RSA Cryptosystem

Math 4176

§5.3. The RSA Cryptosystem

The basic idea behind RSA Cryptosystem is that Bob can find three very large positive integers a, b, and n such that

$$(x^b)^a \equiv x \pmod{n}$$

for all x, and that even knowing b and n, it can be extremely difficult for the third party (Oscar) to find a.

§5.3. The RSA Cryptosystem

The basic idea behind RSA Cryptosystem is that Bob can find three very large positive integers a, b, and n such that

$$(x^b)^a \equiv x \pmod{n}$$

for all x, and that even knowing b and n, it can be extremely difficult for the third party (Oscar) to find a.

- Then Bob sends b and n (called public key) to Alice through a public channel.

§5.3. The RSA Cryptosystem

The basic idea behind RSA Cryptosystem is that Bob can find three very large positive integers a, b, and n such that

$$(x^b)^a \equiv x \pmod{n}$$

for all x, and that even knowing b and n, it can be extremely difficult for the third party (Oscar) to find a.

- Then Bob sends b and n (called public key) to Alice through a public channel.
- Alice encrypts the plaintext x by the encryption rule

$$e(x) \equiv x^b \pmod{n} = y$$

§5.3. The RSA Cryptosystem

The basic idea behind RSA Cryptosystem is that Bob can find three very large positive integers a, b, and n such that

$$(x^b)^a \equiv x \pmod{n}$$

for all x, and that even knowing b and n, it can be extremely difficult for the third party (Oscar) to find a.

- Then Bob sends b and n (called public key) to Alice through a public channel.
- Alice encrypts the plaintext x by the encryption rule

$$e(x) \equiv x^b \pmod{n} = y$$

- Upon receiving the ciphertext y, Bob decrypts by the decryption rule

$$d(y) \equiv y^a \pmod{n} = x$$

§5.3. The RSA Cryptosystem

Bob carries out the following to achieve this:

- Chooses two very large prime numbers p and q and $pq = n$ (keeps p and q secret).

§5.3. The RSA Cryptosystem

Bob carries out the following to achieve this:

- Chooses two very large prime numbers p and q and $pq = n$ (keeps p and q secret).
- Then $\phi(n) = (p-1)(q-1)$.

§5.3. The RSA Cryptosystem

Bob carries out the following to achieve this:

- Chooses two very large prime numbers p and q and $pq = n$ (keeps p and q secret).
- Then $\phi(n) = (p-1)(q-1)$.
- Chooses very large integer b such that $2 \leq b \leq \phi(n) - 1$ and b is co-prime to $\phi(n)$.

§5.3. The RSA Cryptosystem

Bob carries out the following to achieve this:

- Chooses two very large prime numbers p and q and $pq = n$ (keeps p and q secret).
- Then $\phi(n) = (p-1)(q-1)$.
- Chooses very large integer b such that $2 \leq b \leq \phi(n) - 1$ and b is co-prime to $\phi(n)$.
- Computes a such that $2 \leq a \leq \phi(n) - 1$ and $ab \equiv 1 \pmod{\phi(n)}$.

§5.3. The RSA Cryptosystem

Bob carries out the following to achieve this:

- Chooses two very large prime numbers p and q and $pq = n$ (keeps p and q secret).
- Then $\phi(n) = (p-1)(q-1)$.
- Chooses very large integer b such that $2 \leq b \leq \phi(n) - 1$ and b is co-prime to $\phi(n)$.
- Computes a such that $2 \leq a \leq \phi(n) - 1$ and $ab \equiv 1 \pmod{\phi(n)}$.
- Announces the pair (b, n) publicly which is called the public key.

§5.3. The RSA Cryptosystem

Bob carries out the following to achieve this:

- Chooses two very large prime numbers p and q and $pq = n$ (keeps p and q secret).
- Then $\phi(n) = (p-1)(q-1)$.
- Chooses very large integer b such that $2 \leq b \leq \phi(n) - 1$ and b is co-prime to $\phi(n)$.
- Computes a such that $2 \leq a \leq \phi(n) - 1$ and $ab \equiv 1 \pmod{\phi(n)}$.
- Announces the pair (b, n) publicly which is called the public key.
- Keeps (p, q, a) secret which form the private key.

