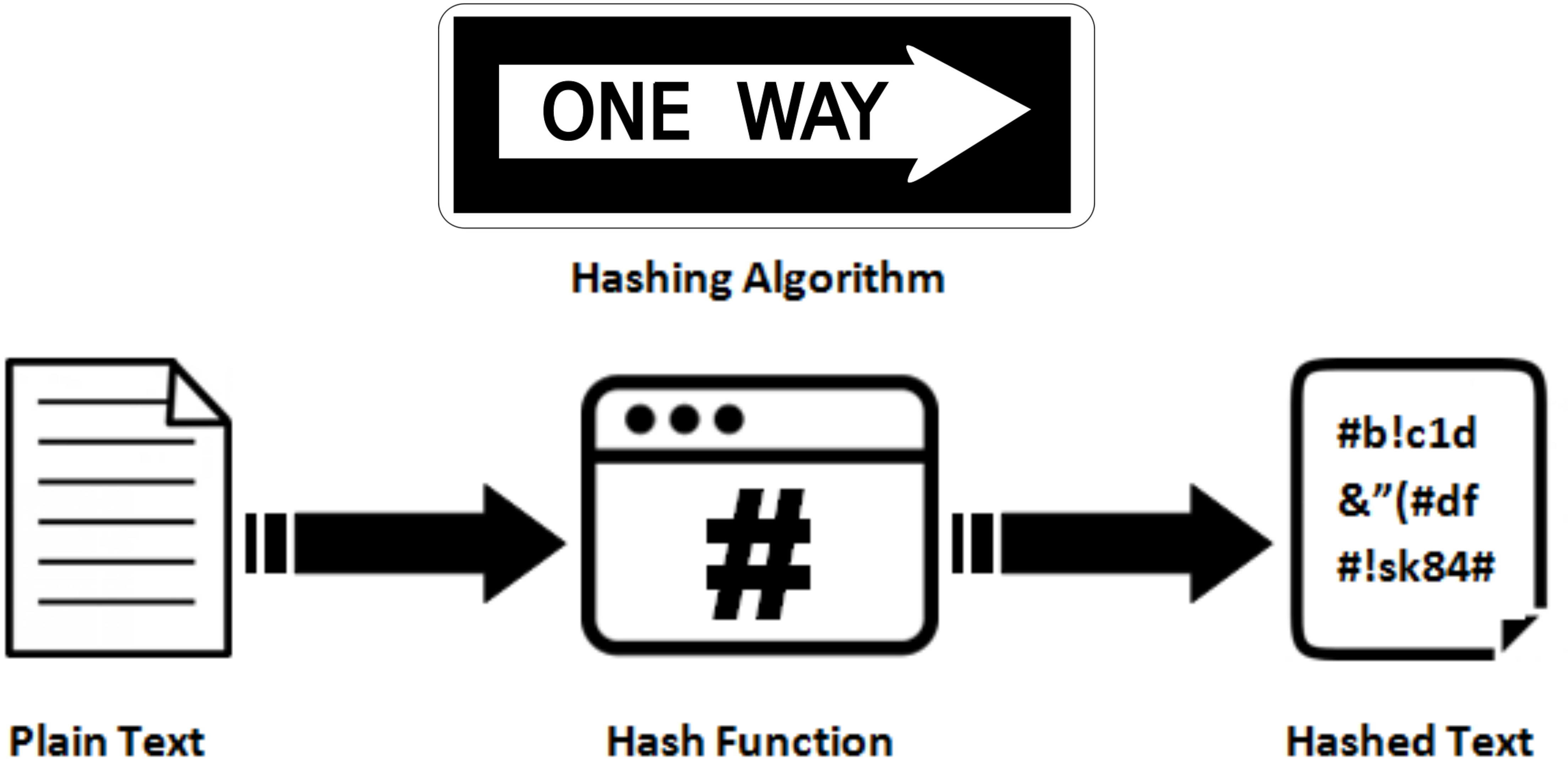
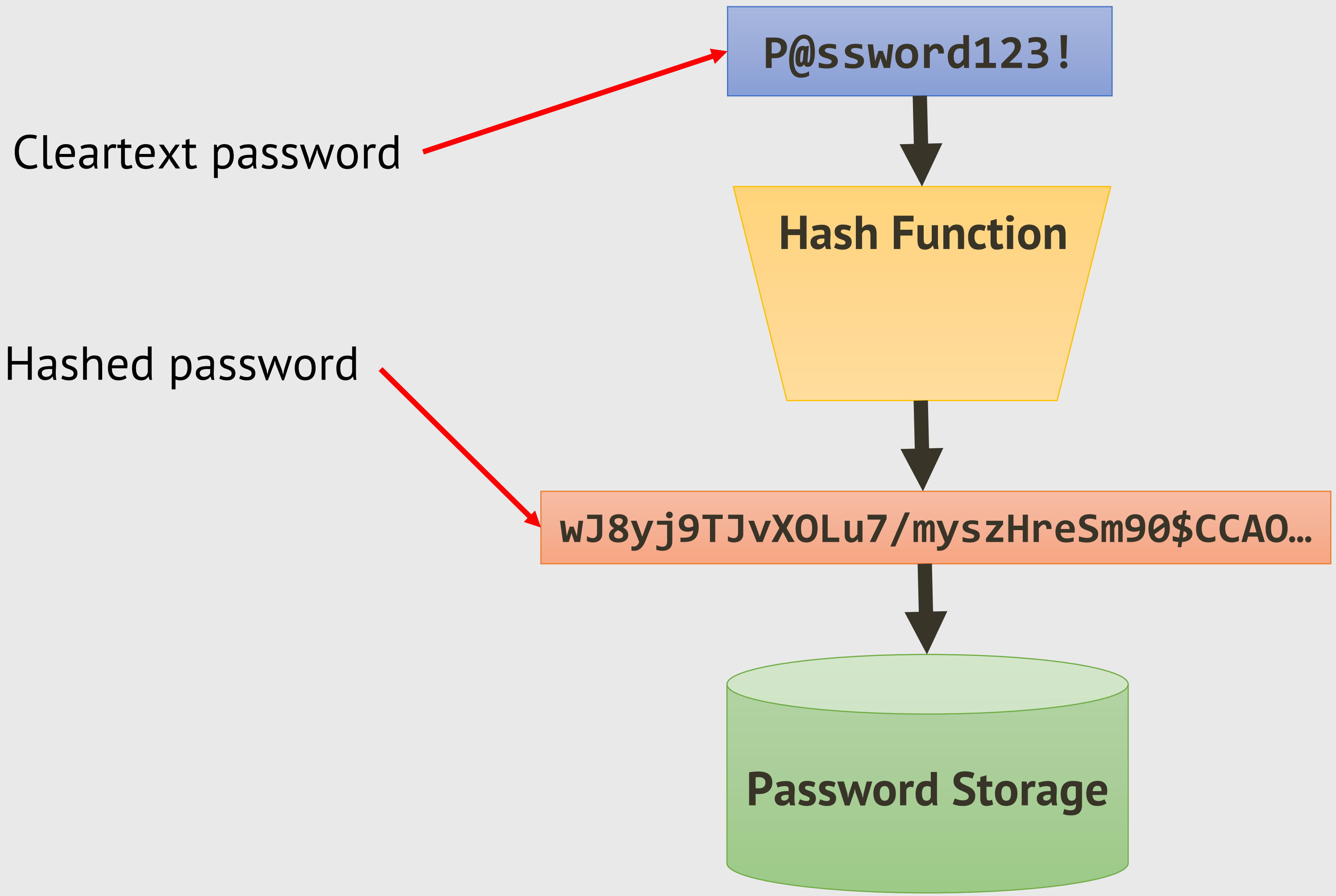
