Radboud University

Introduction to public key cryptography

Cryptography, Autumn 2021

Lecturers: J. Daemen, B. Mennink

November 9, 2021

Institute for Computing and Information Sciences
Radboud University

Outline

Problems in key management

Public-key crypto: the idea

Modular arithmetic

Finite groups

Some elementary number theory

The discrete logarithm

Conclusions

Problems in key management

The blessings of crypto

Using (symmetric) crypto . . .

- Alice can protect her private data
- Alice and Bob can build a communication channel offering security:
  - ensure confidentiality of content
  - ensure authenticity of messages
  - over any communication medium
  - with respect to any adversary Eve
  - . . . that has access to communication medium
- Companies can protect their business
  - secure financial transactions
  - hide customer database from competitors
  - patch their products in the field for security/functionality
  - protect intellectual property in software, media, etc.
  - enforce their monopoly on games/accessories/etc.

The curse of crypto

- Alice and Bob need to share a secret cryptographic key
- A company/bank/gov't needs to distribute many cryptographic keys (*rolling them out*)
- ... in a way such that Eve cannot get her hands on them
- The security is only as good as the secrecy of these keys

Important lesson:

- Cryptography does not fully solve problems, but only reduces them to ...
  - securely generating cryptographic keys
  - securely establishing or rolling out cryptographic keys
  - keeping the keys out of Eve's hands

Key establishment

How do Alice and Bob establish a shared secret?

- When they physically meet:
  - exchange on a piece of paper or business card (unique pairs)
  - on a USB stick: requires trust in stick and PC/smartphone
  - but all cryptography requires trust in devices!
- When they don't meet, it is harder. Two cases:
  - there is a common and trusted *friend*: TTP
  - no such friend
- For companies key management is much harder
  - *Eve* is ubiquitous
  - keys must be protected *in the field*

Remote key establishment with trusted third parties (TTP)

Alice and Bob both trust a TTP and both share a secret key with it so they can communicate securely with that TTP

- They use TTP key distribution protocol for establishing K_{AB}
- Think of how this could work, as an exercise
- Problem is now: TTP has K_{AB} too

Alice and Bob trust multiple TTPs

- Alice and Bob establish one common key K_i per TTP
- Alice and Bob compute unique shared key K_{AB} as sum of all K_i
- Remaining risks
  - conspiracy: if TTPs collaborate, they can still cheat
  - denial-of-service: misbehaving TTP can prevent key setup
    - identifying saboteur is not easy

Remote key establishment w/o trusted third party

- Tamper-evident physically unclonable envelopes
  - tamper-evident: you cannot open it without leaving traces
  - unclonable: cannot fabricate one *looking the same*
- Sending by tamper-evident envelopes:
  - Alice sticks a 5 Euro banknote on the envelope with superglue
  - Alice writes down the serial number of the banknote
  - Alice sends a key K to Bob in the envelope
  - Upon receipt, Bob checks that the envelope was not opened
  - Bob checks whether the banknote is not counterfeit
  - Bob calls Alice and they check the banknote's serial number
  - Bob gets the key K from the envelope
  - The banknote makes the envelope hard-to-clone

Expensive and time-consuming, but could be worthwhile

- if you can keep your shared key secret, you only have to do this once
- the # people you need to communicate securely with is small

Two especially problematic use cases

Peer-to-peer networks with participants coming and going

- Every new contact Alice-Bob requires setting up a key $K_{A,B}$
- Can be done with central trusted third party (TTP)
- Each user shares a key with TTP, e.g., Bob has $K_{\mathsf{TTP},B}$
- but in peer-to-peer we don't want a TTP!

