

CS 4803

Computer and Network Security

Alexandra (Sasha) Boldyreva
 RSA function and encryption.

1

The RSA system. The basics.

- Def. Let $N, f \geq 1$ be integers. The RSA function associated to N, f is the function $\text{RSA}_{N,f} : \mathbb{Z}_N^* \rightarrow \mathbb{Z}_N^*$ defined by

$$\text{RSA}_{N,f}(w) = w^f \pmod N \text{ for all } w \in \mathbb{Z}_N^*.$$
- Claim. Let $N \geq 2$ and $e, d \in \mathbb{Z}_{\phi(N)}^*$ be integers such that $ed \equiv 1 \pmod{\phi(N)}$. Then the RSA functions $\text{RSA}_{N,e}$ and $\text{RSA}_{N,d}$ are
 - both permutations on \mathbb{Z}_N^* and
 - inverses of each other, ie. $\text{RSA}_{N,e}^{-1} = \text{RSA}_{N,d}$ and $\text{RSA}_{N,d}^{-1} = \text{RSA}_{N,e}$.
- Proof. For any $x \in \mathbb{Z}_N^*$, modulo N :
 - $\text{RSA}_{N,d}(\text{RSA}_{N,e}(x)) \equiv (x^e)^d \equiv x^{ed} \equiv x^{ed \pmod{\phi(N)}} \equiv x^1 \equiv x$
 - Similarly, $\text{RSA}_{N,e}(\text{RSA}_{N,d}(y)) \equiv y$

2

- The RSA function associated to N, f can be efficiently computed using MOD-EXP(\cdot, f, N) algorithm.
 - Hence, $\text{RSA}_{N,e}(\cdot)$ is efficiently computable given N, e
 - $\text{RSA}_{N,e}^{-1}(\cdot) = \text{RSA}_{N,d}(\cdot)$ is efficiently computable given N, d
 - But $\text{RSA}_{N,e}^{-1}(\cdot) = \text{RSA}_{N,d}(\cdot)$ is believed hard (without d) for a proper choice of parameters (good for crypto).
- Let's build algorithms that generate RSA parameters.
- Claim. There is an $O(k^2)$ time algorithm that on inputs $\phi(N), e$ where $e \in \mathbb{Z}_{\phi(N)}^*$ and $N < 2^k$, returns $d \in \mathbb{Z}_{\phi(N)}^*$ satisfying $ed \equiv 1 \pmod{\phi(N)}$.

3

- The RSA modulus generator:

Algorithm $\mathcal{K}_{\text{mod}}^s(k)$

```

 $\ell_1 \leftarrow \lfloor k/2 \rfloor ; \ell_2 \leftarrow \lceil k/2 \rceil$ 
Repeat
   $p \xleftarrow{\$} \{2^{\ell_1-1}, \dots, 2^{\ell_1} - 1\} ; q \xleftarrow{\$} \{2^{\ell_2-1}, \dots, 2^{\ell_2} - 1\}$ 
Until the following conditions are all true:
- TEST-PRIME( $p$ ) = 1 and TEST-PRIME( $q$ ) = 1
-  $p \neq q$ 
-  $2^{k-1} \leq pq$ 
 $N \leftarrow pq$ 
Return  $(N, p, q)$ 

```

4

- The random-exponent RSA generator:

Algorithm $\mathcal{K}_{\text{rsa}}^{\$}(k)$

- $(N, p, q) \xleftarrow{\$} \mathcal{K}_{\text{mod}}^{\$}$
- $M \leftarrow (p-1)(q-1)$
- $e \xleftarrow{\$} \mathbf{Z}_M^*$
- $d \leftarrow \text{MOD-INV}(e, M)$
- Return $((N, e), (N, p, q, d))$
-

- Often for efficiency we want e to be small, e.g. 3. Then

Algorithm $\mathcal{K}_{\text{rsa}}^e(k)$

Repeat

$(N, p, q) \xleftarrow{\$} \mathcal{K}_{\text{mod}}^{\$}(k)$

Until

- $e < (p-1)$ and $e < (q-1)$
- $\text{gcd}(e, (p-1)) = \text{gcd}(e, (q-1)) = 1$

$M \leftarrow (p-1)(q-1)$

$d \leftarrow \text{MOD-INV}(e, M)$

Return $((N, e), (N, p, q, d))$

5

Hard RSA-related problem

- Def [ow-kea] For an adversary A consider an experiment:

- Experiment $\text{Exp}_{\mathcal{K}_{\text{rsa}}}^{\text{ow-kea}}(A)$
- $((N, e), (N, p, q, d)) \xleftarrow{\$} \mathcal{K}_{\text{rsa}}(k)$
- $x \xleftarrow{\$} \mathbf{Z}_N^*$; $y \leftarrow x^e \bmod N$
- $x' \xleftarrow{\$} A(N, e, y)$
- If $x' = x$ then return 1 else return 0

The *ow-kea* - advantage of A is defined as the probability of the above experiment outputting 1.

RSA assumption. There is no efficient adversary whose *ow-kea* advantage is not negligible.

