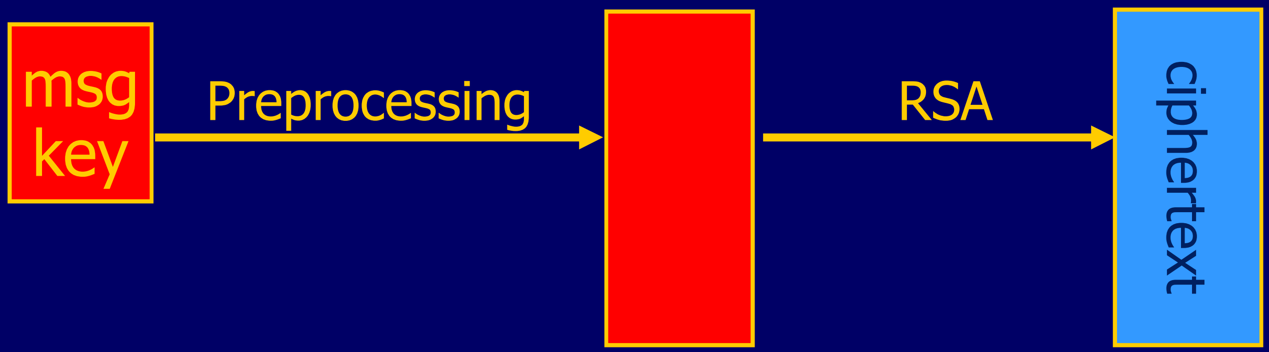
- G: generate RSA params: pk = (N,e), sk = (N,d)

- E(pk, m): (1) choose random x in Z_N
  (2) $u \leftarrow RSA(x) = x^e$, $k \leftarrow H(x)$
  (3) output $(u, E_s(k,m))$

- D(sk, (u, c)): output $D_s(H(RSA^{-1}(u)), c)$

RSA encryption in practice

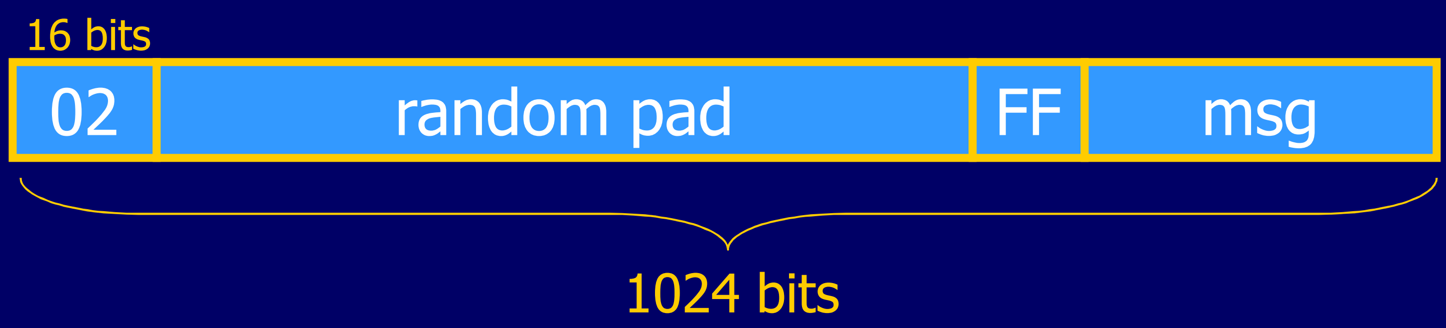
- Never use textbook RSA.
- RSA in practice (since ISO standard is not often used) :



- Main question:
  - How should the preprocessing be done?
  - Can we argue about security of resulting system?