One-to-many authentication

- Software patches of Microsoft, Apple, Philips, Samsung, ...
- Device shall authenticate patch with secret key, kept in SIM (or so)
- Can be dealt with in different ways, each with disadvantages
  - 1 key: MS can broadcast single message-and-tag but compromise of one key breaks complete system
  - Unique keys: MS must compute tag per device per message

It would be great to have methods for:

Establishing a key $K_{A,B}$ without secret channel

Authenticating a message where receiver needs no secret

Key management challenges for companies/gov't

Some examples

- Bank: getting keys in all banking cards
- Microsoft or Apple: getting software verification key in all PCs
- Spotify or Netflix: getting keys in user PC/laptop/smartphones
- Government: getting keys in ID cards and travel passports
- More complex eco-systems
  - WWW: establishing keys between User PCs and internet sites
  - Public sector: keys in OV-Chipkaart and readers
  - Mobile phone: ensuring billing and confidentiality while roaming
- etc.

Public key cryptography to the rescue!

Public-key crypto: the idea

Public-key crypto wish list

It would be nice to:

- Set up a key remotely without the need for secret channel
- Authenticate an entity without having to share a secret key with it
- Authenticate documents without writer's secret key:
  - Cryptographic Signatures!
    AKA Digital Signatures
    AKA Electronic signatures

Public-key cryptography can do all that!

... and much more

Public-key crypto functionality

Public-key crypto is counter-intuitive: requires a **key pair** per user

- private key PrK: never to be revealed to the outside world
- public key PK: to be published and distributed freely

There are different types of public-key cryptosystems. Most used:

- Signature schemes
  - Alice uses PrK_A for signing message: $m, \mathrm{Sign}_{PrK_A}(m)$
  - anyone can use PK_A for verifying Alice's signature
  - PrK_A is also called *signing key* and PK_A *verification key*
- Key establishment: setting up of a shared secret
  - Key *agreement* (as in Diffie-Hellman)
    - Bob uses PrK_B and PK_A to compute secret K_{AB}
    - Alice uses PrK_A and PK_B to compute same secret K_{AB}
  - Key *transport*
    - Alice uses PK_B to transfer secret K_{AB} to Bob, that uses PrK_B

The translation dictionary analogy

- Translation dictionaries English-Navajo (native Americans)
  - Private key *PrK* is Dictionary Navajo to English
  - Public key *PK* is Dictionary English to Navajo
- Say Alice keeps the last copy of the Dictionary Navajo to English
  - Encryption: translate to Navajo using *PK*
  - Decryption: translate from Navajo using *PrK*
- Private key *PrK* can be reconstructed from public key *PK*!
  - Not secure?
  - In pre-computer days this was a huge task!
- Same for actual public-key cryptography
  - *PrK* can in principle be computed from *PK*
  - this needs to be a hard (mathematical) problem
  - an important part of public-key crypto is coming up with such problems
  - note: you don't have this in symmetric crypto

Public-key crypto: some history

- The idea of public-key crypto and first key-establishment scheme
  - R. Merkle, W. Diffie, M. Hellman in 1976
- The first public-key signature and *encryption scheme*
  - R. Rivest, A. Shamir and L. Adleman (RSA) in 1978
- Elliptic-Curve Cryptography
  - published independently by N. Koblitz and V. Miller in 1985
  - Most public-key crypto in use today is of this type
- Nowadays literally thousands of public-key systems (but few actually used)

Current hype: post-quantum crypto

- Quantum computer
  - can break all public-key crypto on the previous slide
  - very exotic: *computes in superposition* (kind of)
  - still hypothetical, though billions are spent on it
  - NSA/GCHQ, Google, IBM, etc. could possibly build one
- The need: public-key crypto resisting attackers with quantum computers
- (finalized) European project PQCRYPTO, see `http://pqcrypto.eu.org/`
- NIST contest for post-quantum crypto, currently ongoing see `https://csrc.nist.gov/projects/post-quantum-cryptography`
- Active involvement of Radboud colleagues

