Textbook RSA and Semantic Security

Douglas Wikström
KTH Stockholm
dog@csc.kth.se

February 16

- Textbook RSA
- Semantic Security

The RSA Cryptosystem (1/2)

Key Generation.

- Choose n-bit primes p and q randomly and define $N = pq$.
- Choose e randomly in $\mathbb{Z}^*_{\phi(N)}$ and compute $d = e^{-1} \bmod \phi(N)$.
- Output the key pair $((N, e), (p, q, d))$, where (N, e) is the public key and (p, q, d) is the secret key.

The RSA Cryptosystem (2/2)

Encryption. Encrypt a plaintext m by computing

$$c = m^e \bmod N \ .$$

Decryption. Decrypt a ciphertext c by computing

$$m = c^d \bmod N \ .$$

Factoring From Order of Multiplicative Group

Given N and $\phi(N)$, we can find p and q by solving

$$\begin{aligned} N &= pq \\ \phi(N) &= (p-1)(q-1) \end{aligned}$$

Factoring From Encryption & Decryption Exponents (1/3)

- If $N = pq$ with p and q prime, then the CRT implies that

$$x^2 = 1 \bmod N$$

has **four distinct solutions** in $\mathbb{Z}_N^*$, and **two** of these are **non-trivial**, i.e., distinct from ± 1.

- If x is a non-trivial root, then

$$(x-1)(x+1) = tN$$

but $N \nmid (x-1), (x+1)$, so

$$\gcd(x-1, N) > 1 \quad \text{and} \quad \gcd(x+1, N) > 1 \ .$$

Factoring From Encryption & Decryption Exponents (2/3)

- The encryption & decryption exponents satisfy

$$ed = 1 \bmod \phi(N) \enspace ,$$

so if we have $ed - 1 = 2^s r$ with r odd, then

$$\begin{aligned}(p-1) &= 2^{s_p} r_p \mid 2^s r \quad \text{and} \\ (q-1) &= 2^{s_q} r_q \mid 2^s r \enspace .\end{aligned}$$

- If $v \in \mathbb{Z}_N^*$ is random, then $w = v^r$ is random in the subgroup of elements with order 2^i for some $0 \leq i \leq \max\{s_p, s_q\}$.

Factoring From Encryption & Decryption Exponents (3/3)

Suppose $s_p \geq s_q$. Then for some $0 < i < s_p$,

$$w^{2^i} = \pm 1 \bmod q$$

and

$$w^{2^i} \bmod p$$

is uniformly distributed in $\{1, -1\}$.

Conclusion.
$w^{2^i} \pmod N$ is a non-trivial root of 1 with probability $1/2$, which allows us to factor N.

Small Encryption Exponents

Suppose that $e = 3$ is used by all parties as encryption exponent.

- **Small Message.** If m is small, then $m^e < N$. Thus, **no reduction takes place**, and m can be computed in $\mathbb{Z}$ by taking the eth root.

Small Encryption Exponents

Suppose that $e = 3$ is used by all parties as encryption exponent.

- **Small Message.** If m is small, then $m^e < N$. Thus, **no reduction takes place**, and m can be computed in $\mathbb{Z}$ by taking the eth root.
- **Identical Plaintexts.** If a message m is encrypted under moduli N_1, N_2, N_3, and N_4 as c_1, c_2, c_3, and c_3, then CRT implies a $c \in \mathbb{Z}^*_{N_1 N_2 N_3 N_4}$ such that $c = c_i \bmod N_i$ and $c = m^e \bmod N_1 N_2 N_3 N_4$ with $m < N_i$.

Additional Caveats

- **Identical Moduli.** If a message m is encrypted as c_1 and c_2 using distinct encryption exponents e_1 and e_2 with $\gcd(e_1, e_2) = 1$, and a modulus N, then we can find a, b such that $ae_1 + be_2 = 1$ and $m = c_1^a c_2^b \bmod N$.

