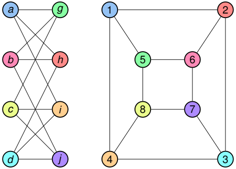
Week 9: Public-Key Cryptography (Textbook Chapter 10)

Prof. Michael Gastpar

Slides by Prof. M. Gastpar and Prof. em. B. Rimoldi

Spring Semester 2025



$f(a) = 1$
$f(b) = 6$
$f(c) = 8$
$f(d) = 3$
$f(g) = 5$
$f(h) = 2$
$f(i) = 4$
$f(j) = 7$

LAST WEEK

$(G, \star)$: commutative group with n elements.

- order(a) must divide n

$\Rightarrow \quad a^{n} = e \qquad a^{\ell n} = e, \forall \ell \in \mathbb{Z}^{+}$

$\Rightarrow \quad a^{n+1} = a \qquad a^{\ell n+1} = a,$ −"−

$\Rightarrow$ inverse of a is a^{n-1}

LAST WEEK

$(\mathbb{Z}/m\mathbb{Z}^*, \cdot)$: commutative group with $\phi(m)$ elements.

- order (a) must divide $\phi(m)$

$\Rightarrow \quad a^{\phi(m)} = e$

$\Rightarrow \quad a^{\phi(m)+1} = a$

$\Rightarrow$ inverse of a is $a^{\phi(m)-1}$

SPECIAL GROUP :

$$G = \{ g, g^2, g^3, \ldots, g^n \}$$

CALLED **CYCLIC GROUP** .

SNEAK PEAK OF RSA

Alice

Plaintext

t

Bob

(m, d)

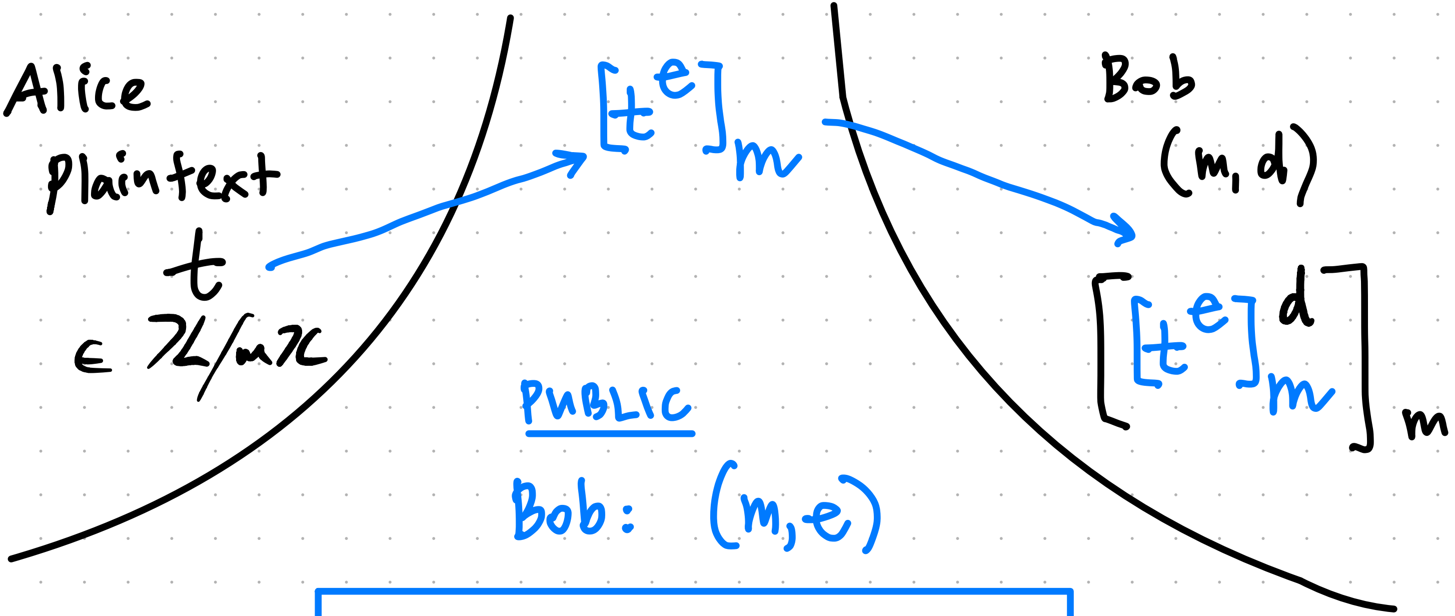
PUBLIC

Bob: (m, e)

all operations are in

$(\mathbb{Z}/m\mathbb{Z}, \cdot)$

SNEAK PEAK OF RSA



Alice

Plaintext

t

$\in \mathbb{Z}/m\mathbb{Z}$

$[t^e]_m$

Bob

(m, d)

$[[t^e]_m^d]_m$

<u>PUBLIC</u>

Bob: (m, e)

all operations are in

$(\mathbb{Z}/m\mathbb{Z}, \cdot)$

$$\left[[t^{e}]_{m}^{d} \right]_{m} = \left[t^{ed} \right]_{m} = [t]_{m}$$

can we make this happen?

$$\boxed{\text{Let us select } m = p, \text{ a } \underline{\text{prime}}.}$$

Consider $\underbrace{\left(\mathbb{Z}/p\mathbb{Z}^*, \cdot\right)}$.

cardinality $\phi(p) = p-1$.

We have seen that

$$[t^{\ell\phi(p)}]_p = [1]_p$$

holds for $\underline{\text{all}}$ $t \in \mathbb{Z}/p\mathbb{Z}^*$.

Hence $$[t^{\ell\phi(p)+1}]_p = [t]_p$$

Let us select $m = p$, a prime.

Moreover, for $t = [0]$, we also have

$$[t^{\ell \phi(p) + 1}]_p = [t]_p .$$

Hence, for all $t \in \mathbb{Z}/p\mathbb{Z}$: no star!

$$[t^{\ell \phi(p)+1}]_p = [t]_p$$

$$\boxed{\text{Let us select } m = p, \text{ a } \underline{\text{prime}}.}$$

We want

$$[t^{ed}]_p = [t]_p .$$

Hence, select e, d such that

$$ed = \ell\,\phi(p) + 1$$

$$\forall\, a, b \;\; \exists\, u, v : \;\; au + bv = \gcd(a, b)$$

Let us select $m = p$, a prime.

We want

$$[t^{ed}]_p = [t]_p .$$

Hence, select e, d such that

$$ed = \ell \, \phi(p) + 1$$

By Bézout, if e and $\phi(p)$ are coprime, then d and ℓ exist to satisfy this!

BuT: Is this a good crypto system?

Bob publishes (p, e)

$$\phi(p) = p - 1$$

BUT: Is this a good crypto system?

At the cost of Extended Euclid, anyone can find d.

→ not secure at all!

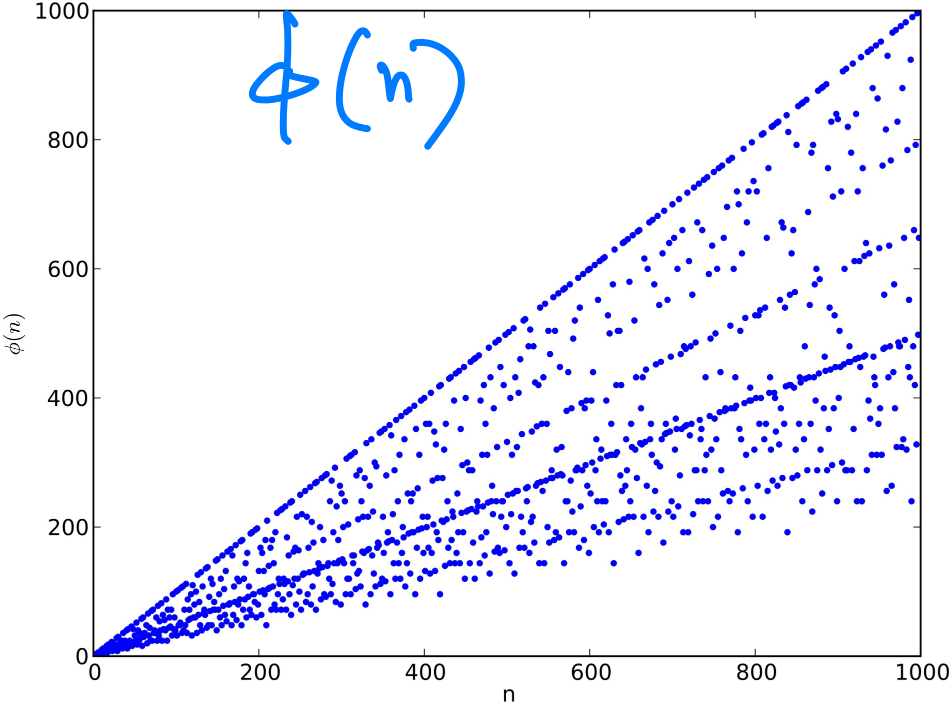
$$[t^{\ell\phi(m)+1}]_m = [t]_m$$

→ Dream:

Even if I tell you m, you can't find $\phi(m)$.

