RSA public key crypto system

12 The RSA public-key cryptosystem

12.1 Public-key cryptosystems

participants X:

- $X = A$, Alice, $X = B$ Bob, want private communication and authentication (signature) of documents from a *message domain* D.

- $X = E$ eavesdropper, listens on public channel

12 The RSA public-key cryptosystem

12.1 Public-key cryptosystems

participants X:

- $X = A$, Alice, $X = B$ Bob, want private communication and authentication (signature) of documents from a *message domain* D.
- $X = E$ eavesdropper, listens on public channel

keys of X:

- public key P_X
- secret key S_X
- for messages $M \in D$ one denotes by $P_X(M)$ resp. $S_X(M)$ the result of applying key P_X resp. S_X to d. Obvious overloading of notation.
- functions
$$P_X, S_X : D \to D$$
are bijective (i.e. permutations) and inverses of each other
$$P_X(S_X(M)) = S_X(P_X(M) = M \quad \text{for all } M \in D$$
- the hard part: even if P_X is known it is for the eavesdropper computationally extremely hard to discover S_X

12 The RSA public-key cryptosystem

12.1 Public-key cryptosystems

participants X:

- $X = A$, Alice, $X = B$ Bob, want private communication and authentication (signature) of documents from a *message domain* D.
- $X = E$ eavesdropper, listens on public channel

keys of X:

- public key P_X
- secret key S_X
- for messages $M \in D$ one denotes by $P_X(M)$ resp. $S_X(M)$ the result of applying key P_X resp. S_X to d. Obvious overloading of notation.
- functions
$$P_X, S_X : D \to D$$
are bijective (i.e. permutations) and inverses of each other
$$P_X(S_X(M)) = S_X(P_X(M) = M \quad \text{for all } M \in D$$
- the hard part: even if P_X is known it is for the eavesdropper computationally extremely hard to discover S_X

how Bob encrypts message M:

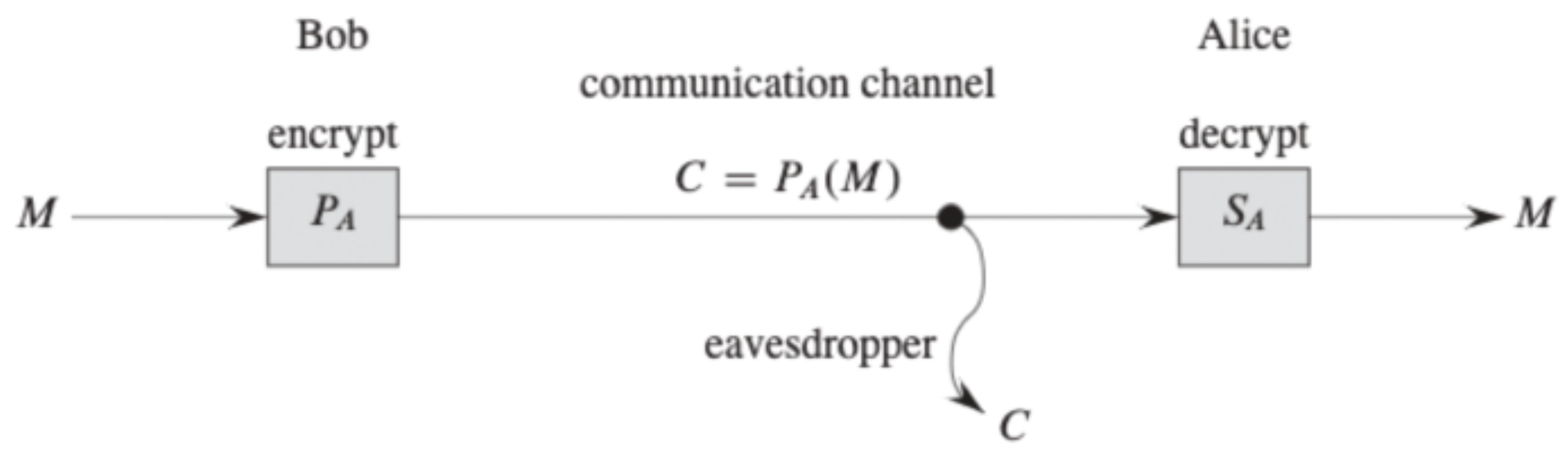

Figure 1: from [CLRS]: encrypting and decrypting a message

- Using public key of Alice Bob computes
$$C = P_A(M)$$
- eavesdropper can observe C and is hopefully unable do discover M
- using her secret key Alice decodes
$$M = S_A(C) = S_A(P_A(M)) = M$$