PKCS1 V1.5

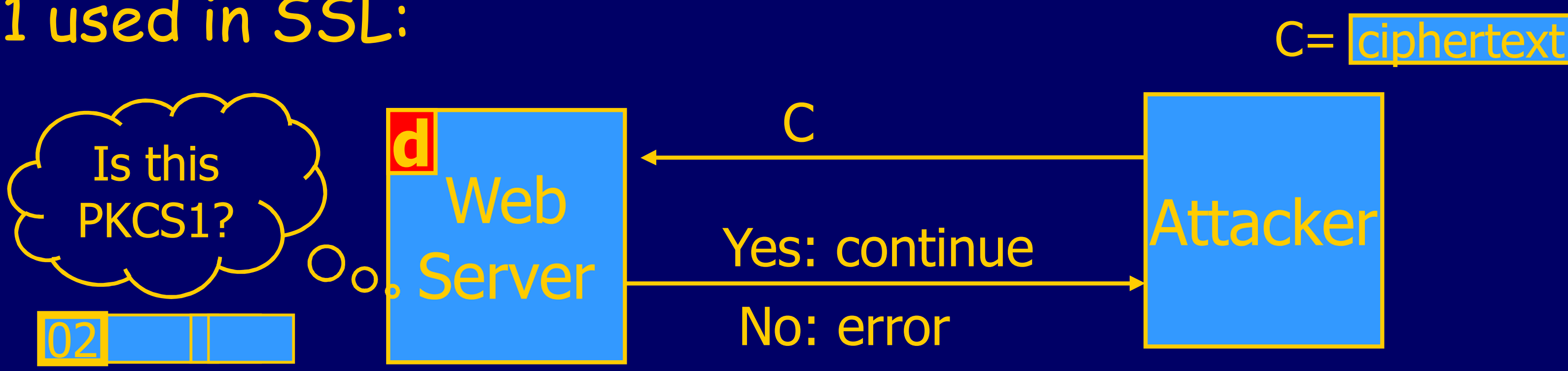
- PKCS1 mode 2: (encryption)



- Resulting value is RSA encrypted.
- Widely deployed in web servers and browsers.
- No security analysis !!

Attack on PKCS1

- Bleichenbacher 98. *Chosen-ciphertext attack.*
- PKCS1 used in SSL:



$\Rightarrow$ attacker can test if 16 MSBs of plaintext = '02'.

- *Attack: to decrypt a given ciphertext C do:*
  - Pick $r \in Z_N$. Compute $C' = r^e \cdot C = (r \cdot \text{PKCS1}(M))^e$.
  - Send C' to web server and use response.

Review: chosen CT security (CCS)

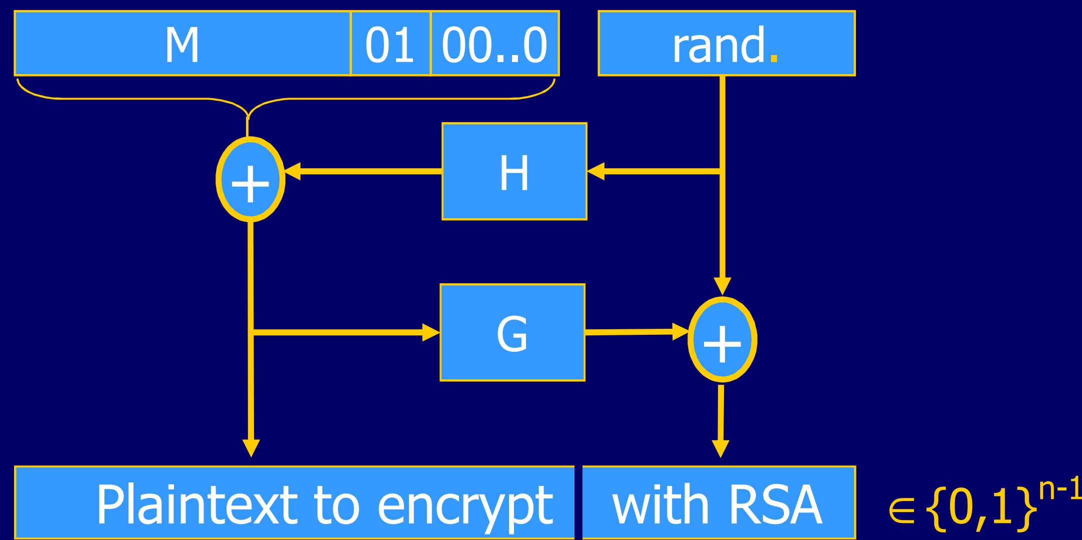
- No efficient attacker can win the following game:
  (with non-negligible advantage)