Modular arithmetic

Some notation

- $\mathbb{Z}$: the set of integers: $\{\ldots -3, -2, -1, 0, 1, 2, 3, \ldots\}$
- $a \in A$: this means that a is an element of a set A
  - $2 \in \mathbb{Z}$: 2 is element of set of integers $\mathbb{Z}$, or just 2 is an integer
  - $4/5 \in \mathbb{Q}$: $4/5$ is a rational number
- $\forall$: *for all* or *for every*
  - $\forall a \in \mathbb{Z} : a + 1 \in \mathbb{Z}$: for every integer a, $a + 1$ is also an integer
- $\exists$: *there exists*
  - $\forall a \in \mathbb{Z}, \exists b \in \mathbb{Z} : a + b = 0$ means: for every integer there exists an integer that when added to that integer gives 0
- $C = A \setminus B$ (set minus): C contains elements of A that are not in B
- $\#A$: the cardinality of a set, the number of elements it has
  - $\#\{\text{January}, \text{February}, \ldots, \text{December}\} = 12$

Residue classes modulo n

- In cryptography we want to work with finite sets
- One such finite set is the set of integers $\{0, 1, \ldots, n-1\}$
- We can do arithmetic on them, *modulo* n
- The underlying mathematics is the theory of *residue classes*

One writes $\mathbb{Z}/n\mathbb{Z}$ for the set of residue classes modulo n:

$$\mathbb{Z}/n\mathbb{Z} \quad = \quad \left\{\, \overline{0},\ \overline{1},\ \overline{2},\ \ldots\ , \overline{n-1} \right\}$$

with $\overline{m} = \{k \mid k \bmod n = m\}$

$\#(\mathbb{Z}/n\mathbb{Z}) = n$

We represent $\overline{m}$ of $\mathbb{Z}/n\mathbb{Z}$ by its member in the interval $[0, n-1]$

Modular addition

- $\mathbb{Z}/n\mathbb{Z}$ represented by positive integers smaller than n including zero
- Consider addition modulo n as an operation:
  (1) $c \leftarrow a + b$
  (2) if $c \geq n$, $c \leftarrow c - n$
- Notation: $a + b \bmod n$ or just $a + b$
- Interesting properties
  - the result of $a + b \bmod n$ is in $\mathbb{Z}/n\mathbb{Z}$
  - $a + b \bmod n = b + a \bmod n$: the order does not matter
  - $(a + b \bmod n) + c \bmod n = (a + (b + c) \bmod n) \bmod n$: the order of execution does not matter
  - $a + 0 \bmod n = a$: adding 0 has no effect
  - $a + b \bmod n = 0$ if $b = n - a$. So for every a there is a value b so that their sum is 0

Modular multiplication

- Consider now multiplication modulo n as an operation
  (1) $c \leftarrow a \cdot b$
  (2) do the result modulo n: $c \leftarrow c \bmod n$
- Notation: $a \cdot b \bmod n$ or $a \times b$
- Interesting properties:
  - the result of $a \cdot b \bmod n$ is in $\mathbb{Z}/n\mathbb{Z}$
  - $a \cdot b \bmod n = b \cdot a \bmod n$: the order does not matter
  - $((a \cdot b) \bmod n \cdot c) \bmod n = (a \cdot (b \cdot c) \bmod n) \bmod n$: the order of execution does not matter
  - $a \cdot 1 \bmod n = a$: multiplying by 1 has no effect
  - $a \cdot 0 \bmod n = 0$: multiplying by 0 always gives 0
  - $a \cdot b \bmod n = 1$ if, ...well, hmm, let's keep that for later

Finite groups

Group definition

- Couple $(A, \star)$ of a set A and an operation $\star$
- The binary operation must satisfy following properties:

closed:	$\forall a, b \in A:$	$a \star b \in A$
associative:	$\forall a, b, c \in A:$	$(a \star b) \star c = a \star (b \star c)$
neutral element:	$\exists e \in A, \forall a \in A:$	$a \star e = e \star a = a$
inverse element:	$\forall a \in A, \exists a' \in A:$	$a \star a' = a' \star a = e$
abelian (optional)	$\forall a, b \in A$	$a \star b = b \star a$