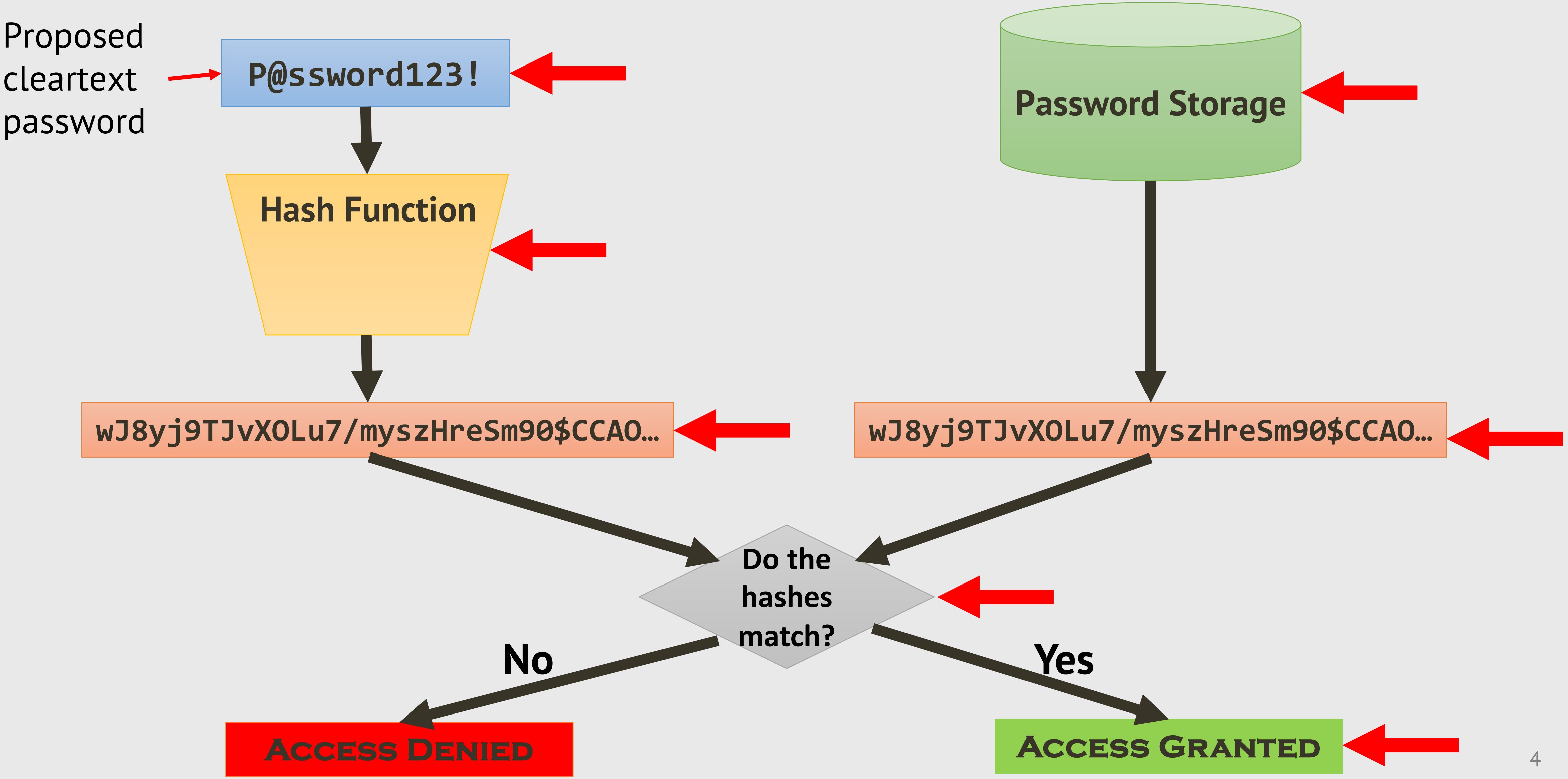
Securing Data