Additional Caveats

- **Identical Moduli.** If a message m is encrypted as c_1 and c_2 using distinct encryption exponents e_1 and e_2 with $\gcd(e_1, e_2) = 1$, and a modulus N, then we can find a, b such that $ae_1 + be_2 = 1$ and $m = c_1^a c_2^b \bmod N$.
- **Reiter-Franklin Attack.** If e is small then encryptions of m and $f(m)$ for a polynomial $f \in \mathbb{Z}_N[x]$ allows efficient computation of m.

Additional Caveats

- **Identical Moduli.** If a message m is encrypted as c_1 and c_2 using distinct encryption exponents e_1 and e_2 with $\gcd(e_1, e_2) = 1$, and a modulus N, then we can find a, b such that $ae_1 + be_2 = 1$ and $m = c_1^a c_2^b \bmod N$.
- **Reiter-Franklin Attack.** If e is small then encryptions of m and $f(m)$ for a polynomial $f \in \mathbb{Z}_N[x]$ allows efficient computation of m.
- **Wiener's Attack.** If $a < N^{1/4}$ and $q < p < 2q$, then N can be factored in polynomial time with good probability.

Factoring

The obvious way to break RSA is to factor the public modulus N and recover the prime factors p and q.

- The number field sieve factors N in time

$$O\left(e^{(1.92+o(1))\left((\ln N)^{1/3}+(\ln\ln N)^{2/3}\right)}\right) .$$

- The elliptic curve method factors N in time

$$O\left(e^{(1+o(1))\sqrt{2\ln p\ln\ln p}}\right) .$$

Factoring

The obvious way to break RSA is to factor the public modulus N and recover the prime factors p and q.

- The number field sieve factors N in time

$$O\left(e^{(1.92+o(1))\left((\ln N)^{1/3}+(\ln\ln N)^{2/3}\right)}\right) \quad .$$

- The elliptic curve method factors N in time

$$O\left(e^{(1+o(1))\sqrt{2\ln p\ln\ln p}}\right) \quad .$$

Note that the latter only depends on the size of p!

Birthday Paradox

Lemma. Let $q_0, \ldots, q_k$ be randomly chosen in a set S. Then

1. the probability that $q_i = q_j$ for some $i \neq j$ is approximately $1 - e^{-\frac{k^2}{2s}}$, where $s = |S|$, and
2. with $k \approx \sqrt{-2s \ln(1-\delta)}$ we have a collision-probability of δ.

Proof.

$$\left(\frac{s-1}{s}\right)\left(\frac{s-2}{s}\right)\cdot\ldots\cdot\left(\frac{s-k}{s}\right) \approx \prod_{i=1}^{k} e^{-\frac{i}{s}} \approx e^{-\frac{k^2}{2s}} \quad .$$

Pollard-ρ (1/2)

Fact. Let $a, a' \in \mathbb{Z}_N$ such that:

$$a > a' \quad \text{and} \quad a = a' \bmod \mathbf{p} \enspace ,$$

then

$$p \leq \gcd(a - a', n) < n \enspace .$$

Pollard-ρ (2/2)

Idea.

1. Generate "random" elements $a_1, a_2, \ldots$ using polynomial $f(\cdot) \in \mathbb{Z}_N[x]$ recursively, i.e., $a_i = f(a_{i-1}) \bmod N$.

Pollard-ρ (2/2)

Idea.

1. Generate "random" elements $a_1, a_2, \ldots$ using polynomial $f(\cdot) \in \mathbb{Z}_N[x]$ recursively, i.e., $a_i = f(a_{i-1}) \bmod N$.
2. Find "collisions" (a_i, a_j) after $O(\sqrt{p})$ samples.

Pollard-ρ (2/2)

Idea.

1. Generate "random" elements $a_1, a_2, \ldots$ using polynomial $f(\cdot) \in \mathbb{Z}_N[x]$ recursively, i.e., $a_i = f(a_{i-1}) \bmod N$.
2. Find "collisions" (a_i, a_j) after $O(\sqrt{p})$ samples.
3. Avoid GCD-computations using:

$$a = a' \bmod p \quad \Longrightarrow \quad f(a) = f(a') \bmod p$$

and "double stepping".

