RSA: Concelation and Signature System

Software Security

Steffen Helke
Chair of Software Engineering

10th December 2018

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Objectives of today's lecture

- ➜ Getting to know *how to generate a key pair* for the asymmetric-key cryptosystem RSA
- ➜ Understanding the basic idea behind the *proof of correctness* using Fermat's little theorem
- ➜ Being able to perform attacks using *Fermat's factorization method* and based on the *multiplicative property of RSA*

RSA: An Asymmetric Concelation and Signature System

RSA: A Concelation and Signature System

Security level \ System type		Concelation: sym. — sym. concelation system	Concelation: asym. — asym. concelation system	Authentikation: sym. — sym. authentication system	Authentikation: asym. — digital signature system
information theoretical		Vernam-Chiffre (one-time pad)		Authentication codes	
crypto-graphi-cally strong against...	active attack	Pseudo-one-time-pad with s^2-mod-n-Generator			GMR
	passive attack		System with s^2-mod-n-Generator		
well re-searched	mathe-matical		RSA		RSA
	chaos	DES/AES		DES/AES	

Source: *Andreas Pfitzmann: Security in IT-Networks*, 2012

General Remarks on the RSA Cryptosystem

- Inventors
  - Ronald L. **R**iverest
  - Adi **S**hamir
  - Leonard M. **A**dleman
- First version published in **1978**
- Can be used either as an asymmetric encryption system or digital signature system
- RSA is based on the factorization assumption
- Under the assumption of factorization, however, the *correctness of RSA is not yet formally proven*

➜ Hence RSA is not cryptographically strong, only "well researched"!

Basis: Modular exponentiation of messages in the residue class ring

How to generate a suitable RSA key pair?

1. Select security parameter l
2. Select prime numbers p and q with $|p| \approx |q| = l$ and $p \neq q$
3. Calculate the product $n = p \cdot q$
4. Select c with $3 \leq c < \varphi(n)$ and $gcd(c, \varphi(n)) = 1$
   ➜ Note: $\varphi(n) = (p-1) \cdot (q-1)$
5. Calculate d as multiplicative inverse of c with $c \cdot d \equiv 1 \bmod \varphi(n)$ using the *extended Euclidean algorithm*

Secret parameters for key generation
- p, q und $\varphi(n)$

Secret key
- d, (p and q)

Public key
- c and n

Procedure: Concelation using RSA

Secret key	**Public key**
- d	- c und n

A plaintext m is communicated as follows

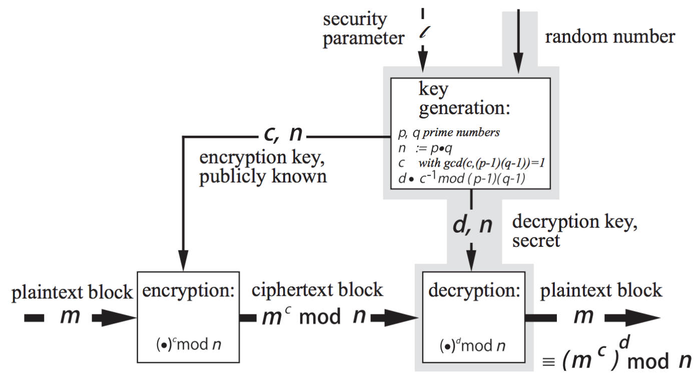
Encryption

➜ $e = m^c \bmod n$

Decryption

➜ $m = e^d \bmod n$

Asymmetric Encryption System
Naive version of RSA



Source: *Andreas Pfitzmann: Security in IT-Networks*, 2012

Note that this is only the *naive version of RSA*, which means that this setup is *not secure against attacks based on the multiplicative property* of RSA!