12 The RSA public-key cryptosystem

12.1 Public-key cryptosystems

participants X:

- $X = A$, Alice, $X = B$ Bob, want private communication and authentication (signature) of documents from a *message domain* D.
- $X = E$ eavesdropper, listens on public channel

keys of X:

- public key P_X
- secret key S_X
- for messages $M \in D$ one denotes by $P_X(M)$ resp. $S_X(M)$ the result of applying key P_X resp. S_X to d. Obvious overloading of notation.
- functions
$$P_X, S_X : D \to D$$
are bijective (i.e. permutations) and inverses of each other
$$P_X(S_X(M)) = S_X(P_X(M) = M \quad \text{for all } M \in D$$
- the hard part: even if P_X is known it is for the eavesdropper computationally extremely hard to discover S_X

how Alice signs message M':

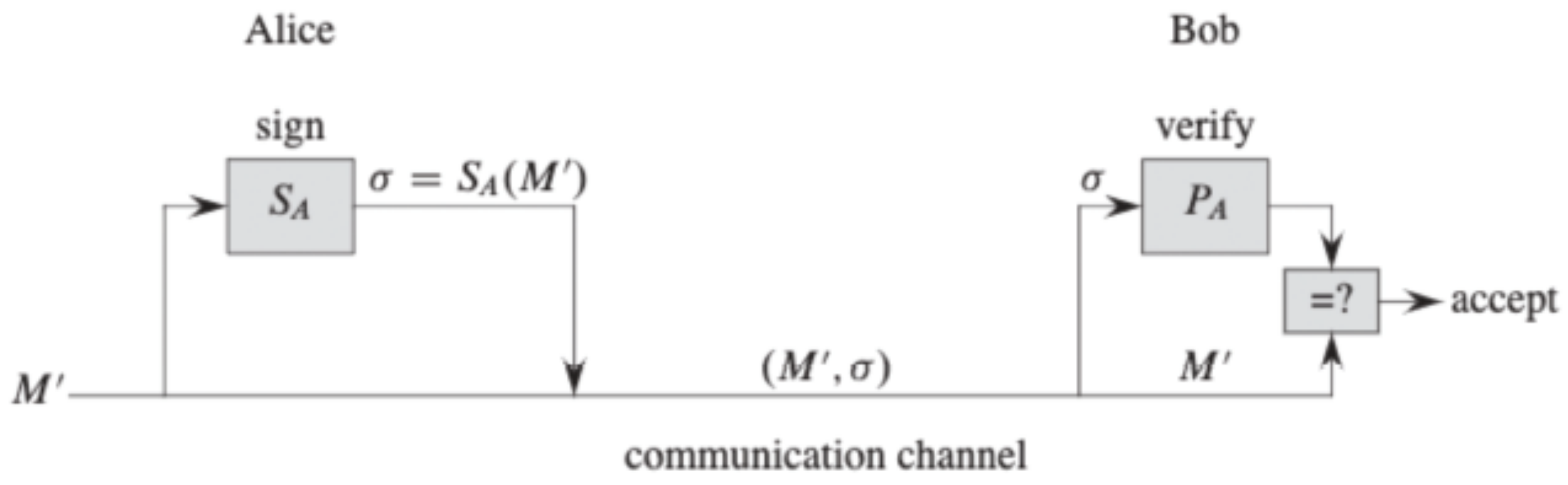

Figure 2: from [CLRS]: signing a message and checking the signature

- using her private key Alice computes signature
$$\sigma = S_A(M')$$
- then Alice transmits
$$(M', \sigma) \quad \text{. i.e. message and signature}$$
- using Alices public key Bob decodes the signature. The result should be the transmitted message
$$P_A(\sigma) = P_A(S_A(M')) = M'$$
He accepts, if this is the case

protocols can be combined

12 The RSA public-key cryptosystem

12.2 The RSA cryptosystem

exploits that finding large primes is easy (we show how to do this later) and that factoring their product is (up till now) hard.