CLIFFORD COCKS

1973



$\phi(n)$

⟶ RSA, 1978

PROPOSAL:

$m = pq$, p, q two primes.

$\phi(m) = (p-1)(q-1)$

CANNOT FIND $\phi(m)$ UNLESS YOU KNOW p AND q !

$\rightsquigarrow$ PRIME FACTORIZATION IS HARD.

SO: WE NEED TO UNDERSTAND

$$(\mathbb{Z}/m\mathbb{Z}, \cdot)$$

no star!

WHEN $m = pq$.

$p = 3, \quad q = 5 \quad \Rightarrow \quad pq = 15$

$$\begin{array}{c|ccccc}
 & 0 & 1 & 2 & 3 & 4 \\
\hline
0 & 0^{15} & 6 & 12 & 3^{18} & 9 \\
1 & 10 & 1^{16} & 7 & 13 & 4 \\
2 & 5 & 11 & 2^{17} & 8 & 14
\end{array}$$

$[8] \rightarrow (2, 3) \qquad [8]_3 = 2 \qquad [8]_5 = 3$

Outline

Introduction and Organization

Entropy and Data Compression

Cryptography

- One-Time Pad, Perfect Secrecy, Public-Key (Diffie-Hellman)
- Rudiments of Number Theory
- Modular Arithmetic
- Commutative Groups
- Public-Key Cryptography
- Summary of Chapter 2

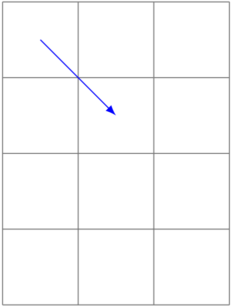
Channel Coding

The Chinese Remainders Theorem

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.

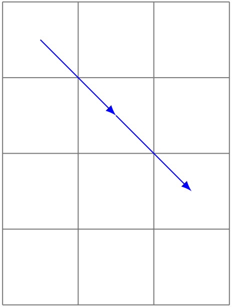
The Chinese Remainders Theorem

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



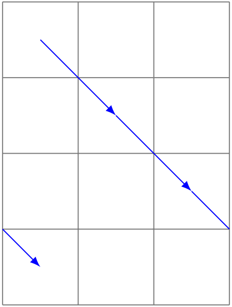
THE CHINESE REMAINDERS THEOREM

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



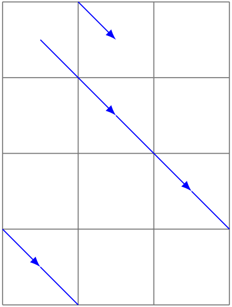
THE CHINESE REMAINDERS THEOREM

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



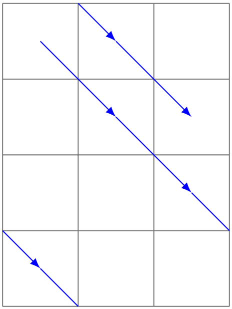
THE CHINESE REMAINDERS THEOREM

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



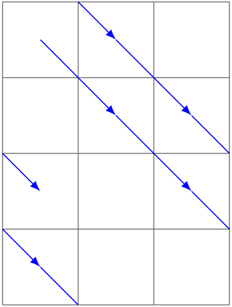
The Chinese Remainders Theorem

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



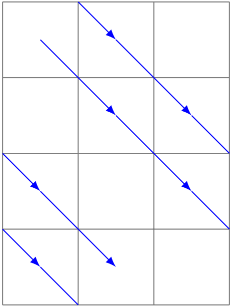
The Chinese Remainders Theorem

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



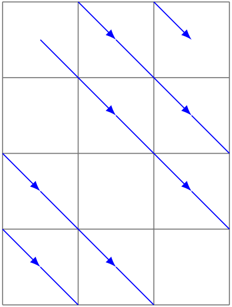
The Chinese Remainders Theorem

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



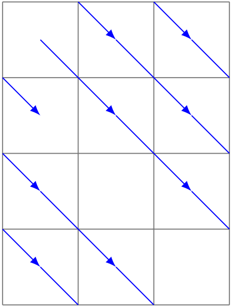
The Chinese Remainders Theorem

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



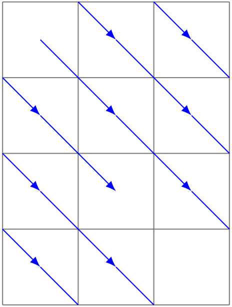
THE CHINESE REMAINDERS THEOREM

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



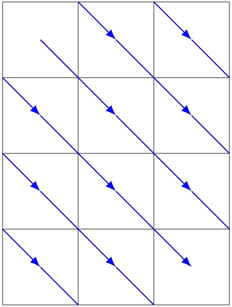
THE CHINESE REMAINDERS THEOREM

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



The Chinese Remainders Theorem

- Consider filling a table, going down diagonals, following the "torus rule"
- i.e., you start on the main diagonal . . .
- and when you drop off from an edge, you re-enter from the opposite edge.



Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0		

EXAMPLE (FILLED TABLE)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0		
	1	

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0		
	1	
		2

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0		
	1	
		2
3		

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0	4	
	1	
		2
3		

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0	4	
	1	5
		2
3		

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0	4	
	1	5
6		2
3		

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0	4	
	1	5
6		2
3	7	

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0	4	8
	1	5
6		2
3	7	

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0	4	8
9	1	5
6		2
3	7	

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0	4	8
9	1	5
6	10	2
3	7	

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0	4	8
9	1	5
6	10	2
3	7	11

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0,12	4	8
9	1	5
6	10	2
3	7	11

Example (filled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 23, \ldots$

0,12	4,16	8,20
9,21	1,13	5,17
6,18	10,22	2,14
3,15	7,19	11,23

Example (unfilled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 7, \ldots$

In this case, the table will never be filled.

Question: under which conditions will the table eventually fill?

Example (unfilled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 7, \ldots$

0			

In this case, the table will never be filled.

Question: under which conditions will the table eventually fill?

EXAMPLE (UNFILLED TABLE)

Consider filling the table with the integers $0, 1, 2, \ldots, 7, \ldots$

0			
	1		

In this case, the table will never be filled.

Question: under which conditions will the table eventually fill?

Example (unfilled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 7, \ldots$

0		2	
	1		

In this case, the table will never be filled.

Question: under which conditions will the table eventually fill?

Example (unfilled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 7, \ldots$

0		2	
	1		3

In this case, the table will never be filled.

Question: under which conditions will the table eventually fill?

Example (unfilled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 7, \ldots$

0,4		2	
	1		3

In this case, the table will never be filled.

Question: under which conditions will the table eventually fill?

Example (unfilled table)

Consider filling the table with the integers $0, 1, 2, \ldots, 7, \ldots$

0,4		2,6	
	1,5		3,7

In this case, the table will never be filled.

Question: under which conditions will the table eventually fill?

Mathematical formulation:

- we have $m_1 m_2$ numbers to be placed in $m_1 \times m_2$ drawers (m_1 rows and m_2 columns, matrix convention);
- we can see the numbers as elements of $\mathbb{Z}/m_1 m_2\mathbb{Z}$;
- and we can index the drawers with the elements of $\mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z}$.

The placing

$$[k]_{m_1 m_2} \mapsto \big([k]_{m_1}, [k]_{m_2}\big)$$

can be seen as the action of a map

$$\psi : \mathbb{Z}/m_1 m_2\mathbb{Z} \to \mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z}.$$

Is this map onto? (In which case it is a bijection).