ONE WAY
Hashing Algorithm
#b!c1d
&"(#df
#!sk84#
Plain Text
Hash Function
Hashed Text

Storing a hash instead of a password



Testing a proposed password against stored



Spicing up the password with

Password: spaghetti

Salt: guHtGCfTx

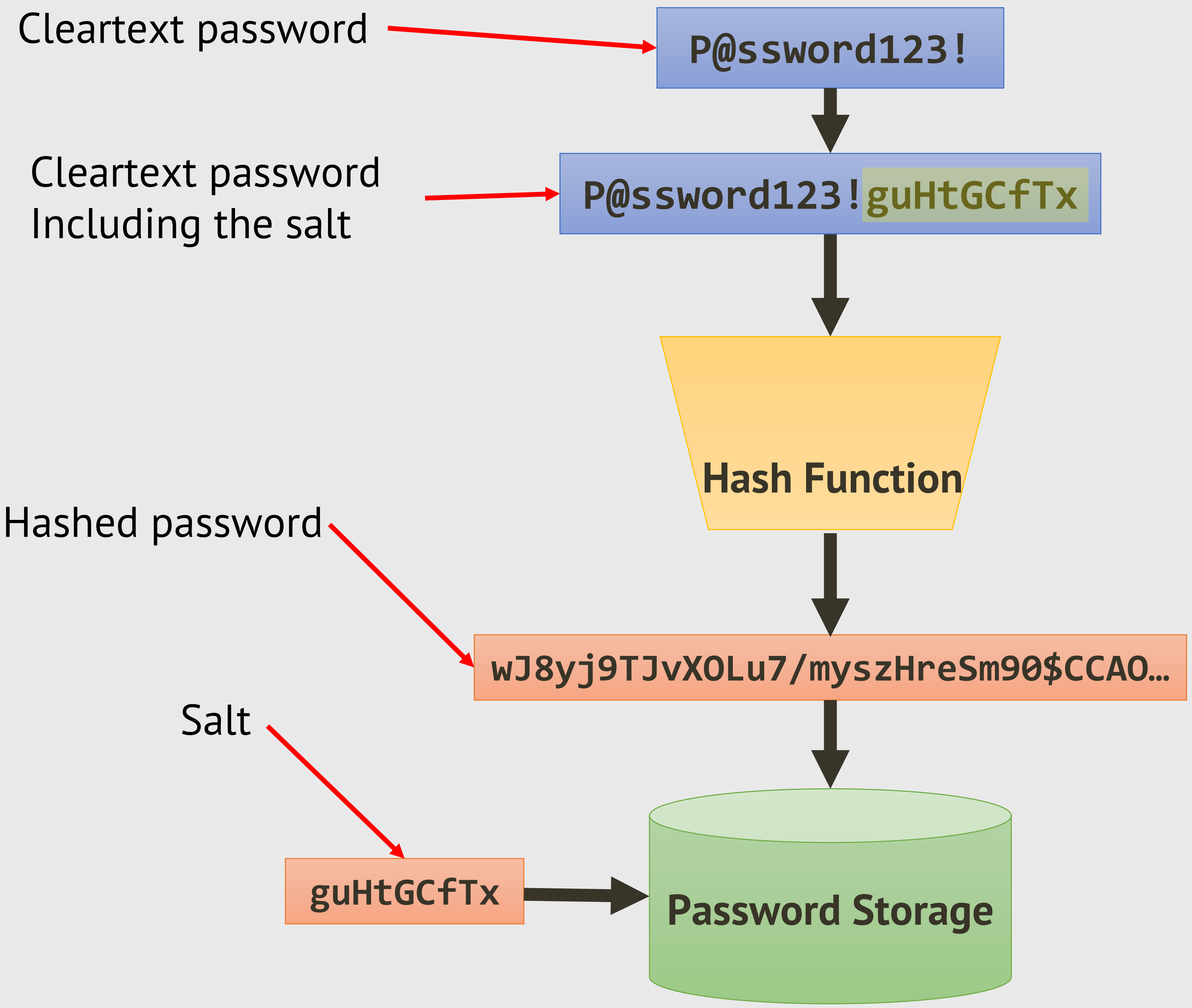
Salted Password: spaghettiguHtGCfTx

Spicing up the password with

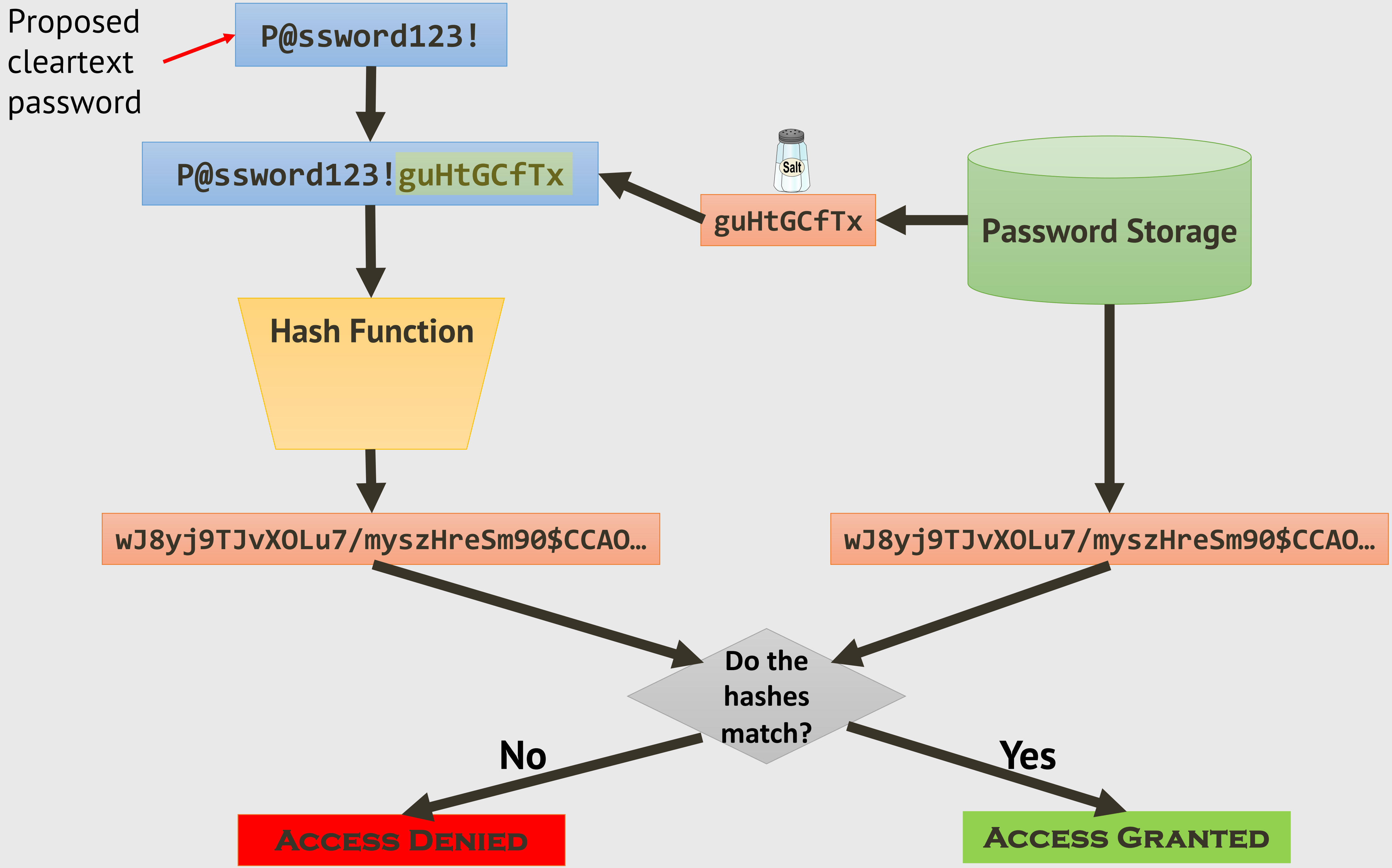
Salt is stored along **with** the hashed password

```
user6:$6$guHtGCfTx$Lk9AyxmfIJ7gav9TC3MfN7wwzadtb:18323:0:99999:7:::
```

Hashed password



Cleartext password
P@ssword123!
Cleartext password
Including the salt
P@ssword123!guHtGCfTx
Hash Function
Hashed password
wJ8yj9TJvXOLu7/myszHreSm90$CCAO…
Salt
guHtGCfTx
Password Storage


Proposed cleartext password
P@ssword123!
P@ssword123!guHtGCfTx
Salt
guHtGCfTx
Password Storage
Hash Function
wJ8yj9TJvXOLu7/myszHreSm90$CCAO…
wJ8yj9TJvXOLu7/myszHreSm90$CCAO…
Do the hashes match?
No
Yes
Access Denied
Access Granted