Random Squares

Fact. Given $x \neq \pm y \bmod N$ such that $x^2 = y^2 \bmod N$, $\gcd(x - y, N)$ is a non-trivial factor of N.

Idea.

1. Find z_i, primes $p_{i,j}$, and exponents $e_{i,j}$ such that:

$$z_i^2 = \prod_j p_{i,j}^{e_{i,j}}$$

2. Find subset S such that

$$\prod_{i \in S} z_i^2 = \prod_{i \in S} \prod_j p_{i,j}^{e_{i,j}} = \prod_j p_{i,j}^{e'_{i,j}}$$

with $e'_{i,j}$ even, i.e., both sides are squares.

Semantic Security (1/3)

- RSA clearly provides some kind of "security", but it is clear that we need to be more careful with what we ask for.

Semantic Security (1/3)

- RSA clearly provides some kind of “security”, but it is clear that we need to be more careful with what we ask for.
- Intuitively, we want to leak no information of the encrypted plaintext.

Semantic Security (1/3)

- RSA clearly provides some kind of "security", but it is clear that we need to be more careful with what we ask for.
- Intuitively, we want to leak no **knowledge** of the encrypted plaintext.

Semantic Security (1/3)

- RSA clearly provides some kind of "security", but it is clear that we need to be more careful with what we ask for.
- Intuitively, we want to leak no **knowledge** of the encrypted plaintext.
- In other words, no function of the plaintext can efficiently be guessed notably better from its ciphertext than without it.

Semantic Security (2/3)

$\mathbf{Exp}^{b}_{\mathcal{CS},A}$ **(Semantic Security Experiment).**

1. **Generate Public Key.** $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$.
2. **Adversarial Choice of Messages.** $(m_0, m_1) \leftarrow A(\mathsf{pk})$.
3. **Guess Message.** Return the first bit output by $A(\mathsf{E}_{\mathsf{pk}}(m_b))$.

Semantic Security (2/3)

$\mathrm{Exp}^{b}_{\mathcal{CS},A}$ **(Semantic Security Experiment).**

1. **Generate Public Key.** $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$.
2. **Adversarial Choice of Messages.** $(m_0, m_1) \leftarrow A(\mathsf{pk})$.
3. **Guess Message.** Return the first bit output by $A(\mathsf{E}_{\mathsf{pk}}(m_b))$.

Definition. A cryptosystem $\mathcal{CS} = (\mathsf{Gen}, \mathsf{E}, \mathsf{D})$ is said to be **semantically secure** if for every polynomial time algorithm A

$$|\Pr[\mathrm{Exp}^{0}_{\mathcal{CS},A} = 1] - \Pr[\mathrm{Exp}^{1}_{\mathcal{CS},A} = 1]|$$

is negligible.

Semantic Security (3/3)

Every semantically secure cryptosystem must be probabilistic!

Semantic Security (3/3)

Every semantically secure cryptosystem must be probabilistic!

Theorem. Suppose that $\mathcal{CS} = (\mathsf{Gen}, \mathsf{E}, \mathsf{D})$ is a semantically secure cryptosystem.

Then the related cryptosystem where a $t(n)$-list of messages, with $t(n)$ polynomial, is encrypted by **repeated independent encryption** of each component using the **same public key** is also semantically secure.

Semantic Security (3/3)

Every semantically secure cryptosystem must be probabilistic!

Theorem. Suppose that $\mathcal{CS} = (\mathsf{Gen}, \mathsf{E}, \mathsf{D})$ is a semantically secure cryptosystem.

Then the related cryptosystem where a $t(n)$-list of messages, with $t(n)$ polynomial, is encrypted by **repeated independent encryption** of each component using the **same public key** is also semantically secure.

Semantic security is useful!