§5.3. The RSA Cryptosystem

Now we define RSA cryptosystem formally as follows:

§5.3. The RSA Cryptosystem

Now we define RSA cryptosystem formally as follows:

RSA Cryptosystem: Let $n = pq$, where p and q are primes. Let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_n$, and define

$$\mathcal{K} = \{(n, p, q, a, b) : ab \equiv 1 \pmod{\phi(n)}\}$$

For $K = (n, p, q, a, b)$, define

$$e_K(x) \equiv x^b \pmod{n}$$

and

$$d_K(y) \equiv y^a \pmod{n}$$

where $x, y \in \mathbb{Z}_n$. The values n and b comprise the public key, and the values p, q, and a form the private key.

§5.3. The RSA Cryptosystem

Now let us look at the example we discussed in 5.1.

- In that example, Bob chooses $p = 23$, $q = 37$ and so $n = 851$.

§5.3. The RSA Cryptosystem

Now let us look at the example we discussed in 5.1.

- In that example, Bob chooses $p = 23$, $q = 37$ and so $n = 851$.
- Then $\phi(851) = (23 - 1)(37 - 1) = 22 \times 36 = 792$.

§5.3. The RSA Cryptosystem

Now let us look at the example we discussed in 5.1.

- In that example, Bob chooses $p = 23$, $q = 37$ and so $n = 851$.
- Then $\phi(851) = (23 - 1)(37 - 1) = 22 \times 36 = 792$.
- Bob's choice for b is 85 by checking that $\gcd(85, 792) = 1$.

§5.3. The RSA Cryptosystem

Now let us look at the example we discussed in 5.1.

- In that example, Bob chooses $p = 23$, $q = 37$ and so $n = 851$.
- Then $\phi(851) = (23 - 1)(37 - 1) = 22 \times 36 = 792$.
- Bob's choice for b is 85 by checking that $\gcd(85, 792) = 1$.
- Check by Division Algorithm that $85^{-1} \pmod{792} = 205$ and so 23, 37 and 205 form the private key (see next slide).

§5.3. The RSA Cryptosystem

Now let us look at the example we discussed in 5.1.

- In that example, Bob chooses $p = 23$, $q = 37$ and so $n = 851$.
- Then $\phi(851) = (23 - 1)(37 - 1) = 22 \times 36 = 792$.
- Bob's choice for b is 85 by checking that $\gcd(85, 792) = 1$.
- Check by Division Algorithm that $85^{-1} \pmod{792} = 205$ and so 23, 37 and 205 form the private key (see next slide).

Exercise: If Bob's public key is $(3233, 17)$, guess the private key.

§5.3. The RSA Cryptosystem

Now let us look at the example we discussed in 5.1.

- In that example, Bob chooses $p = 23$, $q = 37$ and so $n = 851$.
- Then $\phi(851) = (23 - 1)(37 - 1) = 22 \times 36 = 792$.
- Bob's choice for b is 85 by checking that $\gcd(85, 792) = 1$.
- Check by Division Algorithm that $85^{-1} \pmod{792} = 205$ and so 23, 37 and 205 form the private key (see next slide).

Exercise: If Bob's public key is $(3233, 17)$, guess the private key.

Note: Security of the RSA cryptosystem relies on the difficulty in finding p and q. Since 3233 is fairly a small number, one should be able to find p and q. In the real world, p and q are usually at least 512 bits long each!!

§5.3 Division Algorithm

We get the following (Division Algorithm) table:

u_1	v_1	u_2	v_2	u_3	v_3	q
1	0	0	1	792	85	0
0	1	1	-9	85	27	9
1	-3	-9	28	27	4	3
-3	19	28	-177	4	3	6
19	-22	-177	205	3	1	1
-22	*	205	*	1	0	3

§5.3 Division Algorithm

We get the following (Division Algorithm) table:

u_1	v_1	u_2	v_2	u_3	v_3	q
1	0	0	1	792	85	0
0	1	1	-9	85	27	9
1	-3	-9	28	27	4	3
-3	19	28	-177	4	3	6
19	-22	-177	205	3	1	1
-22	*	205	*	1	0	3

Now $\gcd(792, 85) = 1$ and $(792)(-22) + (85)(205) = 1$.