Spicing up the password with

A **pepper** is like a regular salt, <u>but not stored</u>

Securing Data

1. Codes
2. Ciphers
3. Symmetric-Key Encryption
4. Public-Key (Asymmetric) Cryptography


Eve
Alice
Bob

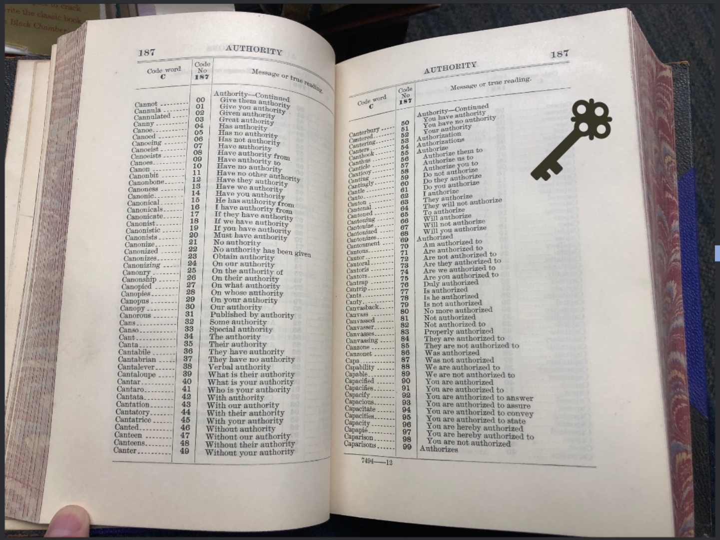
Securing Data

1. Codes
2. Ciphers
3. Symmetric-Key Encryption
4. Public-Key (Asymmetric) Cryptography

Code word C	Code No 187	Message or true reading.
		Authority—Continued
Cannot	00	Give them authority
Cannula	01	Give you authority
Cannulated	02	Given authority
Canny	03	Great authority
Canoe	04	Has authority
Canoed	05	Has no authority
Canoeing	06	Has not authority
Canoeist	07	Have authority
Canoeists	08	Have authority from
Canoes	09	Have authority to
Canon	10	Have no authority
Canonbit	11	Have no other authority
Canonbone	12	Have they authority
Canoness	13	Have we authority
Canonic	14	Have you authority
Canonical	15	He has authority from
Canonicals	16	I have authority from
Canonicate	17	If they have authority
Canonist	18	If we have authority
Canonistic	19	If you have authority
Canonists	20	Must have authority
Canonize	21	No authority
Canonized	22	No authority has been given
Canonizes	23	Obtain authority
Canonizing	24	On our authority
Canonry	25	On the authority of
Canonship	26	On their authority
Canopied	27	On what authority
Canopies	28	On whose authority
Canopus	29	On your authority
Canopy	30	Our authority
Canorous	31	Published by authority
Cans	32	Some authority
Canso	33	Special authority
Cant	34	The authority
Canta	35	Their authority
Cantabile	36	They have authority
Cantabrian	37	They have no authority
Cantalever	38	Verbal authority
Cantaloupe	39	What is their authority
Cantar	40	What is your authority
Cantaro	41	Who is your authority
Cantata	42	With authority
Cantation	43	With our authority
Cantatory	44	With their authority
Cantatrice	45	With your authority
Canted	46	Without authority
Canteen	47	Without our authority
Canteens	48	Without their authority
Canter	49	Without your authority

Code word C	Code No 187	Message or true reading.
		Authority—Continued
Canterbury	50	You have authority
Cantered	51	You have no authority
Cantering	52	Your authority
Canters	53	Authorization
Canthook	54	Authorizations
Canthus	55	Authorize
Canticle	56	Authorize them to
Canticoy	57	Authorize us to
Canting	58	Authorize you to
Cantingly	59	Do not authorize
Cantle	60	Do they authorize
Canto	61	Do you authorize
Canton	62	I authorize
Cantonal	63	They authorize
Cantoned	64	They will not authorize
Cantoning	65	To authorize
Cantonize	66	Will authorize
Cantonized	67	Will not authorize
Cantonizes	68	Will you authorize
Cantonment	69	Authorized
Cantons	70	Am authorized to
Cantor	71	Are authorized to
Cantoral	72	Are not authorized to
Cantoris	73	Are they authorized to
Cantors	74	Are we authorized to
Cantrap	75	Are you authorized to
Cantrip	76	Duly authorized
Cants	77	Is authorized
Canty	78	Is he authorized
Canvasback	79	Is not authorized
Canvass	80	No more authorized
Canvassed	81	Not authorized
Canvasser	82	Not authorized to
Canvasses	83	Properly authorized
Canvassing	84	They are authorized to
Canzone	85	They are not authorized to
Canzonet	86	Was authorized
Capa	87	Was not authorized
Capability	88	We are authorized to
Capable	89	We are not authorized to
Capacified	90	You are authorized
Capacifies	91	You are authorized to
Capacify	92	You are authorized to answer
Capacious	93	You are authorized to assure
Capacitate	94	You are authorized to convey
Capacities	95	You are authorized to state
Capacity	96	You are hereby authorized
Capapié	97	You are hereby authorized to
Caparison	98	You are not authorized
Caparisons	99	Authorizes