Example ($m_1 = 3$, $m_2 = 4$)

	$[0]_4$	$[1]_4$	$[2]_4$	$[3]_4$
$[0]_3$	$[0]_{12}$	$[9]_{12}$	$[6]_{12}$	$[3]_{12}$
$[1]_3$	$[4]_{12}$	$[1]_{12}$	$[10]_{12}$	$[7]_{12}$
$[2]_3$	$[8]_{12}$	$[5]_{12}$	$[2]_{12}$	$[11]_{12}$

map ψ

$$
\begin{aligned}
&[0]_{12} \mapsto ([0]_3, [0]_4)\\
&[1]_{12} \mapsto ([1]_3, [1]_4)\\
&[2]_{12} \mapsto ([2]_3, [2]_4)\\
&[3]_{12} \mapsto ([3]_3, [3]_4) = ([0]_3, [3]_4)\\
&\quad\vdots\\
&[7]_{12} \mapsto ([7]_3, [7]_4) = ([1]_3, [3]_4)\\
&[8]_{12} \mapsto ([8]_3, [8]_4) = ([2]_3, [0]_4)\\
&\quad\vdots
\end{aligned}
$$

Theorem (Chinese Remainders)

If m_1 and m_2 are **relatively prime**, the map ψ defined by

$$\psi : \mathbb{Z}/m_1 m_2\mathbb{Z} \to \mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z}$$

$$[k]_{m_1 m_2} \mapsto ([k]_{m_1}, [k]_{m_2})$$

is

1. bijective
2. an isomorphism with respect to "$+$" and with respect to "$\cdot$".

If m_1 and m_2 are **not relatively prime**, ψ is neither onto nor one-to-one.

Example (bijective yes/no)

	0	1	2
0	0	4	8
1	9	1	5
2	6	10	2
3	3	7	11

	0	1	2	3
0	0,4		2,6	
1		1,5		3,7

$\gcd(m_1, m_2) = \gcd(4, 3) = 1$

$\Rightarrow$ bijective ψ

$\gcd(m_1, m_2) = \gcd(2, 4) \neq 1$

$\Rightarrow \psi$ is neither surjective nor injective

Example (Bijection ⇒ isomorphism)

$\mathbb{Z}/12\mathbb{Z}$

0	1	2	3	4	5	6	7	8	9	10	11

$\mathbb{Z}/4\mathbb{Z} \times \mathbb{Z}/3\mathbb{Z}$

	0	1	2
0	0	4	8
1	9	1	5
2	6	10	2
3	3	7	11

isomorphism w.r.t. "$+$":

$$[8]_{12} + [10]_{12} = [6]_{12}$$

$$([0]_4, [2]_3) + ([2]_4, [1]_3) = ([2]_4, [0]_3)$$

isomorphism w.r.t. "$\cdot$":

$$[8]_{12} \cdot [2]_{12} = [4]_{12}$$

$$([0]_4, [2]_3) \cdot ([2]_4, [2]_3) = ([0]_4, [1]_3)$$

Proof that: m_1 and m_2 coprime $\Rightarrow$ ψ is bijective.

First we prove that ψ is one-to-one:

- suppose $[k]_{m_1} = [k']_{m_1}$ and $[k]_{m_2} = [k']_{m_2}$;
- $\Leftrightarrow$ m_1 and m_2 divide $(k - k')$;
- because m_1 and m_2 have no common factors, $m_1 m_2$ divides $(k - k')$;
- hence $[k]_{m_1 m_2} = [k']_{m_1 m_2}$;
- the map is one-to-one.

The function is bijective because it is one-to-one and the co-domain has the same cardinality as the domain. □

PROOF OF ISOMORPHISM W.R.T. $+$

Claim: For all $k, k' \in \mathbb{Z}/m_1m_2\mathbb{Z}$:

$$\psi(k \underset{\text{in } \mathbb{Z}/m_1m_2\mathbb{Z}}{+} k') = \psi(k) \underset{\text{in } \mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z}}{+} \psi(k')$$

Proof:

$$\psi(k+k') = \left([k+k']_{m_1} , [k+k']_{m_2} \right)$$

$$= \left([k]_{m_1} + [k']_{m_1} , [k]_{m_2} + [k']_{m_2} \right)$$

$$\psi(k) = \left([k]_{m_1} , [k]_{m_2} \right)$$

$$\psi(k') = \left([k']_{m_1} , [k']_{m_2} \right)$$

Proof that: m_1 and m_2 coprime $\Rightarrow$ Isomorphism w.r.t. "$+$"

By the definition of ψ and the modulo arithmetic,

$$\begin{aligned}[k+l]_{m_1 m_2} \mapsto & ([k+l]_{m_1}, [k+l]_{m_2}) \\ & ([k]_{m_1} + [l]_{m_1}, [k]_{m_2} + [l]_{m_2}) \\ & ([k]_{m_1}, [k]_{m_2}) + ([l]_{m_1}, [l]_{m_2})\end{aligned}$$

□

Proof that: m_1 and m_2 coprime $\Rightarrow$ Isomorphism w.r.t. "$\cdot$"

By the definition of ψ and the modulo arithmetic,

$$\begin{aligned}
[k \cdot l]_{m_1 m_2} \mapsto & ([k \cdot l]_{m_1}, [k \cdot l]_{m_2}) \\
& ([k]_{m_1} \cdot [l]_{m_1}, [k]_{m_2} \cdot [l]_{m_2}) \\
& ([k]_{m_1}, [k]_{m_2}) \cdot ([l]_{m_1}, [l]_{m_2}).
\end{aligned}$$

□

Proof that: $\gcd(m_1, m_2) \neq 1 \;\Rightarrow\;$ neither One-To-One nor Onto

We show that if $m_1 = aq$ and $m_2 = bq$, the map is not one-to-one.

- Consider $k = abq$
- Properties of k: $\quad 0 < k < m_1 m_2 = abq^2; \quad k = m_1 b; \quad k = m_2 a$
- Hence ψ maps $[k]_{m_1 m_2} \mapsto ([0]_{m_1}, [0]_{m_2})$
- But it maps also $[0]_{m_1 m_2} \mapsto ([0]_{m_1}, [0]_{m_2})$
- ψ is not one-to-one

Since the co-domain has the same cardinality as the domain, the map is not onto either. □

$$12 = 3 \cdot 4$$

EXAMPLE

Find all solutions of $x^3 = [7]_{12}$, $x \in \mathbb{Z}/12\mathbb{Z}$.

↕

$$\begin{array}{l}(x_1, x_2)^3 = (3, 1)\\ \quad \| \\ (x_1^3, x_2^3)\end{array}$$

$$x_1 \in \mathbb{Z}/4\mathbb{Z}$$

$$x_2 \in \mathbb{Z}/3\mathbb{Z}$$

$$x_1^3 = 3$$

$$x_2^3 = 1$$

SOLUTION

Since $12 = 3 \times 4$ and $\gcd(3, 4) = 1$, $\psi : \mathbb{Z}/12\mathbb{Z} \to \mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/4\mathbb{Z}$ is an isomorphism w.r.t. $+$ and $\times$.

Instead of solving $x^3 = [7]_{12}$, we can work in $\mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/4\mathbb{Z}$ and solve

$$(x_1, x_2)^3 = ([7]_3, [7]_4).$$

Same as solving

$$\begin{cases} x_1^3 = [1]_3 & x \in \mathbb{Z}/3\mathbb{Z} \\ x_2^3 = [3]_4 & x \in \mathbb{Z}/4\mathbb{Z}. \end{cases}$$

The solution (by inspection) is

$$\begin{cases} x_1 = [1]_3 \\ x_2 = [3]_4 \end{cases} \Rightarrow x = [7]_{12}.$$

Example (recalling some old stuff)

Recall the following: $([x]_{96})^2 = [0]_{96} \quad \not\Rightarrow [x]_{96} = [0]_{96}$.

Reason:

- $96 = 2^5 \cdot 3$
- We can find a number, such as $k = 2^3 \cdot 3$, which fulfills $k < 96$ and k^2 is a multiple of 96.
- Hence $[k]_{96} \neq [0]_{96}$ and $[k^2]_{96} = [0]_{96}$.

However, for a prime modulus, like 97: $([x]_{97})^2 = [0]_{97} \quad \Rightarrow [x]_{97} = [0]_{97}$.

Reason:

- If $[x]_{97} = 0$ we are done. Otherwise, $\Rightarrow [x]_{97}$ has an inverse;
- $\Rightarrow [x]_{97} \cdot [x]_{97} = [0]_{97}$ implies $[x]_{97} = [0]_{97}$.

$$77 = 7 \cdot 11.$$

Example (How about this one)

$([x]_{77})^2 = [0]_{77}$ implies $[x]_{77} = [0]_{77}$?