- Notational conventions

additive:	$(A, +)$	$e = 0$	$a' = -a$
multiplicative:	$(A, \cdot)$	$e = 1$	$a' = a^{-1}$

- Groups can be finite or infinite, depending on A

Terminology: Group order

Order of a finite group $(A, \star)$, denoted $\#A$, is number of elements in A

Examples of groups and non-groups

- Groups
  - $(\mathbb{Z}, +), (\mathbb{Q}, +), (\mathbb{R}, +), (\mathbb{C}, +)$
  - $(\mathbb{Q} \setminus \{0\}, \cdot), (\mathbb{R} \setminus \{0\}, \cdot), (\mathbb{C} \setminus \{0\}, \cdot)$
- Non-groups
  - $(\mathbb{N}, +)$: no neutral element, no inverses
  - $(\mathbb{Z} \setminus \{0\}, \cdot)$: elements without inverse
  - $(\mathbb{Q}, \cdot)$: zero has no inverse

Addition modulo n is a group

- Notation: $(\mathbb{Z}/n\mathbb{Z}, +)$
  - the set $\mathbb{Z}/n\mathbb{Z}$ with operation modular addition $+$
  - if operation is clear from the context, denoted as $\mathbb{Z}/n\mathbb{Z}$
- satisfies all required group properties and is abelian
- $(\mathbb{Z}/n\mathbb{Z}, +)$ is a group of order n

Multiplication modulo n is a group?

- Notation: $(\mathbb{Z}/n\mathbb{Z}, \times)$
- Satisfies required group properties, minus one
- 0 has no inverse, so $(\mathbb{Z}/n\mathbb{Z}, \times)$ is not a group
- maybe removing 0 may fix the problem?
- is $(\mathbb{Z}/n\mathbb{Z} \setminus \{0\}, \times)$ a group? Let's see later ...

Multiplication table, e.g., for $n = 7$:

$\mathbb{Z}/7\mathbb{Z}$	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Cyclic behaviour in finite groups

- Let $a \in A$ with $(A, \star)$ a group
- Consider the sequence:
  - $i = 1 : a$
  - $i = 2 : a \star a$
  - $i = 3 : a \star a \star a$
  - ...
  - $i = n : [n]a$ (additive) or a^n (multiplicative)
- In a finite group $(A, \star)$:
  - $\forall a \in A$ this sequence is periodic
  - period of this sequence: order of a, denoted $\mathrm{ord}(a)$

Terminology: Order of a group element

The order of a group element a, denoted $\mathrm{ord}(a)$, is the smallest integer $k > 0$ such that $a^k = 1$ (multiplicative) or $[k]a = 0$ (additive)

Cyclic groups and generators

- Let $g \in (A, \star)$
- Consider the set $[0]g, [1]g, [2]g, \ldots$
- This is a group, called a cyclic group, denoted: $\langle g \rangle$
  - Composition law: $[i]g + [j]g = [i + j \bmod \text{ord}(g)]g$
  - Neutral element $[0]g$
  - Inverse of $[i]g$: $[\text{ord}(g) - i]g$
- g is called the generator of this cyclic group
- Example of cyclic group $(\mathbb{Z}/n\mathbb{Z}, +)$
  - generator: $g = 1$
  - $[i]g = i$

Subgroups

A subset B of A that is also a group (under the same operation) is called a subgroup of A.

- $(B, \star)$ is a subgroup of $(A, \star)$ if
  - B is a subset of A
  - $e \in B$
  - $\forall a, b \in B : a \star b \in B$
  - $\forall a \in B :$ the inverse of a is in B

Lagrange's Theorem

If $(B, \star)$ is a subgroup of $(A, \star)$: $\#B$ divides $\#A$

- Case of cyclic subgroup: $\forall a \in A : \langle a \rangle$ is a subgroup of $(A, \star)$

Corollary (for order of elements)

For any element $a \in A$: $\mathrm{ord}(a)$ divides $\#A$

Example on orders: $(\mathbb{Z}/21\mathbb{Z}, +)$

- Order of $\mathbb{Z}/21\mathbb{Z}$: 21
- Order of 0: 1
- Order of 1: 21
- Order of 2: 21
- Order of 3: 7
- ...