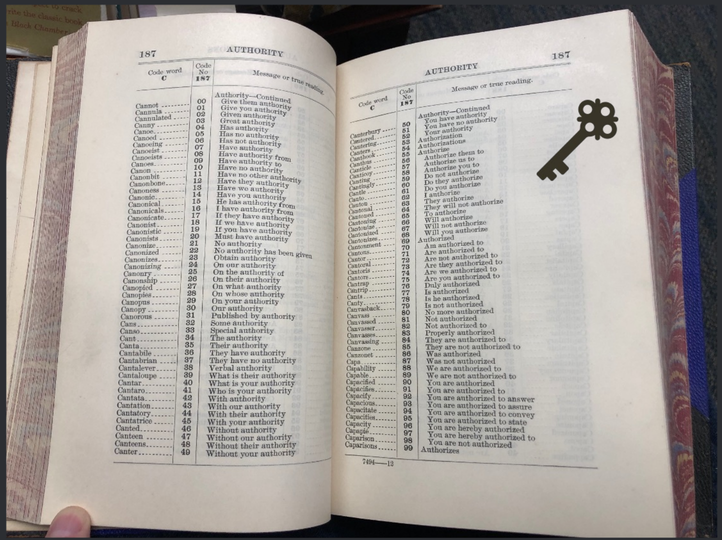
Sender



"They have authority to authorize"

18736 18765

Receiver



"They have authority to authorize"

Encode

plaintext → codetext

Decode

codetext → plaintext

Securing Data

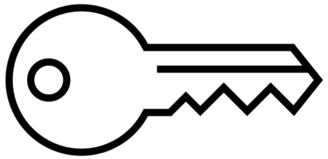
1. Codes
2. Ciphers
3. Symmetric-Key Encryption
4. Public-Key (Asymmetric) Cryptography

Caesar Cipher

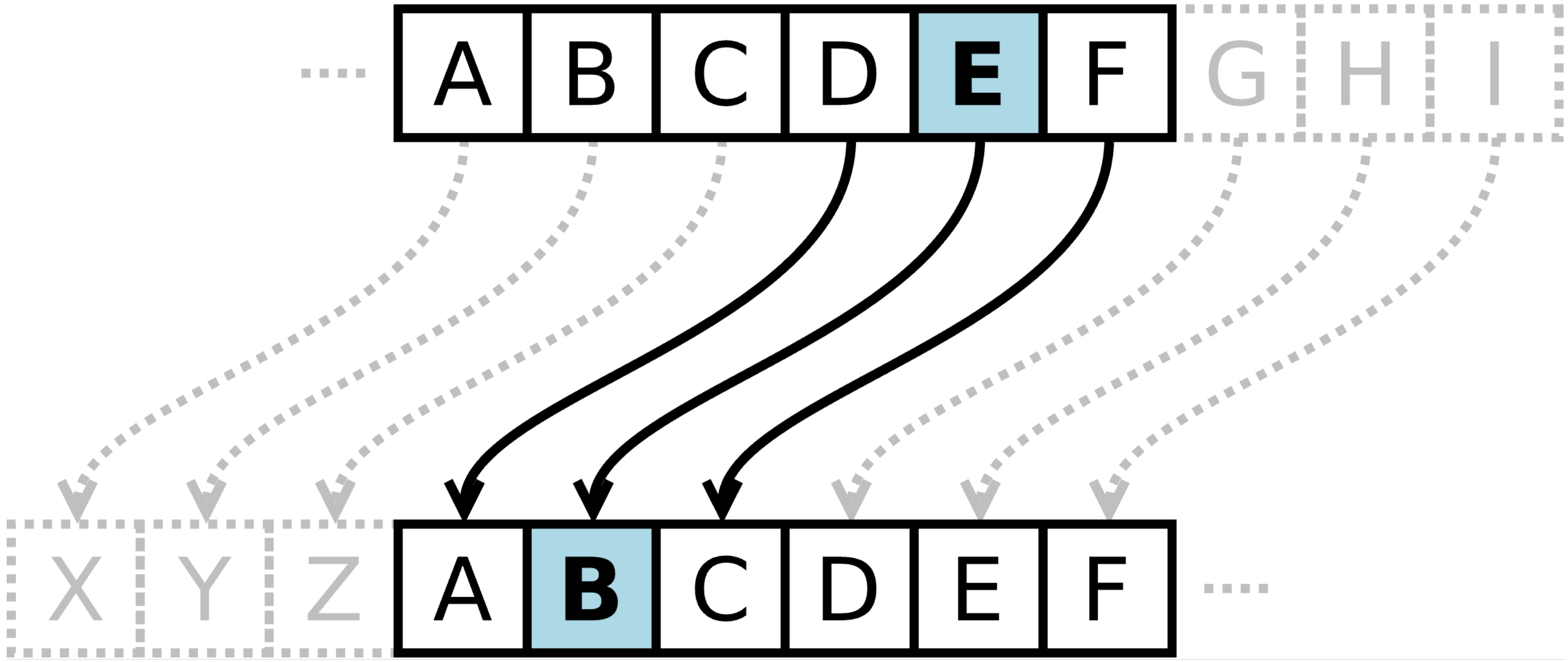
"If he had anything confidential to say, he wrote it in cipher, that is, by so changing the order of the letters of the alphabet, that not a word could be made out. If anyone wishes to decipher these, and get at their meaning, he must substitute the fourth letter of the alphabet, namely D, for A, and so with the others."