Attacker wins if b=b′

PKCS1 V2.0 - OAEP

- New preprocessing function: OAEP [BR94]

Check pad
on decryption.
Reject CT if invalid.

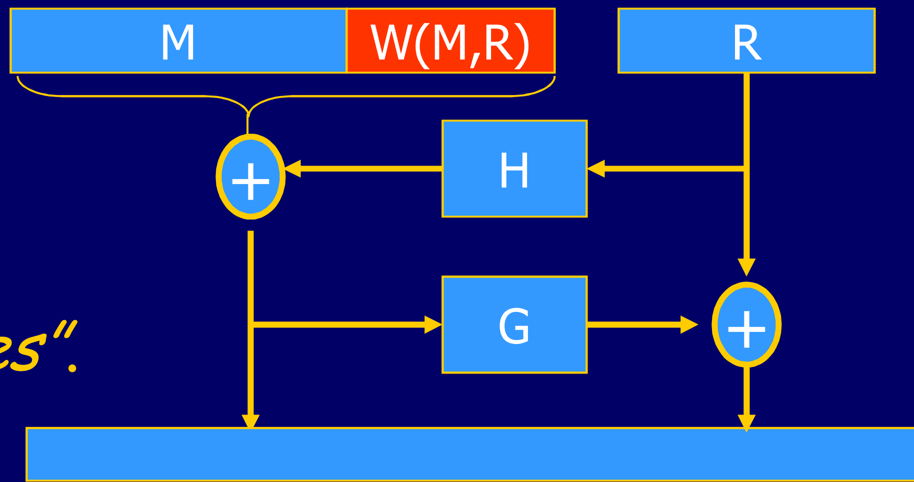


- Thm [FOPS'01] : RSA is trap-door permutation ⇒
  RSA-OAEP is CCS when H,G are "*random oracles*".
- In practice: use SHA-256 for H and G.

OAEP Improvements

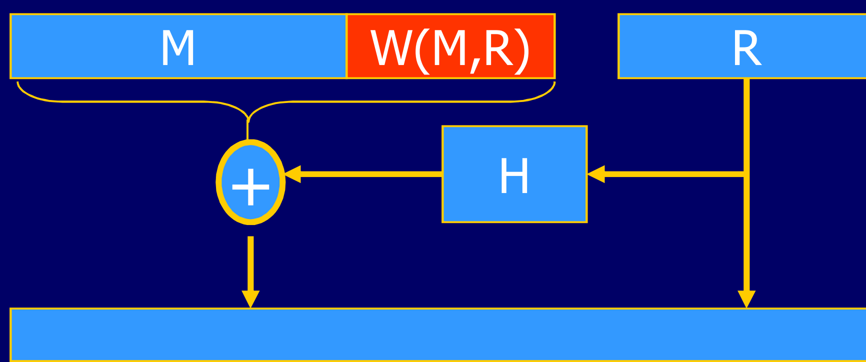
- OAEP+: [Shoup'01]

  ∀ trap-door permutation F
  F-OAEP+ is CCS when
  H,G,W are "*random oracles*".



- SAEP+: [B'01]

  RSA trap-door perm ⇒
  RSA-SAEP+ is CCS when
  H,W are "*random oracle*".