12 The RSA public-key cryptosystem

12.2 The RSA cryptosystem

exploits that finding large primes is easy (we show how to do this later) and that factoring their product is (up till now) hard.

generation of keys (requires a trusted agency)

1. select two large prime numpers p,q with $p \neq q$. [CLRS] suggest 1028 bits, but that was a long time ago. We show how to do this later.

2. compute $n = pq$

3. select a small odd integer e relatively prime to $\varphi(n)$

$$gcd(e,\varphi(n)) = \text{gcd}(e,(p-1)(q-1)) = 1$$

4. compute the multiplicative inverse d of e modulo $\varphi(n))$.

$$de \equiv 1 \bmod \varphi(n)$$

   Lemma 24 $\rightarrow$ d exists and is unique. Compute by

$$ext - eucl(e,\varphi(n)) = (1,x,y)$$

   Then

$$1 = xe + y\varphi(n) \quad , \quad xe \equiv 1 \bmod \varphi(n) \quad , \quad d = x \bmod \varphi(n)$$

12 The RSA public-key cryptosystem

12.2 The RSA cryptosystem

exploits that finding large primes is easy (we show how to do this later) and that factoring their product is (up till now) hard.

generation of keys (requires a trusted agency)

1. select two large prime numpers p,q with $p \neq q$. [CLRS] suggest 1028 bits, but that was a long time ago. We show how to do this later.

2. compute $n = pq$

3. select a small odd integer e relatively prime to $\varphi(n)$

$$gcd(e,\varphi(n)) = \gcd(e,(p-1)(q-1)) = 1$$

4. compute the multiplicative inverse d of e modulo $\varphi(n)$).

$$de \equiv 1 \bmod \varphi(n)$$

Lemma 24 $\rightarrow$ d exists and is unique. Compute by

$$ext - eucl(e,\varphi(n)) = (1,x,y)$$

Then

$$1 = xe + y\varphi(n) \quad , \quad xe \equiv 1 \bmod \varphi(n) \quad , \quad d = x \bmod \varphi(n)$$

5. publish public key

$$P = (e,n)$$

6. Inform only the user who requested key generation of the secret key

$$S = (d,n)$$

12 The RSA public-key cryptosystem

12.2 The RSA cryptosystem

exploits that finding large primes is easy (we show how to do this later) and that factoring their product is (up till now) hard.

generation of keys (requires a trusted agency)

1. select two large prime numpers p, q with $p \neq q$. [CLRS] suggest 1028 bits, but that was a long time ago. We show how to do this later.
2. compute $n = pq$
3. select a small odd integer e relatively prime to $\varphi(n)$

$$gcd(e, \varphi(n)) = \gcd(e, (p-1)(q-1)) = 1$$

4. compute the multiplicative inverse d of e modulo $\varphi(n))$.

$$de \equiv 1 \bmod \varphi(n)$$

Lemma 24 $\rightarrow$ d exists and is unique. Compute by

$$ext - eucl(e, \varphi(n)) = (1, x, y)$$

Then

$$1 = xe + y\varphi(n) \quad , \quad xe \equiv 1 \bmod \varphi(n) \quad , \quad d = x \bmod \varphi(n)$$

5. publish public key

$$P = (e, n)$$

6. Inform only the user who requested key generation of the secret key

$$S = (d, n)$$

coding and decoding:

- encoding M

$$P(M) = M^e \bmod n$$

- decoding C

$$S(C) = C^d \bmod n$$

12 The RSA public-key cryptosystem

12.2 The RSA cryptosystem

exploits that finding large primes is easy (we show how to do this later) and that factoring their product is (up till now) hard.

generation of keys (requires a trusted agency)

1. select two large prime numpers p, q with $p \neq q$. [CLRS] suggest 1028 bits, but that was a long time ago. We show how to do this later.

2. compute $n = pq$

3. select a small odd integer e relatively prime to $\varphi(n)$

$$gcd(e, \varphi(n)) = \gcd(e, (p-1)(q-1)) = 1$$

4. compute the multiplicative inverse d of e modulo $\varphi(n)$).

$$de \equiv 1 \bmod \varphi(n)$$

   Lemma 24 $\rightarrow d$ exists and is unique. Compute by

$$ext - eucl(e, \varphi(n)) = (1, x, y)$$

   Then

$$1 = xe + y\varphi(n) \quad , \quad xe \equiv 1 \bmod \varphi(n) \quad , \quad d = x \bmod \varphi(n)$$