Solution

- $77 = 7 \cdot 11$
- $(\mathbb{Z}/77\mathbb{Z}, \cdot)$ is isomorphic to $(\mathbb{Z}/7\mathbb{Z}, \cdot) \times (\mathbb{Z}/11\mathbb{Z}, \cdot)$
- Hence $[x]_{77} \cdot [x]_{77} = [0]_{77}$

  $\Leftrightarrow ([x]_7, [x]_{11}) \cdot ([x]_7, [x]_{11}) = ([0]_7, [0]_{11})$

  $\Leftrightarrow ([x]_7 \cdot [x]_7, [x]_{11} \cdot [x]_{11}) = ([0]_7, [0]_{11})$
- We are back to the prime modulus case

  which implies $[x]_7 = [0]_7$ and $[x]_{11} = [0]_{11}$

  which implies $[x]_{77} = [0]_{77}$.

CONCLUSION

If $\gcd(m_1, m_2) = 1$,

the Chinese remainders theorem says that

we can calculate in $\mathbb{Z}/m_1 m_2\mathbb{Z}$

or in $\mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z}$

whichever is more convenient.

THE INVERSE MAP

The map to

$$\psi : \mathbb{Z}/m_1 m_2\mathbb{Z} \to \mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z}$$
$$[k]_{m_1 m_2} \mapsto ([k]_{m_1}, [k]_{m_2}),$$

is easy to compute.

How about the inverse map?

$$\psi^{-1} : \mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z} \to \mathbb{Z}/m_1 m_2\mathbb{Z}.$$

We use the extended Euclid's algorithm to find integers u and v such that

$$1 = \gcd(m_1, m_2) = m_1 u + m_2 v.$$

Let

$$\begin{aligned} a &= m_2 v, \\ b &= m_1 u. \end{aligned}$$

Notice that

$$\begin{aligned} [a]_{m_2} &= [m_2 v]_{m_2} = [0]_{m_2}, \\ [a]_{m_1} &= [m_2 v]_{m_1} = [1 - m_1 u]_{m_1} = [1]_{m_1}. \end{aligned}$$

Similarly,

$$\begin{aligned} [b]_{m_1} &= [m_1 u]_{m_1} = [0]_{m_1}, \\ [b]_{m_2} &= [m_1 u]_{m_2} = [1 - m_2 v]_{m_2} = [1]_{m_2}. \end{aligned}$$

Hence, for any integers k_1 and k_2,

$$\psi\big([ak_1 + bk_2]_{m_1 m_2}\big) = ([k_1]_{m_1}, [k_2]_{m_2}),$$

implying that

$$\begin{aligned}\psi^{-1} : \mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z} &\to \mathbb{Z}/m_1 m_2\mathbb{Z}\\ ([k_1]_{m_1}, [k_2]_{m_2}) &\mapsto ([ak_1 + bk_2]_{m_1 m_2})\end{aligned}$$

is the inverse map.

$m_1 = 3$, $m_2 = 5$ $\Rightarrow$ $m_1 m_2 = 15$

	0	1	2	3	4
0	0	6	12	3	9
1	10	1	7	13	4
2	5	11	2	8	14

INVERSE MAP:

① $1 = 3 \cdot u + 5 \cdot v$

$u = -3$, $v = 2$

② $a = 10$

$b = -9$

③

Ex: $k_1 = 2$, $k_2 = 3$

$\Rightarrow [20 - 27]_{15} = [-7]_{15} = [8]_{15}$

$m_1 = 3, \quad m_2 = 5 \quad \Rightarrow m_1 m_2 = 15$

	0	1	2	3	4
0	0	6	12	3	9
1	10	1	7	13	4
2	5	11	2	8	14

INVERSE MAP:

① $1 = 3u + 5v$

CAN CHOOSE

$u = -3, \ v = 2$

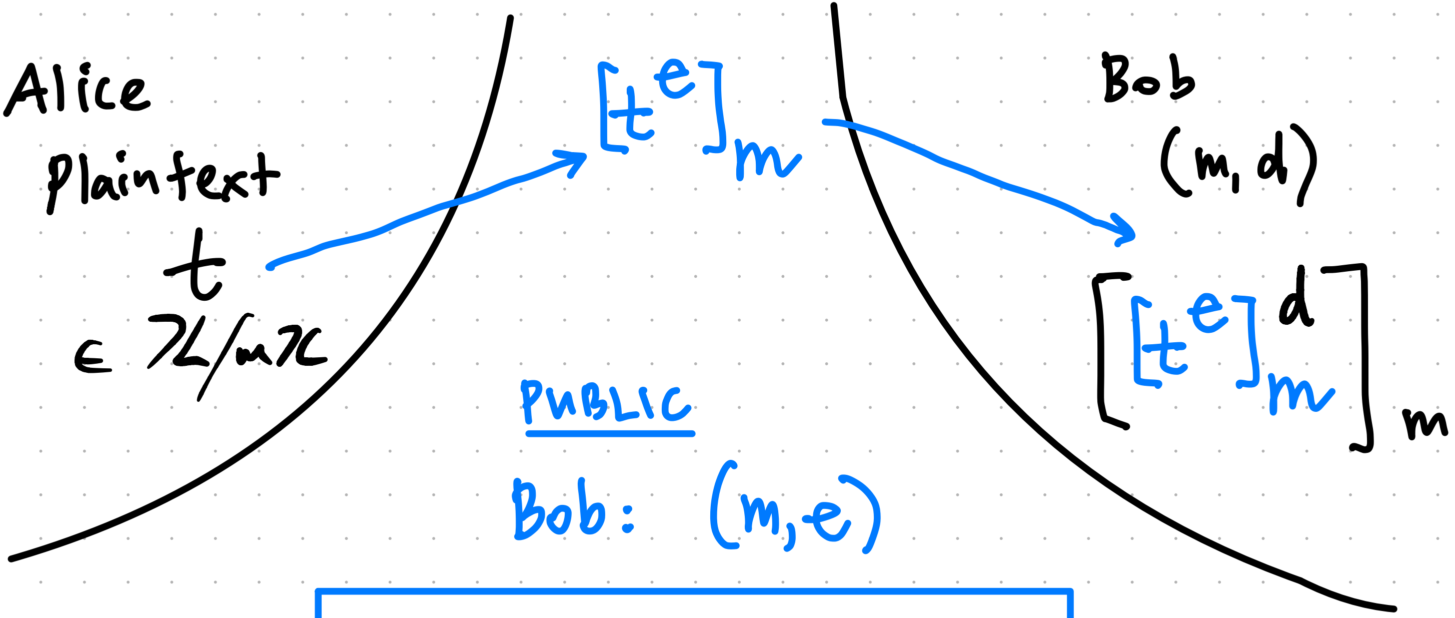
② $a = 10$

$b = -9$

③ $k = [10k_1 - 9k_2]_{15}$

$k_1 = 2, k_2 = 3 \rightsquigarrow [20 - 27]_{15} = [-7]_{15} = [8]_{15}$

BACK TO RSA



Alice

Plaintext

$t \in \mathbb{Z}/m\mathbb{Z}$

$[t^e]_m$

Bob

(m, d)

$\left[[t^e]_m^d \right]_m$

PUBLIC

Bob: (m, e)

all operations are in

$(\mathbb{Z}/m\mathbb{Z}, \cdot)$

BACK TO RSA

$$\left[[t^{e}]_m^{d} \right]_m = [t^{ed}]_m = [t]_m$$

CAN WE SELECT

e, d

SUCH THAT THIS HOLDS FOR ALL t ?

BACK TO RSA

$$\left[\left[t^e \right]_m^d \right]_m = \left[t^{ed} \right]_m = [t]_m$$

- WORKS IF $m = p$, A PRIME.

  → BUT NOT SECURE AT ALL!

- NOW: $m = pq$

BACK TO RSA

$$\left[[t^{e}]_{m}^{d}\right]_{pq} = [t^{ed}]_{pq} \overset{?}{=} [t]_{pq}$$

$$\Updownarrow \quad (\text{CRT})$$

$$\left([t^{ed}]_{p}, [t^{ed}]_{q}\right) \underset{?}{=} \left([t]_{p}, [t]_{q}\right)$$

SO, IF WE CAN SELECT

e, d

SUCH THAT:

$$\begin{cases} [t^{ed}]_p = [t]_p \\ [t^{ed}]_q = [t]_q \end{cases}$$

WE ARE DONE.