Find the smallest i such that $i \cdot x$ is a multiple of n

Fact: order of an element in $(\mathbb{Z}/n\mathbb{Z}, +)$

$\mathrm{ord}(x) = n/\gcd(n, x)$ with
$\gcd(n, x)$: the greatest common divisor of x and n

Some elementary number theory

Prime numbers and factorization

- A number is prime if it is divisible only by 1 and by itself

  Prime numbers are: 2, 3, 5, 7, 11, 13, ... (infinitely many)
- Each number can be written in a unique way as product of primes (possibly multiple times), as in:

$$30 = 2 \cdot 3 \cdot 5 \qquad 100 = 2^2 \cdot 5^2 \qquad 12345 = 3 \cdot 5 \cdot 823$$

- Finding the prime number factorization is a computationally hard problem
- Easy for $143 = 11 \cdot 13$ but already hard for $2021 = 43 \cdot 47$
- Recently, factoring a 250-digit (829 bits) number $n = p \cdot q$ took 2700 Intel Xeon Gold 6130 CPU core-years (2.1GHz)

One can base public-key cryptosystems on the hardness of factoring

Greatest common divisor

- Definition:

$$\begin{aligned} \gcd(n, m) &= \text{greatest integer } k \text{ that divides both } n \text{ and } m \\ &= \text{greatest } k \text{ with } n = k \cdot n' \text{ and } m = k \cdot m', \\ &\qquad \text{for some } n', m' \end{aligned}$$

- Examples:

$$\gcd(20, 15) = 5 \qquad \gcd(78, 12) = 6 \qquad \gcd(15, 8) = 1$$

- Properties:
  - $\gcd(n, m) = \gcd(m, n)$
  - $\gcd(n, m) = \gcd(n, -m)$
  - $\gcd(n, 0) = n$

Terminology: relatively prime (or coprime)

If $\gcd(n, m) = 1$, one calls n, m *relatively prime* or *coprime*

Euclidean Algorithm

Property (assume $n > m > 0$):

- $\gcd(n, m) = \gcd(m, n \bmod m)$

This can be applied iteratively until one of arguments is 0

Example:

$$
\begin{aligned}
\gcd(171, 111) &= \gcd(111, 171 \bmod 111) = \gcd(111, 60) \\
&= \gcd(60, 111 \bmod 60) = \gcd(60, 51) \\
&= \gcd(51, 60 \bmod 51) = \gcd(51, 9) \\
&= \gcd(9, 51 \bmod 9) = \gcd(9, 6) \\
&= \gcd(6, 9 \bmod 6) = \gcd(6, 3) \\
&= \gcd(3, 6 \bmod 3) = \gcd(3, 0) = 3
\end{aligned}
$$

Variant allowing negative numbers :

$$
\begin{aligned}
\gcd(171, 111) &= \gcd(111, 171 \bmod 111) = \gcd(111, -51) \\
&= \gcd(51, 111 \bmod 51) = \gcd(51, 9) \\
&= \gcd(9, 51 \bmod 9) = \gcd(9, -3) \\
&= \gcd(3, 9 \bmod 3) = \gcd(3, 0) = 3
\end{aligned}
$$

$(\mathbb{Z}/n\mathbb{Z}, \times)$: a group?

- $\times$: Multiplication modulo n
- are group conditions satisfied?
  - closed: yes!
  - associative: yes!
  - neutral element: 1
  - inverse element: no, 0 has no inverse
- Let us exclude 0: so $((\mathbb{Z}/n\mathbb{Z}) \setminus \{0\}, \times)$
- Check properties again with multiplication table
- Examples:
  - $((\mathbb{Z}/7\mathbb{Z}) \setminus \{0\}, \times)$: OK!
  - $((\mathbb{Z}/21\mathbb{Z}) \setminus \{0\}, \times)$: Not OK!