Procedure: Signing using RSA

Secret key	**Public key**
- $d \rightarrow$ is renamed to s	- $c \rightarrow$ is renamed to t and n

A plaintext m is signed as follows

Signing

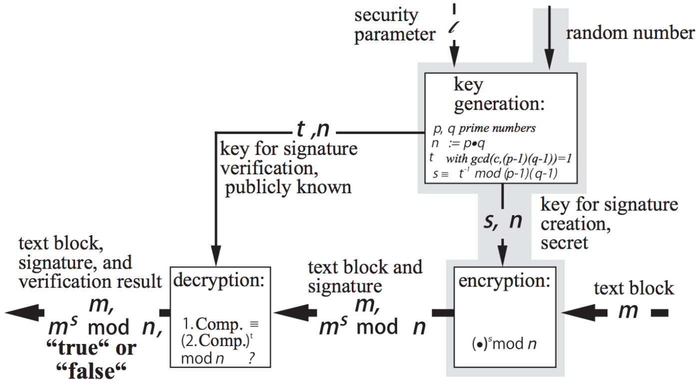
➜ $sig = m^s \bmod n$

Testing

➜ $m \stackrel{?}{=} sig^t \bmod n$

Digital Signature System

Naive version of RSA



Source: *Andreas Pfitzmann: Security in IT-Networks*, 2012

Note that this is only the *naive version of RSA*, which means that this setup is *not secure against attacks based on the multiplicative property* of RSA!

How to implement the RSA cryptosystem?

➜ What are the technical challenges to be solved?

1. Converting the plain text into a digital representation
2. Calculating large prime numbers efficiently
   ➜ e.g. using the *Miller-Rabin primality test*
3. Calculating the Multiplicative inverse
   ➜ e.g using the *Extended Euclidean algorithm*
4. Efficiently exponentiate large numbers
   ➜ e.g. by repeated squaring and multiplication
5. Strategies to prevent attacks, e.g. neutralizing the multiplicative structure of RSA

Module Size

➜ What impact does module size have on the RSA cryptosystem?

- Security increases with larger numbers
- But the *performance decreases*

Recommendations

- Prime numbers p and q shall differ by a few digits in length, i.e. $|p| \approx |q| = \imath$
- BSI recommends at least a module size of 2048 bits for systems that are to be operated until 2022 ($\widehat{=}$ decimal number with approx. 617 digits)
- From 2018 this directive is to be further increased to a module size of at least 3000 bits

Example: RSA Key Generation

How to generate a suitable RSA key pair?

1. We select the prime numbers $p = 11$ and $q = 13$ with $p \neq q$[1]
2. Calculate the product $n = 11 \cdot 13 = 143$
3. Calculate $\varphi(n) = (p-1) \cdot (q-1) = 120$
4. Select $c = 23$ with $3 \leq c < \varphi(n)$ and $gcd(c, \varphi(n)) = 1$
5. Calculate the *multiplicate inverse* of c for the residue class ring of $\varphi(n)$ to get c with
   $d \cdot c \equiv 1 \bmod \varphi(n)$
   is equivalent to
   $d \cdot c + k \cdot \varphi(n) = 1 = gcd(c, \varphi(n))$
   ➜ To solve this equation use the *Extended Euclidean algorithm*

[1] Note, the security parameter for specifying the key length is ignored in this example

Extended Euclidean algorithm

$$120 = 5 \cdot 23 + 5 \qquad (\varphi(n) = s_1 \cdot c + r_1)$$
$$23 = 4 \cdot 5 + 3 \qquad (c = s_2 \cdot r_1 + r_2)$$
$$5 = 1 \cdot 3 + 2 \qquad (r_1 = s_3 \cdot r_2 + r_3)$$
$$3 = 1 \cdot 2 + 1 \qquad (r_2 = s_4 \cdot r_3 + r_4)$$

In the reverse order, i.e. resolve all equations to the rest and then insert them step by step

$$1 = 3 - 1 \cdot 2 \qquad (r_4 = r_2 - 1 \cdot r_3)$$
$$1 = 3 - 1 \cdot (5 - 1 \cdot 3) \qquad (r_4 = r_2 - 1 \cdot (r_1 - 1 \cdot r_2))$$
$$1 = 2 \cdot 3 - 1 \cdot 5 \qquad (r_4 = 2 \cdot r_2 - 1 \cdot r_1)$$
$$1 = 2 \cdot (23 - 4 \cdot 5) - 1 \cdot 5 \qquad (r_4 = 2 \cdot (c - 4 \cdot r_1) - 1 \cdot r_1)$$
$$1 = 2 \cdot 23 - 9 \cdot 5 \qquad (r_4 = 2 \cdot c - 9 \cdot r_1)$$
$$1 = 2 \cdot 23 - 9 \cdot (120 - 5 \cdot 23) \qquad (r_4 = 2 \cdot c - 9 \cdot (\varphi(n) - 5 \cdot c))$$
$$1 = 47 \cdot 23 - 9 \cdot 120 \qquad (r_4 = 47 \cdot c - 9 \cdot \varphi(n))$$

➜ If $c \cdot d + k \cdot \varphi(n) = 1$, than $d = 47$!