⟶ CAN WE ?

SO, IF WE CAN SELECT

e, d

SUCH THAT:

$$\begin{cases} [t^{ed}]_p = [t]_p \\ [t^{ed}]_q = [t]_q \end{cases}$$

HOLDS WHENEVER $ed = \ell_1(p-1)+1$

HOLDS WHENEVER $ed = \ell_2(q-1)+1$

WE ARE DONE.

$\longrightarrow$ CAN WE?

$\Rightarrow$ WE NEED TO SELECT

e, d SUCH THAT

$$ed = \ell_1 (p-1) + 1$$

$$\Leftrightarrow [ed]_{p-1} = [1]_{p-1}$$

$$ed = \ell_2 (q-1) + 1$$

$$\Leftrightarrow [ed]_{q-1} = [1]_{q-1}$$

RECIPE:

1) SELECT k A MULTIPLE OF BOTH $p-1$ AND $q-1$

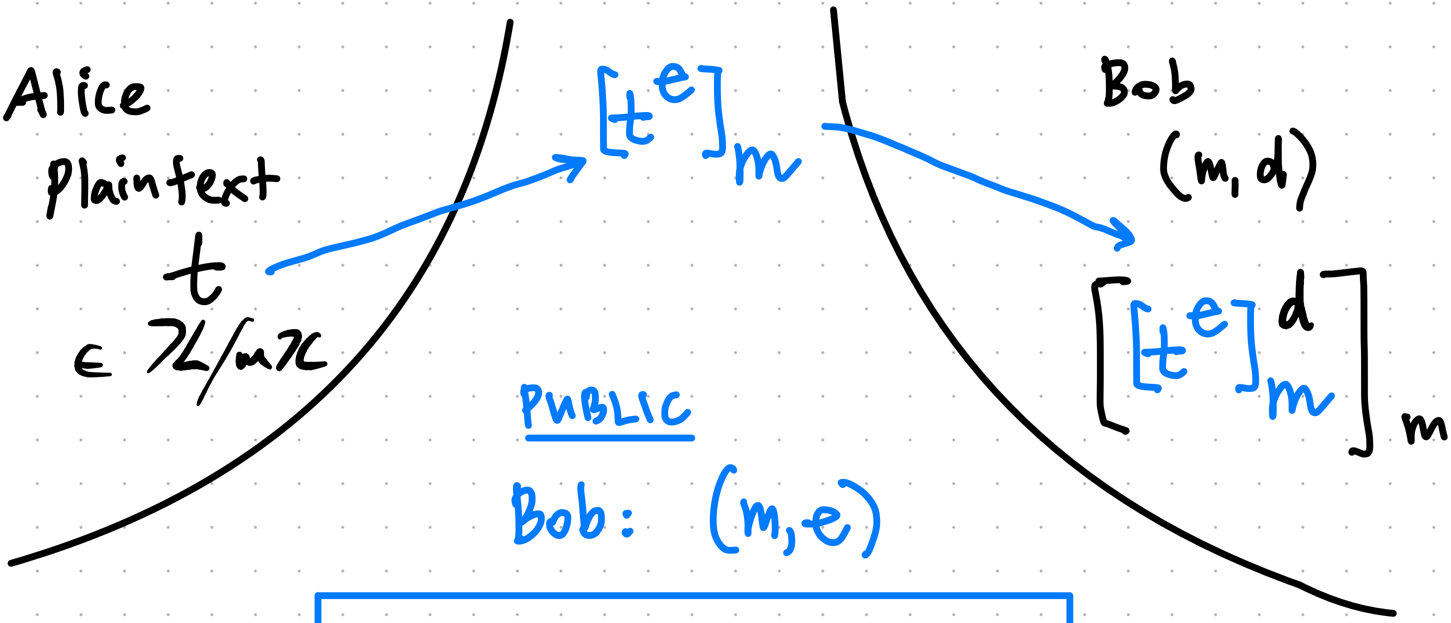
EX. $k = \phi(pq) = (p-1)(q-1)$

$k = \operatorname{lcm}(p-1, q-1)$

2) SELECT e : $\gcd(k, e) = 1$

3) SELECT d : $ed = \ell k + 1$

YESTERDAY



Alice

Plaintext

$t \in \mathbb{Z}/m\mathbb{Z}$

$[t^e]_m$

Bob

(m, d)

$\left[[t^e]_m^d\right]_m$

<u>PUBLIC</u>

Bob: (m, e)

all operations are in

$(\mathbb{Z}/m\mathbb{Z}, \cdot)$

1) SELECT m A PRIME, $m = p$.

→ SELECT e S.T. $\gcd(e, p-1) = 1$

→ SELECT d S.T. $[ed]_{p-1} = [1]_{p-1}$

WORKS BUT IS NOT SECRET.

1) SELECT m A PRIME, $m = p$.

→ SELECT e S.T. $\gcd(e, p-1) = 1$

→ SELECT d S.T. $[ed]_{p-1} = 1$

WORKS BUT IS NOT SECRET.

2) SELECT $m = pq$ (TWO DISTINCT PRIMES)

→ SELECT e S.T. $\gcd(e, p-1) = 1$

$\gcd(e, q-1) = 1$

→ SELECT d S.T. $[ed]_{p-1} = 1$

$[ed]_{q-1} = 1$

WORKS AND IS SECRET.

CRT: $\mathbb{Z}/m_1 m_2 \mathbb{Z}$, m_1, m_2 are coprime

$$[k]_{m_1 m_2} \longmapsto ([k]_{m_1}, [k]_{m_2})$$

$$[0]_{m_1 m_2} \longmapsto ([0]_{m_1}, [0]_{m_2})$$

has no multipliative inverse.

$$[m_1]_{m_1 m_2} \longmapsto ([0]_{m_1}, [m_1]_{m_2})$$

has no multipliative inverse.

CRT with p and q (two distinct primes)

$\mathbb{Z}/pq\mathbb{Z}$:

Q: What are **all** elements that do **not** have a multiplicative inverse?

A: $[k]_{pq} \longmapsto ([0]_p, [k]_q)$

OR

$\longmapsto ([k]_p, [0]_q)$

Fermat + Chinese Remainders

- Let p and q be distinct primes
- let k be a multiple of both $(p-1)$ and $(q-1)$
- for all non-negative integers l,

$$
\begin{aligned}
([a]_p)^{lk+1} &= [a]_p \\
([a]_q)^{lk+1} &= [a]_q
\end{aligned}
$$

- using the Chinese remainders theorem, we combine into

$$([a]_{pq})^{lk+1} = [a]_{pq}.$$

We have proved the following result:

THEOREM (TEXTBOOK THM 10.3)

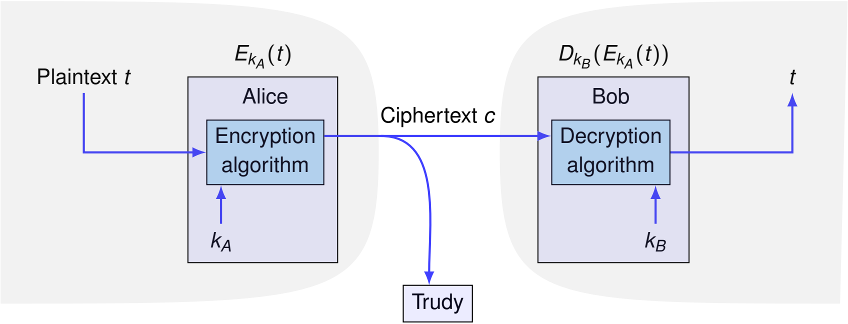
Let p and q be distinct prime numbers and let k be a multiple of both $(p-1)$ and $(q-1)$.

For every integer a, and every non-negative integer l,

$$([a]_{pq})^{lk+1} = [a]_{pq}.$$

Back to Cryptography

- RSA: Rivest, Shamir, Adleman, 1977 (first public-key cryptosystem).