Extended Euclidean Algorithm

The extended Euclidean algorithm returns a pair $x, y \in \mathbb{Z}$ with $m \cdot x + n \cdot y = \gcd(n, m)$

Our earlier example:

$$\begin{aligned}
-51 &= 171 - 2 \cdot 111 \\
9 &= 111 + 2 \cdot (-51) \\
3 &= (-51) + 6 \cdot 9 \\
0 &= (-9) + 3 \cdot 3
\end{aligned}$$

And now backward substitution:

$$\begin{aligned}
3 &= (-51) + 6 \cdot 9 \\
3 &= (-51) + 6 \cdot (111 + 2 \cdot (-51)) \\
3 &= (-51) + 6 \cdot 111 + 12 \cdot (-51) \\
3 &= 6 \cdot 111 + 13 \cdot (-51) \\
3 &= 6 \cdot 111 + 13 \cdot (171 - 2 \cdot 111) \\
3 &= 6 \cdot 111 + 13 \cdot 171 - 26 \cdot 111 \\
3 &= 13 \cdot 171 - 20 \cdot 111
\end{aligned}$$

Invertibility criterion

m has multiplicative inverse modulo n (i.e., in $\mathbb{Z}/n\mathbb{Z}$) iff $\gcd(m, n) = 1$

Proof

($\Rightarrow$) We have $m \cdot x \equiv 1 \pmod{n}$ so there is an integer y such that $m \cdot x = 1 + n \cdot y$ or equivalently $m \cdot x - n \cdot y = 1$. Now $\gcd(m, n)$ divides both m and n, so it divides $m \cdot x - n \cdot y = 1$. But if $\gcd(m, n)$ divides 1, it must be 1 itself.

($\Leftarrow$) Extended Euclidean algorithm yields x, y with $m \cdot x + n \cdot y = \gcd(m, n) = 1$. Taking both sides modulo n gives $m \cdot x \bmod n = 1$, or $x = m^{-1}$ □

Note: you can compute inverse with extended Euclidean algorithm!

Corollary

For p a prime, every non-zero $m \in \mathbb{Z}/p\mathbb{Z}$ has an inverse

$((\mathbb{Z}/p\mathbb{Z})^*, \times)$ with prime p: a cyclic group

- Here $(\mathbb{Z}/p\mathbb{Z})^*$ denotes $\mathbb{Z}/p\mathbb{Z}$ with 0 removed
- *As of now, presence of $*$ indicates operation $\times$, absence $+$*
- Every element has an inverse so now we know it is a group!
- Order of the group is $p-1$
- It can be proven that this group is cyclic
- Inverse of an element x:
  - Lagrange: order of an element divides group order $p-1$
  - so $x^{p-1} = 1$ (AKA Fermat's Little Theorem)
  - and $x^{-1} = x^{(p-1)-1} = x^{p-2}$
  - downside: would cost $p-3$ multiplications (at first sight ...)

Multiplicative prime groups

$(\mathbb{Z}/p\mathbb{Z})^*$ is a cyclic group of order $p-1$

Efficient exponentiation: Square-and-Multiply

- Computing $g^e \bmod p$ in naive way takes $e - 1$ multiplications
- Infeasible if g, e and p are hundreds of decimals long
- More efficient method: square-and-multiply
- Example: computing g^{43} with $g = 714, p = 1019$

$$
\begin{array}{rllrcr}
10 & g^2 & = g \times g & g^2 = & 296 & = & 714 \times 714 \\
100 & g^4 & = g^2 \times g^2 & g^4 = & 1001 & = & 296 \times 296 \\
1000 & g^8 & = g^4 \times g^4 & g^8 = & 324 & = & 1001 \times 1001 \\
10000 & g^{16} & = g^8 \times g^8 & g^{16} = & 19 & = & 324 \times 324 \\
100000 & g^{32} & = g^{16} \times g^{16} & g^{32} = & 361 & = & 19 \times 19
\end{array}
$$

working it out:

$$
\begin{array}{rllrcr}
11 & g^3 & = g^2 \times g & g^3 = & 411 & = & 296 \times 714 \\
1011 & g^{11} & = g^8 \times g^3 & g^{11} = & 694 & = & 324 \times 411 \\
101011 & g^{43} & = g^{32} \times g^{11} & g^{43} = & 879 & = & 361 \times 694
\end{array}
$$