Mathematical Backgrounds of the RSA Cryptosystem

Proof of Correctness for the RSA Cryptosystem (1)

Proof obligation

$$\forall\, m : \mathbb{Z}_n \bullet (m^c)^d = (m^d)^c = m^{c \cdot d} \equiv m \bmod n$$

Proof

according to the assumption applies

$$c \cdot d \equiv 1 \bmod \varphi(n)$$

with

$$\varphi(n) = (p-1) \cdot (q-1) \text{ and}$$
$$a \equiv b \bmod (c \cdot d) \;\Rightarrow\; a \equiv b \bmod c$$

we can deduce

$$c \cdot d \equiv 1 \bmod (p-1)$$
$$\Leftrightarrow\; \exists\, k : \mathbb{Z} \bullet c \cdot d = k \cdot (p-1) + 1$$

i.e. the following condition holds

$$m^{c \cdot d} \equiv m^{k \cdot (p-1)+1} \equiv m \cdot (m^{p-1})^k \bmod p$$

Proof of Correctness for the RSA Cryptosystem (2)

according to Fermat's little theorem we know

if $gcd(m, p) = 1$, than $m^{p-1} \equiv 1 \bmod p$

if m is not a multiple of p, we deduce

$$m \cdot (m^{p-1})^k \equiv m \cdot 1^k \equiv m \bmod p$$

if m is a multiple of p, we deduce $m \equiv 0 \bmod p$ and

$$m \cdot (m^{p-1})^k \equiv m \equiv 0 \bmod p$$

Since p is a prime number, there can be no other cases, i.e. it applies

$$m^{c \cdot d} \equiv m \bmod p$$

The proof is identical for the prime number q

$$m^{c \cdot d} \equiv m \bmod q$$

Using the Chinese Remainder Algorithm follows for $n = p \cdot q$

$$m^{c \cdot d} \equiv m \bmod n$$

Attacks for the RSA Cryptosystem

Types of Attacks

Total break

➜ obtaining the key

Universal break

➜ obtaining a procedure that is equivalent to the key

Message-dependent breaking

➜ breaking is only for some single messages possible

Selective breaks

➜ for a self-chosen message

Existential break

➜ for a random message

Attacks for the RSA Cryptosystem

– Fermat's Factorization Method –

Total Break by a Factorization Attack

➜ Fermat's Factorization Method

- Algorithm for the prime factorization of a natural number
- Method is only efficient when p and q differ only a little from $\sqrt{n}$

$$n = p \cdot q = \underbrace{(a+b)}_{p} \cdot \underbrace{(a-b)}_{q} = a^2 - b^2$$

- Idea: Search for numbers that fulfill the equation
- Start the search at $a = \lfloor \sqrt{n} + 1 \rfloor$
- Increase a stepwise by 1, until $(a^2 - n)$ is a square

Example for Fermat's Factorization Method

➜ Let $n = 143$; We are looking for the prime factors p and q

$$n = p \cdot q = \underbrace{(a+b)}_{p} \cdot \underbrace{(a-b)}_{q} = a^2 - b^2$$

- Select a with $a = \lfloor \sqrt{n} + 1 \rfloor = \lfloor \sqrt{143} + 1 \rfloor = 12$
- Find a suitable b, that fulfills the equation $n = a^2 + b^2$ for a
- $b^2 = a^2 - n = 12^2 - 143 = 1$

  ➜ 1 is a square!

- If $a = 12$ and $b = 1$ than we are able to calculate

  ➜ $p = a + b = 12 + 1 = 13$
  ➜ $q = a - b = 12 - 1 = 11$

How can we prevent the attack of Fermat?