RSA high-level: the very essence of it

Suppose that we can find:

- integer m (modulus),
- integer e (encoding exponent),
- integer d (decoding exponent),

such that, for all integers $t \in \mathbb{Z}/m\mathbb{Z}$ (plaintext),

$$[(t^e)^d]_m = [t]_m.$$

Then:

- the receiver generates m, e, d (we will see how).
- (m, e) is the public encoding key — announced in a phone-like public directory.
- (m, d) is the private decoding key — d never leaves the receiver.
- To send the plaintext $t \in \mathbb{Z}/m\mathbb{Z}$,
- the encoder forms the cryptogram $c = t^e \mod m$. Exponentiation is easy.
- The intended decoder performs $c^d \mod m$ and obtains the plaintext t. Again, this is easy.

Example

$m = 33$, $e = 7$, $d = 3$

- suppose that the plaintext is $t = 2$
- encryption: $c = t^e \mod m = 2^7 \mod 33 = 128 \mod 33 = 29$
- decryption: $c^d \mod m = 29^3 \mod 33 = \cdots = 2$, as expected.

It works similarly with any $t \in \{0, \ldots, 32\}$.

NB: we may want to exclude $t = 0$, because from $c = 0$ we immediately infer $t = 0$.

RSA: KEYS GENERATION (AT THE RECEIVER)

- generate large primes p and q at random
- $m = pq$ is the modulus used for encoding and decoding
- let k be a multiple of $(p - 1)$ and $(q - 1)$, to be kept secret
- for instance, $k = \phi(pq)$ or $k = \text{lcm}(p - 1, q - 1)$
- produce the public (encoding) exponent e such that $\gcd(e, k) = 1$
- (a common choice is $e = 65537 = 2^{16} + 1$ which is a prime number. No need for e to be distinct for each recipient)
- the public key is (m, e)
- k is kept secret. Using Bézout, the receiver produces the positive decoding exponent d such that

$$de + kl = 1.$$

- (m, d) is the private key.

$$\left(\left(\left(t^2\right)^2\right)^2 \cdots\right)^2 \cdot t$$

$$\left(\left(t^2\right)^2\right)^2 = t^{2^3}$$

$$\left(\left(\left(t^2\right)^2\right)^2\right)^2 = t^{2^4}$$

RSA: KEYS GENERATION (AT THE RECEIVER)

- generate large primes p and q at random
- $m = pq$ is the modulus used for encoding and decoding
- let k be a multiple of $(p-1)$ and $(q-1)$, to be kept secret
- for instance, $k = \phi(pq)$ or $k = \mathrm{lcm}(p-1, q-1)$
- produce the public (encoding) exponent e such that $\gcd(e, k) = 1$
- (a common choice is $e = 65537 = 2^{16} + 1$ which is a prime number. No need for e to be distinct for each recipient)
- the public key is (m, e)
- k is kept secret. Using Bézout, the receiver produces the positive decoding exponent d such that

$$de + kl = 1.$$

- (m, d) is the private key.

RSA: HOW DECODING WORKS

$[t]_m \in \mathbb{Z}/m\mathbb{Z}$, with $m = pq$. Hence

$$\begin{aligned}
\left([t]_m^e\right)^d &= [t]_m^{ed} \\
&= [t]_{pq}^{1-kl} \\
&= [t]_{pq} \qquad \text{Fermat + CRs} \\
&= [t]_m.
\end{aligned}$$

Example ("toy-key" generation)

- $p = 3,\ q = 11,\ m = 33,\ \ k = \text{lcm}(2, 10) = 10$
- $e = 7$ which is relatively prime with k
- $d = 3$ (check that $ed \mod k = 1$)
- the public key is $(m, e) = (33, 7)$
- the private key is $(m, d) = (33, 3)$

Example (encrypting and decrypting with the "toy key")

- Each letter of the alphabet is converted into a number in $\{1, 2, \ldots, m-1 = 32\}$ (we avoid 0, to avoid $c = 0 = t$).
- we use the natural order: $a \mapsto 1$, $b \mapsto 2$, etc.
- suppose we want to send the letter "b"
- the encoder maps it into the plaintext $t = 2$
- and encrypts: $c = t^e \mod 33 = 29$
- the decoder decrypts: $t = c^d \mod m = 2$
- and maps back $t = 2$ to the letter b.

In practice, m is very large, and the mapping

$$\text{text} \mapsto \mathbb{Z}/m\mathbb{Z}$$

is done in blocks of letters.

RSA: POSSIBLE ATTACKS

How to decrypt not knowing d? Here the possibilities (that we know of):

- factor m to find p and q. Very hard to do if m is large (say $\approx 2^{500}$).
- in $\mathbb{Z}/m\mathbb{Z}$, solve $c = x^e$ for x. Very hard to do if m is large.
- guess k (good luck!)
- guess t (good luck!)
- guess d (good luck!)

The trapdoor one-way function behind RSA

- The trapdoor one-way function is

$$t \mapsto c = t^e \mod m,$$

where e is called the encoding exponent.

- Instead of publishing the function, it suffices to publish (m, e). This is called the public key.
- Someone that knows (m, d) can perform

$$c \mapsto t = c^d \mod m,$$

where d is called the decoding exponent.

- Hence the trapdoor information is (m, d). It is called the private key.

The trapdoor one-way function behind RSA

- The trapdoor one-way function is

$$t \mapsto c = t^e \mod m,$$

  where e is called the encoding exponent.

- Instead of publishing the function, it suffices to publish (m, e). This is called the public key.

- Someone that knows (m, d) can perform

$$c \mapsto t = c^d \mod m,$$

  where d is called the decoding exponent.

- Hence the trapdoor information is (m, d). It is called the private key.

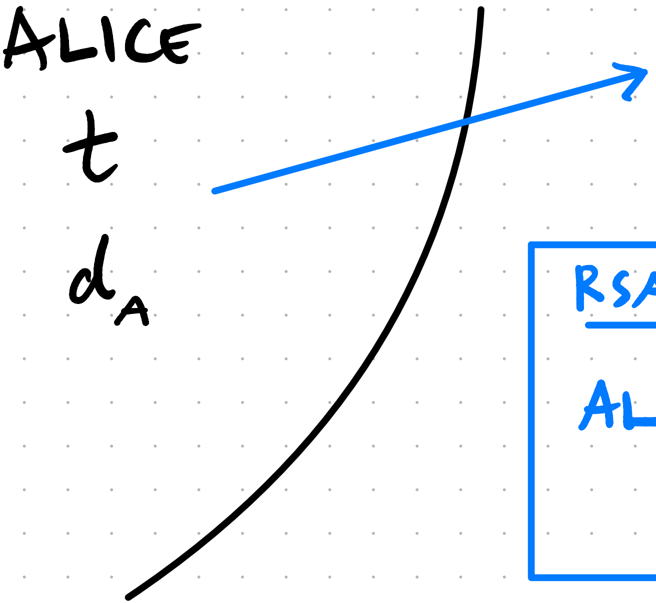
We have used trapdoor one-way functions for privacy.

In conjunction with hash functions, they are equally suited for authenticity.

DIGITAL SIGNATURE

<u>ISSUE</u>: PUBLIC RECEIVES A MESSAGE FROM ALICE.

BUT IS THE MESSAGE <u>REALLY</u> FROM ALICE?



$$(t, [t^{d_A}]_{m_A})$$

<u>RSA PUBLIC DIRECTORY</u>

ALICE .. (m_A, e_A)

HOW DOES THE PUBLIC CHECK?

STEP 1: $\left[\left([t^{d_A}]_{m_A}\right)^{e_A}\right]_{m_A} = \tilde{t}$

STEP 2: IF $\tilde{t} = t$

THEN ACCEPT!