§5.3 Division Algorithm

We get the following (Division Algorithm) table:

u_1	v_1	u_2	v_2	u_3	v_3	q
1	0	0	1	792	85	0
0	1	1	-9	85	27	9
1	-3	-9	28	27	4	3
-3	19	28	-177	4	3	6
19	-22	-177	205	3	1	1
-22	*	205	*	1	0	3

Now $\gcd(792, 85) = 1$ and $(792)(-22) + (85)(205) = 1$.

So $85^{-1} \pmod{792} = 205$.

§5.3. The RSA Cryptosystem

Remark: The best known factorization algorithm today requires long computer time (in months even for the fastest computers) to factor large number n. It is also unknown whether factoring n is the fastest way to attack RSA. In other words, there may exist a faster and nicer algorithm to break RSA in a different manner that remains to be discovered. When one finds it, RSA algorithm may become obsolete.

§5.3. The RSA Cryptosystem

Remark: The best known factorization algorithm today requires long computer time (in months even for the fastest computers) to factor large number n. It is also unknown whether factoring n is the fastest way to attack RSA. In other words, there may exist a faster and nicer algorithm to break RSA in a different manner that remains to be discovered. When one finds it, RSA algorithm may become obsolete.

Anyway, in order to RSA make sense, we need the following proposition:

§5.3. The RSA Cryptosystem

Remark: The best known factorization algorithm today requires long computer time (in months even for the fastest computers) to factor large number n. It is also unknown whether factoring n is the fastest way to attack RSA. In other words, there may exist a faster and nicer algorithm to break RSA in a different manner that remains to be discovered. When one finds it, RSA algorithm may become obsolete.

Anyway, in order to RSA make sense, we need the following proposition:

Proposition: The encryption and decryption in RSA cryptosystem are inverse operations.

§5.3. The RSA Cryptosystem

Remark: The best known factorization algorithm today requires long computer time (in months even for the fastest computers) to factor large number n. It is also unknown whether factoring n is the fastest way to attack RSA. In other words, there may exist a faster and nicer algorithm to break RSA in a different manner that remains to be discovered. When one finds it, RSA algorithm may become obsolete.

Anyway, in order to RSA make sense, we need the following proposition:

Proposition: The encryption and decryption in RSA cryptosystem are inverse operations.

Proof: Recall that

$$ba \equiv 1 \pmod{\phi(n)} \implies ba - 1 = h\phi(n) = h(p-1)(q-1).$$

§5.3. The RSA Cryptosystem

Remark: The best known factorization algorithm today requires long computer time (in months even for the fastest computers) to factor large number n. It is also unknown whether factoring n is the fastest way to attack RSA. In other words, there may exist a faster and nicer algorithm to break RSA in a different manner that remains to be discovered. When one finds it, RSA algorithm may become obsolete.

Anyway, in order to RSA make sense, we need the following proposition:

Proposition: The encryption and decryption in RSA cryptosystem are inverse operations.

Proof: Recall that

$$ba \equiv 1 \pmod{\phi(n)} \implies ba - 1 = h\phi(n) = h(p-1)(q-1).$$

Now we need to show that $x^{ba} \equiv x \pmod{n}$ for all $x \in \mathbb{Z}_n$.

§5.3. The RSA Cryptosystem

- In other words, we have to show that $x^{ba} - x$ is multiple of pq.

§5.3. The RSA Cryptosystem

- In other words, we have to show that $x^{ba} - x$ is multiple of pq.
- The trick is to show that $x^{ba} - x$ is multiple of each of the different primes p and q.

§5.3. The RSA Cryptosystem

- In other words, we have to show that $x^{ba} - x$ is multiple of pq.
- The trick is to show that $x^{ba} - x$ is multiple of each of the different primes p and q.
- We will show that $x^{ba} - x$ is multiple of p and the proof for other prime is similar.

§5.3. The RSA Cryptosystem

- In other words, we have to show that $x^{ba} - x$ is multiple of pq.
- The trick is to show that $x^{ba} - x$ is multiple of each of the different primes p and q.
- We will show that $x^{ba} - x$ is multiple of p and the proof for other prime is similar.
- If x is a multiple of p, then the proof is immediate.