- Only 5 squarings and 3 multiplications instead of 42

Exponentiation by Square-and-Multiply (cont'd)

Actual implementations use exponentiation algorithms like this one

- Computing g^{43} with *left-to-right* square-and-multiply

1	$t = g$	e.g.	$g =$	714,		$p = 1019$
10	$t = g^2 = t \times t$		$g^2 =$	296	$=$	714×714
100	$t = g^4 = t \times t$		$g^4 =$	1001	$=$	296×296
101	$t = g^5 = t \times g$		$g^5 =$	395	$=$	1001×714
1010	$t = g^{10} = t \times t$		$g^{10} =$	118	$=$	395×395
10100	$t = g^{20} = t \times t$		$g^{20} =$	677	$=$	118×118
10101	$t = g^{21} = t \times g$		$g^{21} =$	372	$=$	677×714
101010	$t = g^{42} = t \times t$		$g^{42} =$	819	$=$	372×372
101011	$t = g^{43} = t \times g$		$g^{43} =$	879	$=$	819×714

- Many variants exist, typical computation cost for $g^e \bmod p$:
  - $|e| - 1$ squarings, with $|e|$ the bitlength of e
  - 1 to $|e| - 1$ multiplications, depending on e and method
- Efficient: essential for a lot of public-key crypto to work

Pseudocode for Square-and-Multiply, left-to-right variant

Input: base $g \in \mathbb{Z}/p\mathbb{Z}$, exponent $a \in \mathbb{Z}/\operatorname{ord}(g)\mathbb{Z}$
Output: $A (= g^a) \in \mathbb{Z}/p\mathbb{Z}$

Let $a = a_0 + 2a_1 + 2^2a_2 + 2^3a_3 + \ldots + 2^{n-1}a_{n-1}$ and $\forall i : a_i \in \mathbb{Z}/2\mathbb{Z}$
$t \leftarrow g$
for $i \leftarrow n - 2$ down to 0 **do**
  $t \leftarrow t^2$
  if $a_i = 1$ **then** $t \leftarrow t \times g$
end for
return $A \leftarrow t$

The discrete logarithm

Isomorphism between $(\mathbb{Z}/p\mathbb{Z})^*$ and $\mathbb{Z}/(p-1)\mathbb{Z}$ for p prime

Multiplicative prime groups

If p is prime, $(\mathbb{Z}/p\mathbb{Z})^*$ is a cyclic group of order $p-1$

Alternative way of seeing it:

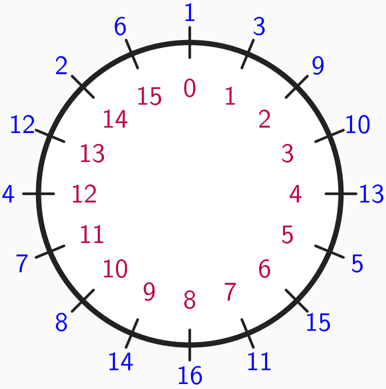
- Find a generator $g \in (\mathbb{Z}/p\mathbb{Z})^*$
- Write elements as powers of the generator: g^i
- Multiplication: find c such that $g^c = g^a \times g^b$
- Clearly: $g^a \times g^b = g^{a+b} = g^{a+b \bmod p-1}$
- So $c = a + b \bmod p - 1$

$(\mathbb{Z}/p\mathbb{Z})^*$ is just $\mathbb{Z}/(p-1)\mathbb{Z}$ in disguise!