– *Suetonius*, *Life of Julius Caesar*

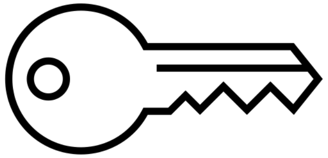
Caesar Cipher



Left shift of 3



Caesar Cipher



Left shift of 3

Plain	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cipher	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W

```
Plaintext:  THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG
Ciphertext: QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD
```


https://en.wikipedia.org/wiki/Caesar_cipher

Caesar Cipher (using modulo)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

$$E_n(x) = (x + n) \; mod \; 26$$

$$D_n(x) = (x - n) \; mod \; 26$$

Modulo

modulo (or "mod") is the remainder after dividing one number by another

Example:

$$14 \, mod \, 12 = 2$$

$$\frac{14}{12} = 1 \; with \; a \; remainder \; of \; 2$$

modulo (or "mod") is the remainder after dividing one number by another

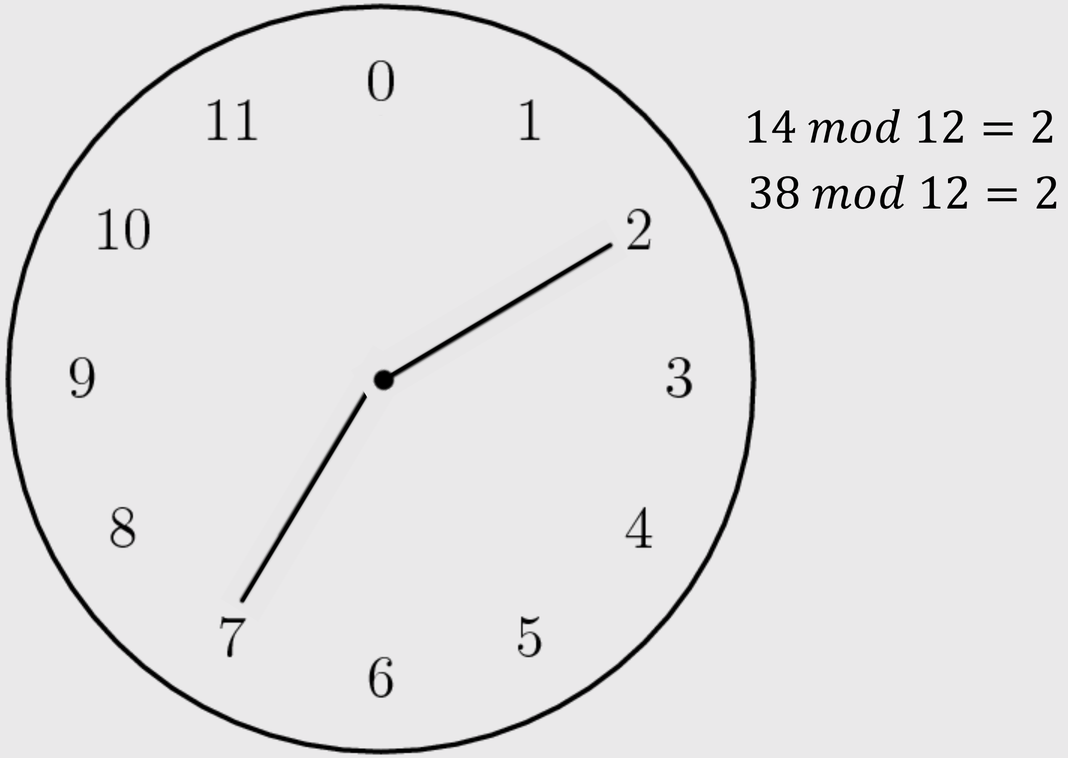
- Think of the value to the left of the mod as the number of steps around the clock

Examples:

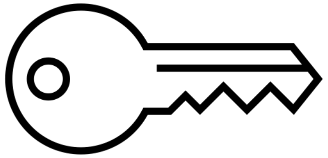
$$7\ mod\ 12 = 7$$

$$14\ mod\ 12 = 2$$

$$38\ mod\ 12 = 2$$



Caesar Cipher (using modulo)



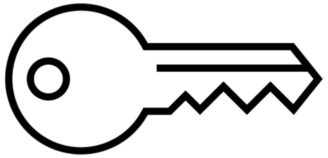
Right shift of n = 5

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

$$E_n(x) = (x + n)\ mod\ 26$$

$$E_5(2) = (2 + 5)\ mod\ 26 = 7$$

Caesar Cipher (using modulo)