§5.3. The RSA Cryptosystem

- In other words, we have to show that $x^{ba} - x$ is multiple of pq.
- The trick is to show that $x^{ba} - x$ is multiple of each of the different primes p and q.
- We will show that $x^{ba} - x$ is multiple of p and the proof for other prime is similar.
- If x is a multiple of p, then the proof is immediate.
- Otherwise, $\gcd(x, p) = 1$ and let $x \equiv \alpha \pmod{p}$ for a nonzero $\alpha \in \mathbb{Z}_p$.

§5.3. The RSA Cryptosystem

- In other words, we have to show that $x^{ba} - x$ is multiple of pq.
- The trick is to show that $x^{ba} - x$ is multiple of each of the different primes p and q.
- We will show that $x^{ba} - x$ is multiple of p and the proof for other prime is similar.
- If x is a multiple of p, then the proof is immediate.
- Otherwise, $\gcd(x, p) = 1$ and let $x \equiv \alpha \pmod{p}$ for a nonzero $\alpha \in \mathbb{Z}_p$.
- Therefore, by Fermat's theorem, we have $x^{p-1} \equiv \alpha^{p-1} \equiv 1 \pmod{p}$.

§5.3. The RSA Cryptosystem

- In other words, we have to show that $x^{ba} - x$ is multiple of pq.
- The trick is to show that $x^{ba} - x$ is multiple of each of the different primes p and q.
- We will show that $x^{ba} - x$ is multiple of p and the proof for other prime is similar.
- If x is a multiple of p, then the proof is immediate.
- Otherwise, $\gcd(x, p) = 1$ and let $x \equiv \alpha \pmod{p}$ for a nonzero $\alpha \in \mathbb{Z}_p$.
- Therefore, by Fermat's theorem, we have $x^{p-1} \equiv \alpha^{p-1} \equiv 1 \pmod{p}$.
- Thus

$$x^{ba} = x^{1+h(p-1)(q-1)} = x^1(x^{p-1})^{h(q-1)} \equiv x(1)^{h(q-1)} \equiv x \pmod{p}$$

§5.3. Square-and-Multiply Algorithm

In order to encrypt the message, Alice needs to calculate $x^b \pmod{n}$. But the existing calculators will not be able to handle large numbers to perform such modular arithmetic.

§5.3. Square-and-Multiply Algorithm

In order to encrypt the message, Alice needs to calculate $x^b \pmod{n}$. But the existing calculators will not be able to handle large numbers to perform such modular arithmetic.

The well-known Square-and-Multiply Algorithm reduces the computation of $x^b \pmod{n}$ to at most 2ℓ modular multiplications, where ℓ is the number of bits in the binary representation of b.

§5.3. Square-and-Multiply Algorithm

In order to encrypt the message, Alice needs to calculate $x^b \pmod{n}$. But the existing calculators will not be able to handle large numbers to perform such modular arithmetic.

The well-known Square-and-Multiply Algorithm reduces the computation of $x^b \pmod{n}$ to at most 2ℓ modular multiplications, where ℓ is the number of bits in the binary representation of b.

Suppose that $b = \sum_{i=0}^{\ell-1} b_i 2^i$ where $b_i = 0$ or 1 for $0 \leq i \leq \ell - 1$.

§5.3. Square-and-Multiply Algorithm

In order to encrypt the message, Alice needs to calculate $x^b \pmod{n}$. But the existing calculators will not be able to handle large numbers to perform such modular arithmetic.

The well-known Square-and-Multiply Algorithm reduces the computation of $x^b \pmod{n}$ to at most 2ℓ modular multiplications, where ℓ is the number of bits in the binary representation of b.

Suppose that $b = \sum_{i=0}^{\ell-1} b_i 2^i$ where $b_i = 0$ or 1 for $0 \leq i \leq \ell - 1$.