➜ Note that the method is only efficient when p and q differ only a little from $\sqrt{n}$

Countermeasures

➜ For the key generation we have to select a module n, where n cannot be factorized with two prime numbers of approximately the same size

➜ The conditions $|p| \approx |q| = \imath$ and $p \neq q$ address this problem, i.e. the lengths of p and q must not be identical

Attacks for the RSA Cryptosystem

– Based on the multiplicative property of RSA –

Passive Attack Using Multiplicative Structure

Assumptions

1. the public key (t, n) for testing signatures,
2. the messages m_1 and m_2, and finally
3. the signatures m_1^s and m_2^s are known to the attacker

Passive Attack

- Calculate $m_3 := m_1 \cdot m_2$ and
- Obtaining the corresponding signature by applying the following calculation rule

$$m_3^s := m_1^s \cdot m_2^s = (m_1 \cdot m_2)^s \bmod n$$

➜ This attack is a *selective break*, where the victim must be *willing to sign two messages* for the attacker

Active Attack defined by Judy Moore

Goal

- The attacker is interested in getting any message signed by the victim

Procedure

1. Select the message to be signed arbitrarily, e.g. m_3
2. Select a number r randomly with $1 \leq r < n$ in such a way, that for r a multiplicative inverse r^{-1} exists
3. Calculate $m_2 := m_3 \cdot r^t \bmod n$
4. Send message m_2 to the victim for signing
5. Calculate $m_3^s := m_2^s \cdot r^{-1} \equiv (m_3 \cdot r^t)^s \cdot r^{-1} \equiv m_3^s \cdot r \cdot r^{-1} \bmod n$

➜ This is a *selective break*, where the victim must be *willing to sign one message* for the attacker

How can we prevent attacks based on the multiplicative property?

➜ Both attacks, the passive attack and the active attack of Judy Moore, use the multiplicative structure of RSA

Countermeasures

➜ *Collision-resistant hash function* are used to neutralize the multiplicative structure

➜ For a digital signature system create the signature only from the hashes, not from the plaintext, because for hash function h holds

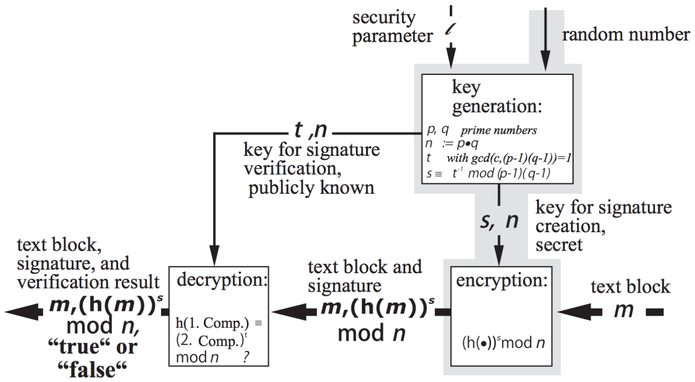
$$h(m_1)^s \cdot h(m_2)^s \neq (h(m_1) \cdot h(m_2))^s$$

➜ For a concelation system, attach the hash of the message to the plaintext and then encrypt the entire text block

➜ After decryption you need to perform additionally a redundancy check using the received hash value

Digital Signature System

Secure version of RSA

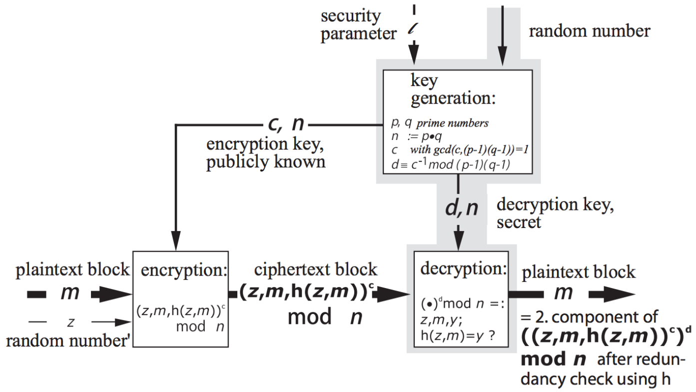


Source: *Andreas Pfitzmann: Security in IT-Networks*, 2012

Note that this secure version of RSA uses collision resistant hash function for signing. Hence the multiplicative structure of RSA is neutralized!

Asymmetric Encryption System

Secure version of RSA



Source: *Andreas Pfitzmann: Security in IT-Networks*, 2012

Simular to the secure RSA signature system this encryption system uses collision resistant hash function to neutralize the multiplicative structure!