Subtleties in implementing OAEP [M '00]

```
OAEP-decrypt(C) {
    error = 0;
        ........
    if ( RSA^-1(C) > 2^n-1 )
        { error =1;  goto exit; }
        ........
    if ( pad(OAEP^-1(RSA^-1(C))) != "01000" )
        { error = 1;  goto exit; }
}
```

- Problem: timing information leaks type of error.
  $\Rightarrow$ Attacker can decrypt any ciphertext C.
- Lesson: Don't implement RSA-OAEP yourself …

Part II:
Is RSA a One-Way Function?

Is RSA a one-way permutation?

- To invert the RSA one-way function (without d) attacker must compute:

  M from $C = M^e \pmod{N}$.

- How hard is computing e'th roots modulo N ??

- Best known algorithm:
  - Step 1: factor N. (hard)
  - Step 2: Find e'th roots modulo p and q. (easy)

Shortcuts?

- Must one factor N in order to compute e'th roots? Exists shortcut for breaking RSA without factoring?
- To prove no shortcut exists show a reduction:
  - Efficient algorithm for e'th roots mod N
    $\Rightarrow$ efficient algorithm for factoring N.
  - Oldest problem in public key cryptography.
- Evidence no reduction exists: (BV'98)
  - "Algebraic" reduction $\Rightarrow$ factoring is easy.
  - Unlike Diffie-Hellman (Maurer'94).

Improving RSA's performance

- To speed up RSA decryption use small private key d. $C^d = M \pmod{N}$

  - Wiener87: if $d < N^{0.25}$ then RSA is insecure.
  - BD'98: if $d < N^{0.292}$ then RSA is insecure
    (open: $d < N^{0.5}$)

  - Insecure: priv. key d can be found from (N,e).

  - Small d should never be used.

Wiener's attack

- Recall: $e \cdot d = 1 \pmod{\varphi(N)}$

$$\Rightarrow \quad \exists\, k \in Z: \quad e \cdot d = k \cdot \varphi(N) + 1$$

$$\Rightarrow \quad \left| \frac{e}{\varphi(N)} - \frac{k}{d} \right| \leq \frac{1}{d\varphi(N)}$$

$\varphi(N) = N-p-q+1 \quad \Rightarrow \quad |N- \varphi(N)| \leq p+q \leq 3\sqrt{N}$

$d \leq N^{0.25}/3 \quad \Rightarrow \left| \frac{e}{N} - \frac{k}{d} \right| \leq \frac{1}{2d^2}$

Continued fraction expansion of e/N gives k/d.

$e \cdot d = 1 \pmod{k} \quad \Rightarrow \quad gcd(d,k)=1$

RSA With Low public exponent

- To speed up RSA encryption (and sig. verify) use a small e. $C = M^e \pmod N$
- Minimal value: $e=3$ ($\gcd(e, \varphi(N)) = 1$)
- Recommended value: $e=65537=2^{16}+1$

  Encryption: 17 mod. multiplies.

- Several weak attacks. Non known on RSA-OAEP.
- Asymmetry of RSA: fast enc. / slow dec.
  - ElGamal: approx. same time for both.

Implementation attacks

- Attack the implementation of RSA.
- Timing attack: (Kocher 97)
  The time it takes to compute $C^d \pmod N$
  can expose d.
- Power attack: (Kocher 99)
  The power consumption of a smartcard while
  it is computing $C^d \pmod N$ can expose d.
- Faults attack: (BDL 97)
  A computer error during $C^d \pmod N$
  can expose d.

  OpenSSL defense: check output. 5% slowdown.

Key lengths

- Security of public key system should be comparable to security of block cipher.

NIST:

<u>Cipher key-size</u>	<u>Modulus size</u>
$\leq$ 64 bits	512 bits.
80 bits	1024 bits
128 bits	3072 bits.
256 bits (AES)	**<u>15360</u>** bits

- High security $\Rightarrow$ very large moduli.

  Not necessary with Elliptic Curve Cryptography.