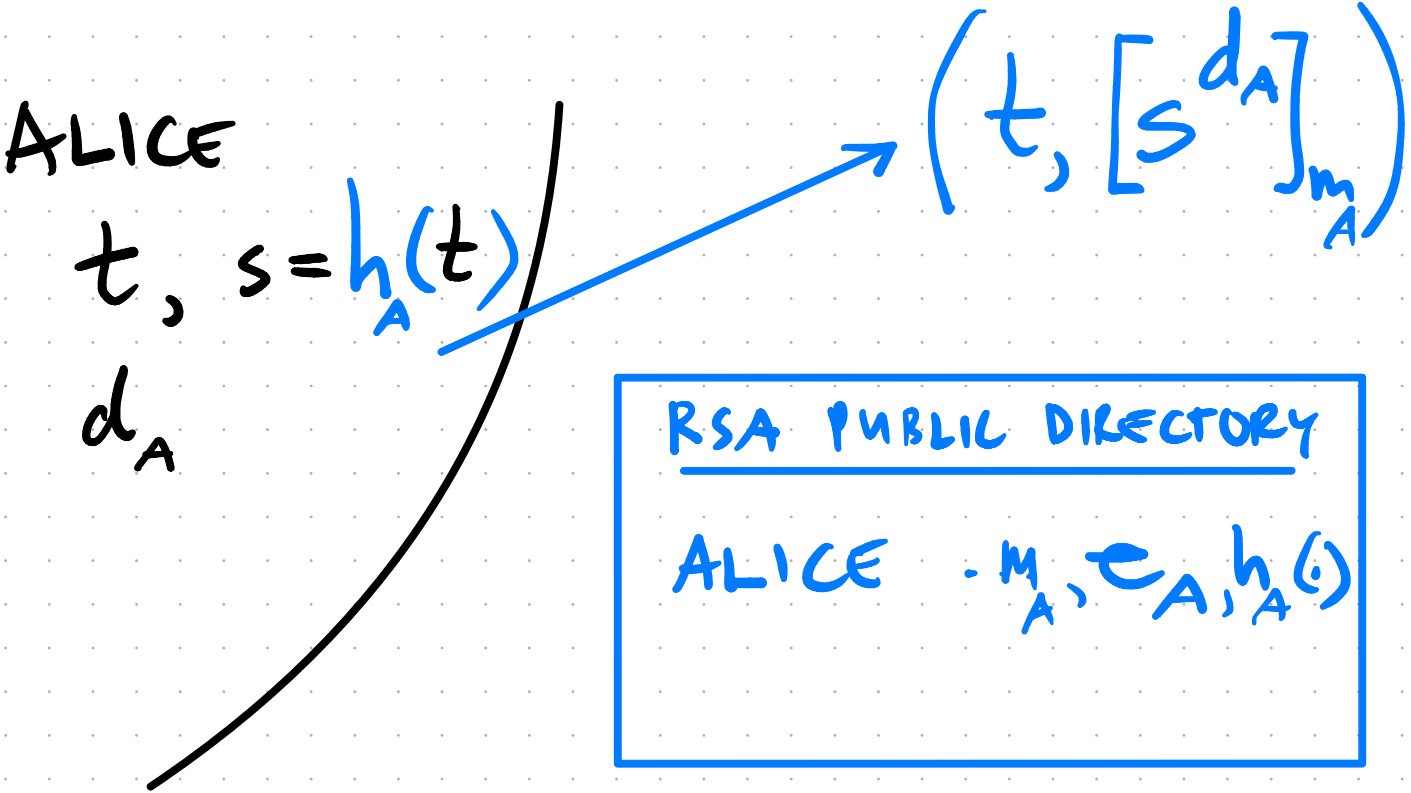
FOLLOW-UP QUESTION

- WITH THIS SCHEME, WE SEND THE MESSAGE **TWICE** ... SEEMS A BIT EXPENSIVE!

→ **IDEA**: "HASHING".

$$t \longmapsto$$

DIGITAL SIGNATURE



ALICE

$t, \; s = h_A(t)$

d_A

$\left(t, \left[s^{d_A}\right]_{m_A}\right)$

RSA PUBLIC DIRECTORY

ALICE $\cdot \; m_A, e_A, h_A(\cdot)$

HOW DOES THE PUBLIC CHECK?

STEP 1: $\left[\left(\left[s^{d_A}\right]_m\right)^{e_A}\right]_m = \hat{s}$

STEP 2: $t \longrightarrow h_A(t) = s$

STEP 3: IF $s = \hat{s}$, ACCEPT.

Hash Function

A hash function is a many-to-one function, used to map a sequence of arbitrary length to a fixed-length bit sequence of, say, 200 bits.

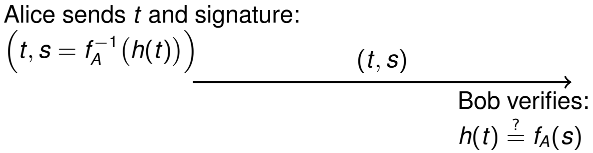
What we expect from a hash function, is that even the smallest change in the input produces a different output.

Ideally it should be so that one has to try about 2^{200} alternative inputs to hope to find a sequence that produces a given output.

DIGITAL SIGNATURE

To sign a document, we append to the document a hash function of the document in such a way that only the signee could have done it. This is done using a trapdoor one-way function as follows:

- let t be Alice's plaintext that she wants to sign;
- let f_A be Alice's trapdoor one-way function (publicly available);
- let h be a hash function (publicly available, the same function for everyone);
- the digital signature is $s = f_A^{-1}(h(t))$;
- the signed document is (t, s).



Alice sends t and signature:

$\left(t, s = f_A^{-1}(h(t))\right) \xrightarrow{(t, s)}$

Bob verifies:

$h(t) \overset{?}{=} f_A(s)$

If $h(t)$ equals $f_A(s)$, Bob trusts that the plaintext t is authentic, since for anybody other than Alice, it is nearly impossible to compute s.

Note 1: For privacy, Alice can encrypt (t, s) using Bob's trapdoor one-way function f_B.

Note 2: Privacy relies on f_B; authenticity relies on f_A.

Trusted Agency

How do we know that the directory storing all the public keys has not been tampered with?

Example

Alice queries the public directory for Bob’s public key.

The directory sends the message "Bob’s public key is k".

Eve, who is sitting on the wire, substitutes "Bob’s public key is k" with "Bob’s public key is $\tilde{k}$", where $\tilde{k}$ is her own public key.

By using $\tilde{k}$ to encrypt, Alice believes that only Bob will be able to decrypt.

But in fact, Eve is the only person that can decrypt.

How to prevent this from happening?

The directory information is signed by a trusted agency, say Symantec. Here is how:

- Symantec's public key is distributed once and for all via a channel that cannot be tampered with (e.g. hard-coded into the crypto hardware).
- Each directory entry is digitally signed by Symantec. We call the result a certificate.
- Anybody that has Symantec's public key can verify that the information received from the directory is authentic.
- Once verified, Alice can be confident that she is using Bob's public key.

Standards

Here are examples of widely-used standards:

- SHA-1 through SHA-3 (Secure Hash Algorithm) family: cryptographic hash functions.
- DSA (Digital Signature Algorithm), ECDSA (Elliptic Curve DSA): standards for digital signature.
- DES (Data Encryption Standard), AES (Advanced Encryption Standard): symmetric-key encryption standards. They are faster than RSA and require less memory.
- RSA (Rivest Shamir Adleman): public-key crypto.

WHY NOT JUST RSA?

With RSA we can encrypt (provide privacy) and sign (verify authenticity). Why do we need other cryptographic standards?

- DSA is faster than RSA in signing (and ECDSA a more recent standard than DSA). When keys have the same length, DSA leads to a shorter signature. RSA 512 bits has been cracked, only a DSA 280 bits has been cracked.
- The symmetric-key standards (DES, AES) are faster than RSA and require less memory. Most CPUs now include hardware that makes AES very fast.

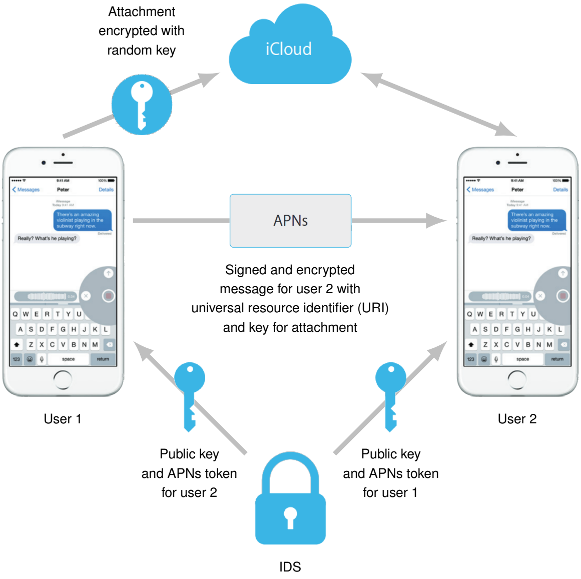
Cryptographic implementations, such as PGP (Pretty Good Privacy), available as a computer program, use symmetric-key and public-key cryptography, as well as digital-signature algorithms.

A COMPREHENSIVE EXAMPLE: HOW APPLE SENDS IMESSAGES

Apple uses all of the four standards mentioned above (SHA-1, ECDSA, AES, RSA). Let's see how.