5. publish public key

$$P = (e, n)$$

6. Inform only the user who requested key generation of the secret key

$$S = (d, n)$$

coding and decoding:

- encoding M

$$P(M) = M^e \bmod n$$

- decoding C

$$S(C) = C^d \bmod n$$

complexity of coding and decoding:

Let

$$\log e = O(1) \quad , \quad \log d \leq \beta \quad , \quad \log n \leq \beta$$

and assume mulitplication of β bit numbers with $O(\beta^2)$ bit operations. Then cost is

- for encoding M: $O(1)$ modular multiplications, $O(\beta^2)$ bit operations
- for decoding C: using repeated squaring $O(\beta)$ modular multiplications, $O(\beta^3)$ bit operations

12 The RSA public-key cryptosystem

generation of keys (requires a trusted agency)

1. select two large prime numpers p, q with $p \neq q$. [CLRS] suggest 1028 bits, but that was a long time ago. We show how to do this later.
2. compute $n = pq$
3. select a small odd integer e relatively prime to $\varphi(n)$

$$gcd(e, \varphi(n)) = \gcd(e, (p-1)(q-1)) = 1$$

4. compute the multiplicative inverse d of e modulo $\varphi(n))$.

$$de \equiv 1 \bmod \varphi(n)$$

Lemma 24 $\rightarrow$ d exists and is unique. Compute by

$$ext - eucl(e, \varphi(n)) = (1, x, y)$$

Then

$$1 = xe + y\varphi(n) \quad , \quad xe \equiv 1 \bmod \varphi(n) \quad , \quad d = x \bmod \varphi(n)$$

coding and decoding:

- encoding M

$$P(M) = M^e \bmod n$$

- decoding C

$$S(C) = C^d \bmod n$$

correctness of RSA:

Lemma 35. *Functions $P(\ )$ and $S(\ )$ defined above satisfy for any $M \in \mathbb{Z}_n$*

$$P(S(M)) = S(P(M)) = M$$

12 The RSA public-key cryptosystem

generation of keys (requires a trusted agency)

1. select two large prime numpers p,q with $p \neq q$. [CLRS] suggest 1028 bits, but that was a long time ago. We show how to do this later.

2. compute $n = pq$

3. select a small odd integer e relatively prime to $\varphi(n)$

$$gcd(e,\varphi(n)) = \gcd(e,(p-1)(q-1)) = 1$$

4. compute the multiplicative inverse d of e modulo $\varphi(n)$).

$$de \equiv 1 \bmod \varphi(n)$$

   Lemma 24 $\rightarrow$ d exists and is unique. Compute by

$$ext-eucl(e,\varphi(n)) = (1,x,y)$$

   Then

$$1 = xe + y\varphi(n) \quad , \quad xe \equiv 1 \bmod \varphi(n) \quad , \quad d = x \bmod \varphi(n)$$

coding and decoding:

- encoding M

$$P(M) = M^e \bmod n$$

- decoding C

$$S(C) = C^d \bmod n$$

correctness of RSA:

Lemma 35. *Functions $P(\ )$ and $S(\ )$ defined above satisfy for any $M \in \mathbb{Z}_n$*

$$P(S(M)) = S(P(M)) = M$$

- $$P(S(M)) = S(P(M)) = M^{ed} \bmod n \quad \text{(definitions)}$$

- $$ed \equiv 1 \bmod \varphi(n) \quad , \quad \varphi(n) = (p-1)(q-1)$$

  hence

$$ed = 1 + k(p-1)(q-1) \quad \text{for some } k \in \mathbb{Z}$$

12 The RSA public-key cryptosystem

generation of keys (requires a trusted agency)

1. select two large prime numpers p,q with $p \neq q$. [CLRS] suggest 1028 bits, but that was a long time ago. We show how to do this later.
2. compute $n = pq$
3. select a small odd integer e relatively prime to $\varphi(n)$
$$gcd(e,\varphi(n)) = \gcd(e,(p-1)(q-1)) = 1$$
4. compute the multiplicative inverse d of e modulo $\varphi(n)$).
$$de \equiv 1 \bmod \varphi(n)$$
Lemma 24 $\rightarrow d$ exists and is unique. Compute by
$$ext-eucl(e,\varphi(n)) = (1,x,y)$$
Then
$$1 = xe + y\varphi(n) \quad , \quad xe \equiv 1 \bmod \varphi(n) \quad , \quad d = x \bmod \varphi(n)$$

coding and decoding:

- encoding M
$$P(M) = M^e \bmod n$$
- decoding C
$$S(C) = C^d \bmod n$$

correctness of RSA:

Lemma 35. *Functions $P(\,)$ and $S(\,)$ defined above satisfy for any $M \in \mathbb{Z}_n$*
$$P(S(M)) = S(P(M)) = M$$

- $$P(S(M)) = S(P(M)) = M^{ed} \bmod n \quad \text{(definitions)}$$
- $$ed \equiv 1 \bmod \varphi(n) \quad , \quad \varphi(n) = (p-1)(q-1)$$
  hence
  $$ed = 1 + k(p-1)(q-1) \quad \text{for some } k \in \mathbb{Z}$$
- claim: $M^{ed} \equiv M \bmod p$ for all M.

  trivial for $M \equiv 0 \bmod p$. Thus assume $M \not\equiv 0 \bmod p$:
$$\begin{aligned} M^{ed} &\equiv M(M^{p-1})^{k(q-1)} \bmod p \\ &\equiv M(M \bmod p)^{p-1})^{k(q-1)} \bmod p \\ &\equiv M(1) \bmod p \quad \text{(lemma 30 , Fermat's theorem)} \\ &\equiv M \bmod p \end{aligned}$$

12 The RSA public-key cryptosystem

correctness of RSA:

Lemma 35. *Functions $P(\ )$ and $S(\ )$ defined above satisfy for any $M \in \mathbb{Z}_n$*

$$P(S(M)) = S(P(M)) = M$$

-
  $$P(S(M)) = S(P(M)) = M^{ed} \bmod n \quad \text{(definitions)}$$

-
  $$ed \equiv 1 \bmod \varphi(n) \quad , \quad \varphi(n) = (p-1)(q-1)$$

  hence

  $$ed = 1 + k(p-1)(q-1) \quad \text{for some } k \in \mathbb{Z}$$

- claim: $M^{ed} \equiv M \bmod p$ for all M.

  trivial for $M \equiv 0 \bmod p$. Thus assume $M \not\equiv 0 \bmod p$:

$$\begin{aligned}
M^{ed} &\equiv M(M^{p-1})^{k(q-1)} \bmod p \\
&\equiv M(M \bmod p)^{p-1})^{k(q-1)} \bmod p \\
&\equiv M(1) \bmod p \quad \text{(lemma 30 , Fermat's theorem)} \\
&\equiv M \bmod p
\end{aligned}$$

12 The RSA public-key cryptosystem

correctness of RSA:

Lemma 35. *Functions $P(\,)$ and $S(\,)$ defined above satisfy for any $M \in \mathbb{Z}_n$*

$$P(S(M)) = S(P(M)) = M$$

- $$P(S(M)) = S(P(M)) = M^{ed} \bmod n \quad \text{(definitions)}$$

- $$ed \equiv 1 \bmod \varphi(n) \quad , \quad \varphi(n) = (p-1)(q-1)$$

  hence

  $$ed = 1 + k(p-1)(q-1) \quad \text{for some } k \in \mathbb{Z}$$

- claim: $M^{ed} \equiv M \bmod p$ for all M.

  trivial for $M \equiv 0 \bmod p$. Thus assume $M \not\equiv 0 \bmod p$:

$$\begin{aligned} M^{ed} &\equiv M(M^{p-1})^{k(q-1)} \bmod p \\ &\equiv M(M \bmod p)^{p-1})^{k(q-1)} \bmod p \\ &\equiv M(1) \bmod p \quad \text{(lemma 30 , Fermat's theorem)} \\ &\equiv M \bmod p \end{aligned}$$

- similarly: $M^{ed} \equiv M \bmod q$ for all M

- Recall lemma 28, corollaray of Chinese remainder theorem:

  Let

  $$n = n_1 n_2 \dots n_k \quad , \quad i \neq j \to gcd(n_i, n_j) = 1 \text{ (pairwise relatively prime)}$$

  and

  $$a, x \in \mathbb{Z}$$

  then

  $$x \equiv a \bmod n_i \text{ for all } i \in [1:k] \quad \leftrightarrow \quad x \equiv a \bmod n$$

$$M^{ed} \equiv M \bmod n$$