6

$$x \begin{array}{c} \xrightarrow{\text{easy}} \\ \xleftarrow{\text{easy with } d} \\ \text{hard without } d \end{array} x^e \bmod N$$

- Let's study several known attacks that "break" RSA, i.e. compute an inverse of the RSA function on random inputs without knowing the trapdoor.

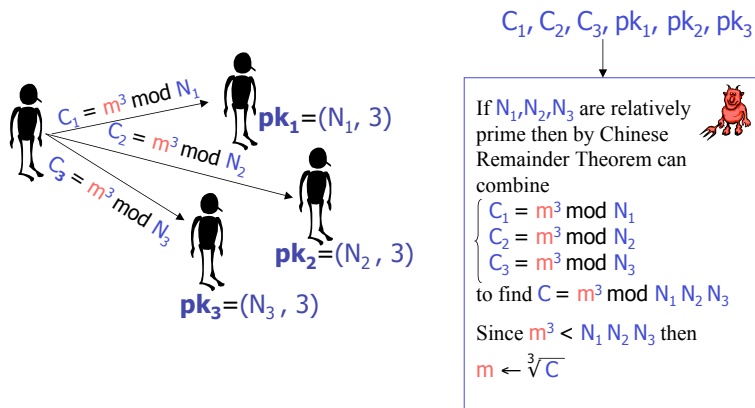
7

Some known attacks on RSA function

- Factoring the RSA modulus.
 - If one can factor N , i.e. compute p, q , s.t. $N=pq$ then one can compute $d=e^{-1} \bmod (p-1)(q-1)$
 - The best known algorithm to factor is GNFS.
- Theorem [RSA with low secret exponent]. Let $N=pq$, where $q < p < 2q$ and p, q are prime. Let $d < 1/3 \cdot N^{1/4}$. Then given (N, e) one can efficiently compute d .

8

- Hastad's broadcast attack for RSA with low public exponent.



9

It seems natural to try to fix the problem, e.g. by padding the messages before applying the RSA function.

However, there are attacks known such as

- broadcast attack on padded RSA with low public exponents,
- related-message attack on RSA with low public exponent,
- short pad attack

that show that patching this problem is not easy and not such a good idea.

10

- Theorem.** Let $N=pq$ be a k -bit RSA modulus. Then given $k/4$ least or most significant bits of p , one can efficiently factor N .
- Theorem.** Let N be a k -bit RSA modulus and let d be an RSA secret exponent. Then given the $k/4$ least significant bits of d , one can efficiently recover all bits of d .

Reference: <http://crypto.stanford.edu/~dabo/abstracts/RSAattack-survey.html>

11

Plain RSA encryption scheme

Algorithm $\mathcal{K}_{\text{mod}}^{\$}$

$\ell_1 \leftarrow \lfloor k/2 \rfloor; \ell_2 \leftarrow \lceil k/2 \rceil$

Repeat

$p \leftarrow \{2^{\ell_1-1}, \dots, 2^{\ell_1} - 1\}; q \leftarrow \{2^{\ell_2-1}, \dots, 2^{\ell_2} - 1\}$

Until the following conditions are all true:

- TEST-PRIME(p) = 1 and TEST-PRIME(q) = 1

- $p \neq q$

- $2^{k-1} \leq pq$

$N \leftarrow pq$

Return (N, p, q)

Algorithm $\mathcal{K}_{\text{rsa}}^{\$}$

$(N, p, q) \leftarrow \mathcal{K}_{\text{mod}}^{\$}$

$\Phi \leftarrow (p-1)(q-1)$

$e \leftarrow \mathbb{Z}_{\Phi}^*$

$d \leftarrow \text{MOD-INV}(e, \Phi)$

Return $((N, e), (N, p, q, d))$

Algorithm \mathcal{K}
 $((N, e), (N, p, q, d)) \leftarrow \mathcal{K}_{\text{rsa}}^{\$}$
 Return $((N, e), (N, d))$

Algorithm $\mathcal{E}_{(N,e)}(M)$
 $C \leftarrow M^e \bmod N$
 Return C

Algorithm $\mathcal{D}_{(N,d)}(C)$
 $M \leftarrow C^d \bmod N$
 Return M

12

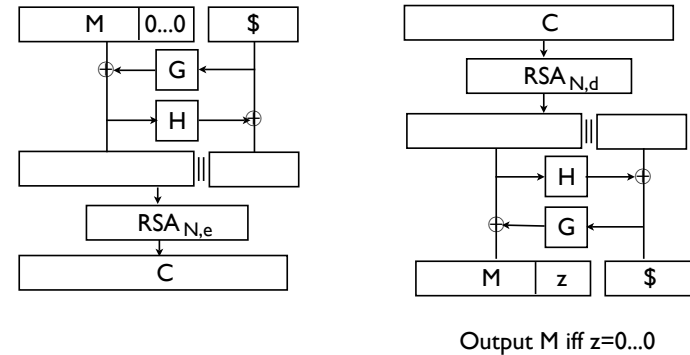
Plain RSA is not secure

- Under the RSA assumption it is hard to recover a message given the public key and a ciphertext.
- $$M \begin{array}{c} \xrightarrow{\text{easy}} \\ \xleftarrow{\text{easy with } d} \end{array} C = M^e \pmod N$$

hard without d
- Nevertheless, the plain RSA is not a good encryption scheme.
- E.g. it is not IND-CPA secure. Why?
- One might try to add a random padding to a message before applying the RSA function, but as we saw it does not necessarily help.