These groups are *isomorphic*

Example of two isomorphic groups: $((\mathbb{Z}/23\mathbb{Z})^*, \times)$ and $(\mathbb{Z}/22\mathbb{Z}, +)$

Illustration with circle diagram: $(\mathbb{Z}/17\mathbb{Z})^*$ and $\mathbb{Z}/16\mathbb{Z}$



For each blue element $3^i \in \langle 3 \rangle$ we have a purple element $i \in \mathbb{Z}/16\mathbb{Z}$

- $C = A \times B = A \cdot B \bmod 17$ maps to $c = a + b \bmod 16$
- $C = A^e \bmod 17$ maps to $c = a \cdot e \bmod 16$

More abstract: Isomorphism between $\langle g \rangle$ and $\mathbb{Z}/\operatorname{ord}(g)\mathbb{Z}$

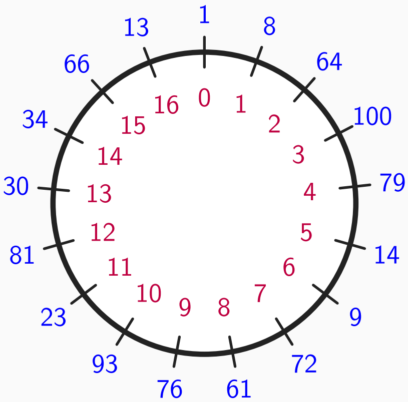
- For any integer x: $g^x = g^{x \bmod \operatorname{ord}(g)}$
- $g^{\operatorname{ord}(g)} = g^0 = 1$
- $A \times B = g^a \times g^b = g^{a+b} = g^{a+b \bmod \operatorname{ord}(g)}$

Correspondence between $\langle g \rangle$ and $\mathbb{Z}/\operatorname{ord}(g)\mathbb{Z}$

For every $A \in \langle g \rangle$ there is a number $a \in \mathbb{Z}/\operatorname{ord}(g)\mathbb{Z}$ such that $A = g^a$

- We call a the exponent of A
- we denote elements of $\langle g \rangle$ as X and their exponents as x
- the correspondence is called a *group isomorphism*

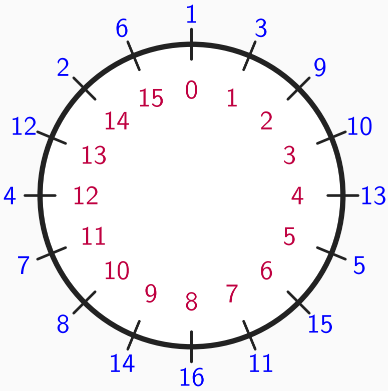
Illustration with a cyclic subgroup of $(\mathbb{Z}/p\mathbb{Z})^*$



Here $g = 8 \in (\mathbb{Z}/103\mathbb{Z})^*$ and $\mathrm{ord}(g) = 17$

For each $i \in \mathbb{Z}/17\mathbb{Z}$ we have $8^i \in (\mathbb{Z}/103\mathbb{Z})^*$

Discrete log



- Given x, compute X such that $X = 3^x \bmod 17$: exponentiation
- Given X, compute x such that $X = 3^x \bmod 17$: *discrete log*
- Exponentiation is easy but discrete log is hard for many groups $\langle g \rangle$

Conclusions

Conclusions

- Public-key crypto can do things symmetric crypto cannot
  - establish a secret key without the need for a confidential channel
  - signatures that can be verified without a secret
- Each participant has two keys
  - a private key that she must keep for herself
  - a public key that she can give to anyone
- The private key can be computed from the public key . . .
  but doing this would imply solving some well-know hard problem
- Two such hard problems:
  - factoring: $PrK = (p, q)$, $PK = n = p \cdot q$
  - discrete log over prime-order groups: $PrK = a$, $PK = A = g^a \bmod p$
- Note: In public-key crypto public keys need to be authenticated