Square-and-Multiply Algorithm (x, b, n)**:**

- Step 1: $z \leftarrow 1$
- Step 2: For $i = \ell - 1$ down to 0, do
  - Step 2.1: $z \leftarrow z^2 \pmod{n}$
  - Step 2.2: If $b_i = 1$, then $z \leftarrow z \times x \pmod{n}$
- Step 3: Return z

§5.3. Square-and-Multiply Algorithm

For example, the binary representation of 41 is 101001. So the Square-and-multiply algorithm for $18^{41} \pmod{26}$ yields:

i	b_i			z
				1
5	1	$1^2 \times 18$	18	18
4	0	18^2	324	12
3	1	$12^2 \times 18$	2592	18
2	0	18^2	324	12
1	0	12^2	144	14
0	1	$14^2 \times 18$	3528	18

§5.3. Square-and-Multiply Algorithm

For example, the binary representation of 41 is 101001. So the Square-and-multiply algorithm for 18^{41} (mod 26) yields:

i	b_i			z
				1
5	1	$1^2 \times 18$	18	18
4	0	18^2	324	12
3	1	$12^2 \times 18$	2592	18
2	0	18^2	324	12
1	0	12^2	144	14
0	1	$14^2 \times 18$	3528	18

Notice that we have $\ell = 6$ squaring and $3(\leq 6)$ multiplication in the above example.

§5.3. Square-and-Multiply Algorithm

For example, the binary representation of 85 is 1010101. So the Square-and-multiply algorithm for $583^{85} \pmod{851}$ yields:

i	b_i		z
			1
6	1	$1^2 \times 583$	583
5	0	583^2	340
4	1	$340^2 \times 583$	706
3	0	706^2	601
2	1	$601^2 \times 583$	233
1	0	233^2	676
0	1	$676^2 \times 583$	395

§5.3. Square-and-Multiply Algorithm

For example, the binary representation of 85 is 1010101. So the Square-and-multiply algorithm for $583^{85} \pmod{851}$ yields:

i	b_i		z
			1
6	1	$1^2 \times 583$	583
5	0	583^2	340
4	1	$340^2 \times 583$	706
3	0	706^2	601
2	1	$601^2 \times 583$	233
1	0	233^2	676
0	1	$676^2 \times 583$	395

Notice that we have $\ell = 7$ squaring and $4(\leq 7)$ multiplication in the above example.

§5.3. Implementing RSA

- If one wants a very secured RSA cryptosystem, it is necessary that $n = pq$ must be very large enough that factoring it will be computationally infeasible.

§5.3. Implementing RSA

- If one wants a very secured RSA cryptosystem, it is necessary that $n = pq$ must be very large enough that factoring it will be computationally infeasible.
- Currently available factoring algorithms are able to factor numbers having up to 512 bits in their binary representation.

§5.3. Implementing RSA

- If one wants a very secured RSA cryptosystem, it is necessary that $n = pq$ must be very large enough that factoring it will be computationally infeasible.
- Currently available factoring algorithms are able to factor numbers having up to 512 bits in their binary representation.
- So in order to be safe, one should choose each prime p and q to be at least 512-bits primes and hence n will be at least 1024-bit modulus.

§5.3. Implementing RSA

- If one wants a very secured RSA cryptosystem, it is necessary that $n = pq$ must be very large enough that factoring it will be computationally infeasible.
- Currently available factoring algorithms are able to factor numbers having up to 512 bits in their binary representation.
- So in order to be safe, one should choose each prime p and q to be at least 512-bits primes and hence n will be at least 1024-bit modulus.
- We will now give an example an integer n that is used in RSA cryptosystem.

§5.6. Factorization Algorithms

RSA-2048 is the largest RSA number with 617 digits (2048 bits), carried the largest cash prize of $200,000, and yet to be factored:

RSA-2048 =
251959084756578934940271832400483985714292821262040320277771
378360436620207075955562640185258807844069182906412495150821
892985591491761845028084891200728449926873928072877767359714
183472702618963750149718246911650776133798590957000973304597
488084284017974291006424586918171951187461215151726546322822
168699875491824224336372590851418654620435767984233871847744
479207399342365848238242811981638150106748104516603773060562
016196762561338441436038339044149526344321901146575444541784
2402092461651572335077870774981712577246796292638635637328991
2154831438167899885040445364023527381951378636564391212010397
122822120720357