13

RSA-OAEP



G, H are hash functions

14

RSA-OAEP

Hash functions: $G: \{0, 1\}^{k_0} \rightarrow \{0, 1\}^{k-k_0}$ $H: \{0, 1\}^{k-k_0} \rightarrow \{0, 1\}^{k_0}$

Algorithm \mathcal{K}
 $((N, e), (N, p, q, d)) \xleftarrow{\$} \mathcal{X}_{rsa}^{\$}$
 Return $((N, e), (N, d))$

Algorithm $\mathcal{E}_{(N,e)}(M)$
 $r \xleftarrow{\$} \{0, 1\}^{k_0}$
 $s \leftarrow M || 0^{k_1} \oplus G(r)$
 $t \leftarrow r \oplus H(s)$
 $C \leftarrow \langle s || t \rangle^e \pmod N$
 Return C

Algorithm $\mathcal{D}_{(N,d)}(C)$
 $W \leftarrow C^d \pmod N$
 Parse W as $s || t$
 $r \leftarrow H(s) \oplus t$
 $M' \leftarrow s \oplus G(r)$
 Parse M' as $M || z$
 If $z = 0^{k_1}$ then return M else return \perp

15

Security of RSA-OAEP

- RSA-OAEP has not been proven IND-CCA secure.
- But it is proven IND-CCA secure assuming the RSA assumption, and when G, H are modeled as random oracles.
- Assuming the RSA problem is hard, RSA-OAEP is IND-CCA secure in the Random Oracle (RO) model.

16

RO model

- The RO model assumes that all parties (adversary included) have oracle access to a truly random function.
- This is not true in reality. The model is ideal.
- In practice real hash functions such as SHA1 are used in place of random oracles.
- The belief is that security of the practical schemes holds in the standard model.
- However there are several examples of uninstantiable schemes (the schemes that are proven secure in the RO model but shown to be insecure for any instantiation of random oracles with a real function.)
- All currently known uninstantiable schemes are rather artificial.

17

Hybrid encryption

- Asymmetric encryption uses number-theoretic operations and is slower than symmetric encryption that often uses block ciphers.
- Also we often want to encrypt long messages.
- In practice one usually
 1. encrypts a randomly chosen symmetric key K using an asymmetric encryption algorithm and then
 2. encrypts a message using a symmetric encryption algorithm and K .
- This is called hybrid encryption

18

Hybrid encryption

- Let $\mathcal{AE} = (\mathcal{K}^a, \mathcal{E}^a, \mathcal{D}^a)$ be an asymmetric encryption scheme and let $\mathcal{SE} = (\mathcal{K}^s, \mathcal{E}^s, \mathcal{D}^s)$ be a symmetric encryption scheme, s.t. the set of keys for \mathcal{SE} is always in the message space of \mathcal{AE} .

Then the associated hybrid scheme $\overline{\mathcal{AE}} = (\mathcal{K}^a, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ is as follows:

- | | |
|--|--|
| <ul style="list-style-type: none"> • Algorithm $\overline{\mathcal{E}}_{pk}(M)$ • $K \xleftarrow{\\$} \mathcal{K}^s$; $C^s \xleftarrow{\\$} \mathcal{E}_K^s(M)$ • If $C^s = \perp$ then return \perp • $C^a \xleftarrow{\\$} \mathcal{E}_{pk}^a(K)$; $C \leftarrow (C^a, C^s)$ • Return C | <ul style="list-style-type: none"> • Algorithm $\overline{\mathcal{D}}_{sk}(C)$ • Parse C as (C^a, C^s) • $K \leftarrow \mathcal{D}_{sk}^a(C^a)$ • If $K = \perp$ then return \perp • $M \leftarrow \mathcal{D}_K^s(C^s)$ • Return M |
|--|--|
- Note that the hybrid scheme is an asymmetric encryption scheme

19

Hybrid encryption

- Theorem. Let $\mathcal{AE} = (\mathcal{K}^a, \mathcal{E}^a, \mathcal{D}^a)$ be an asymmetric encryption scheme and let $\mathcal{SE} = (\mathcal{K}^s, \mathcal{E}^s, \mathcal{D}^s)$ be a symmetric encryption scheme, s.t. the set of keys for \mathcal{SE} is always in the message space of \mathcal{AE} . Let $\overline{\mathcal{AE}} = (\mathcal{K}^a, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ be the associated hybrid scheme as defined on the previous slide. Then if the components are IND-CPA, then the associated hybrid scheme is also IND-CPA.
- An analogous theorem can be stated and proved for the case of chosen-ciphertext attacks (if the components are IND-CCA secure, then the hybrid scheme is IND-CCA secure).

20