To send an iMessage, Apple uses three services:

- IDS (Apple's directory service): It is here that public keys and device addresses are stored.
- APNs (Apple's Push Notification Service): outgoing messages are sent to this service. It is designed for short messages (like SMS).
- iCloud: to temporarily store what exceeds a maximum length. (Typically the case for a photo attachment.)



Messages
Peter
Details
There's an amazing violinist playing in the subway right now.
Really? What's he playing?

Attachment encrypted with random key

iCloud

APNs

Signed and encrypted message for user 2 with universal resource identifier (URI) and key for attachment

User 1

User 2

Public key and APNs token for user 2

Public key and APNs token for user 1

IDS

When the iMessage service is enabled on an Apple device (iPhone, iPad, Mac):

- The device produces the RSA keys (public, private, each 1280 bits) and the ECDSA keys (public, private, each 256 bits).
- The two public keys are sent to the IDS.
- IDS associates the keys to the device's APN address, and lists the APN address(es) under the user's email address (or phone number).

We can think that Bob’s IDS entry looks like this:

bob.cryptoexpert@epfl.ch	APN addr. (iMac)	RSA pub k
		ECDSA pub k
	APN addr. (iPhone)	RSA pub k
		ECDSA pub k
	APN addr. (iPad)	RSA pub k,
		ECDSA pub k
	APN addr. (MacBook Pro)	RSA pub k
		ECDSA pub k

When Alice sends a message to Bob using her Apple device, the following happens:

- The app looks in her contacts to find Bob’s email address (or phone number),
- The app sends a request to the IDS, asking for Bob’s APN addresses and corresponding RSA public keys.

For each of Bob's APN addresses, the following is done:

- The text, say t, is AES-encrypted with a randomly-generated symmetric key k to produce the cryptogram c_t;
- the key k is RSA-encrypted using Bob's public key, producing c_k;
- (c_t, c_k) are SHA-1-hashed and the result ECDSA-signed using Alice's private key, producing s;
- (c_t, c_k, s) is dispatched to the APN for delivery to the intended device.

Upon reception of (c_t, c_k, s), Bob's device does the following:

- Using Alice's public ECDSA key, the integrity of (c_t, c_k) is verified;
- using Bob's private RSA key, the cryptogram c_k is decrypted to obtain the AES symmetric key k;
- the cryptogram c_t is AES-decrypted to obtain the message t.

The APN can only relay messages up to a certain size (4 KB or 16 KB, depending on iOS).

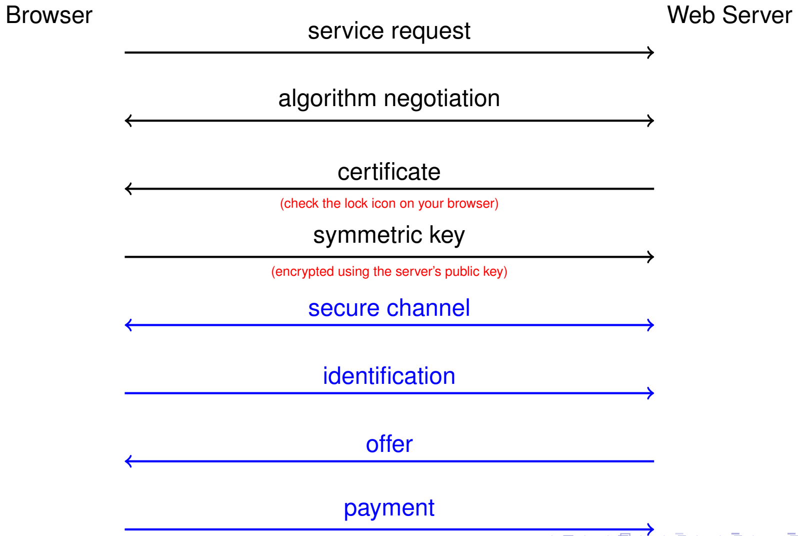
What exceeds this length, (e.g. a photo attachment), is AES-encrypted with a randomly-generated symmetric key, and the cryptogram is uploaded to iCloud.

The key, the URL, and the SHA-1 hash of the cryptogram are part of an iMessage sent to the recipient.

For further details, see the document: *iOS Security, iOS 9.0 or later, Sept. 2015.*

Another example: https

https (Hyper Text Transfer Protocol Secure) is the protocol used to exchange data between a browser and a web server. Sample transaction:



Browser

Web Server

service request

algorithm negotiation

certificate

(check the lock icon on your browser)

symmetric key

(encrypted using the server's public key)

secure channel

identification

offer

payment

OUTLINE

- INTRODUCTION AND ORGANIZATION
- ENTROPY AND DATA COMPRESSION
- CRYPTOGRAPHY
  - One-Time Pad, Perfect Secrecy, Public-Key (Diffie-Hellman)
  - Rudiments of Number Theory
  - Modular Arithmetic
  - Commutative Groups
  - Public-Key Cryptography
  - Summary of Chapter 2
- CHANNEL CODING

Summary of Chapter 2

Perfect secrecy is possible, but requires long keys.

- One-time pad

$$\text{Cryptogram} \;=\text{PlainText} \oplus \text{SharedKey}$$

  - If the SharedKey is perfectly (uniformly) random and shared between encrypter and decrypter ahead of time
  - and the SharedKey is kept secret from anyone else,
  - then the One-time Pad offers perfect secrecy.
  - Hence: It is expensive to implement. Only worth it for spies and such.

Summary of Chapter 2

- Practical cryptography is based on algorithmic/computational complexity.
- Public-key cryptography. Most public-key cryptographic algorithms fall into one of the following two categories:
    - those that are based on the belief that discrete exponentiation (in a multiplicative cyclic group) is a one-way function (e.g. Diffie-Hellman and ElGamal);
    - those that are based on the difficulty of factoring (e.g. RSA).
- To understand RSA and Diffie-Hellman, we need Number Theory and Algebra.

Summary of Chapter 2

Number Theory and Algebra

- Modulo operation, Euclid's algorithm
- Groups.
  - $\mathbb{Z}/m\mathbb{Z}$ with addition is always a group.
  - $\mathbb{Z}/m\mathbb{Z}$ with multiplication: need to retain only those elements that have a multiplicative inverse: $\mathbb{Z}/m\mathbb{Z}^*$
  - Finding multiplicative inverses in $\mathbb{Z}/m\mathbb{Z}$: Bézout's identity; Extended Euclid algorithm.
  - How many elements in $\mathbb{Z}/m\mathbb{Z}$ have a multiplicative inverse? Euler's totient function.
  - Group isomorphism.
  - Order of group elements. Lagrange's theorem: Order of any group element must divide the cardinality of the group.
- Product Groups. Main theorem: Cartesian product of groups is again a group.
- The special isomorphism between $\mathbb{Z}/m_1m_2\mathbb{Z}$ and $\mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z}$ when m_1 and m_2 are coprime.
  - Holds for both addition and multiplication, including for elements that do not have a multiplicative inverse.
  - Hence, this is more than just a group isomorphim.

Summary of Chapter 2

Computationally hard problem 1: Discrete logarithm.

- leads to **Diffie Hellman** (and, by slight extension, El Gamal)
  - Encryption: $A = g^a, B = g^b$.
  - Leads to a shared key: $C = A^b = B^a$.
  - To understand that it works, we need *cyclic groups.*

Summary of Chapter 2

Computationally hard problem 2: Factorization of large integers.

- leads to **Cocks/RSA**
  - Encryption: $t^e \mod m$, where t is the plaintext and $m = pq$, where p and q are primes.
  - Decryption: $(t^e)^d \mod m$
  - To understand that it works (meaning that $(t^e)^d \mod m = t$ for all plaintexts t), we need to understand $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$.

Summary of Chapter 2

- Authenticity: Digital Signatures.
    - Can be done with the same algorithm!

- In practice, so-called symmetric-key cryptosystems are important. The common secret key is typically only a few hundred bits, distributed e.g. via Diffie-Hellman. Encryption/decryption can be implemented more efficiently (faster algorithms, smaller hardware). Think: one-time pad, but with an imperfect key. There is no proof that the resulting algorithm is secure.

Outline

Introduction and Organization

Entropy and Data Compression

Cryptography

Channel Coding

- Error Detection and Error Correction
- Finite Fields and Vector Spaces
- Linear Codes
- Reed Solomon Codes
- Summary of Chapter 3