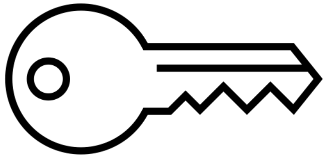
Right shift of n = 5

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

$$E_n(x) = (x + n)\ mod\ 26$$

$$\boldsymbol{E_5(2) = (24 + 5)\ mod\ 26} = 3$$

Caesar Cipher (using modulo)



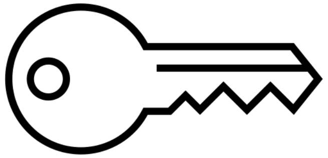
Right shift of n = 5

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

$$D_n(x) = (x - n) \; mod \; 26$$

$$\boldsymbol{D_5(7) = (7 - 5) \; mod \; 26} = 2$$

Caesar Cipher (using modulo)



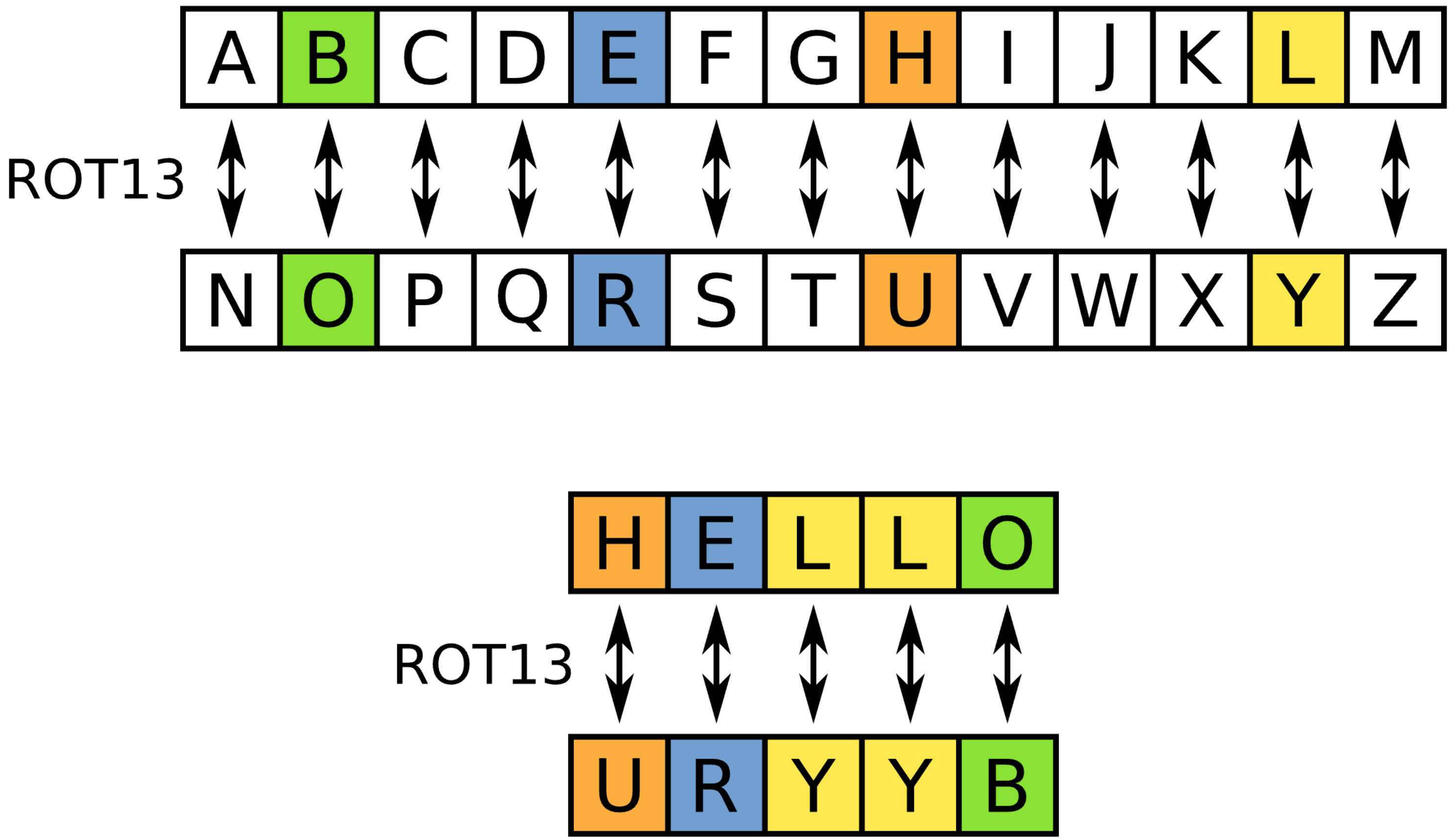
Right shift of n = 5

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

$$D_n(x) = (x - n)\ mod\ 26$$

$$D_5(3) = (3 - 5)\ mod\ 26 = 24$$

ROT13



https://en.wikipedia.org/wiki/ROT13

Caesar Cipher

1. Share a numeric **key** with your partner between 1 and 25
2. Encipher a secret message using https://inventwithpython.com/cipherwheel/
3. Send the ciphertext to your partner
4. Decipher your partner's message
5. Add the letters used in the deciphered message to chart of letter usage on the white board

https://en.wikipedia.org/wiki/Caesar_cipher

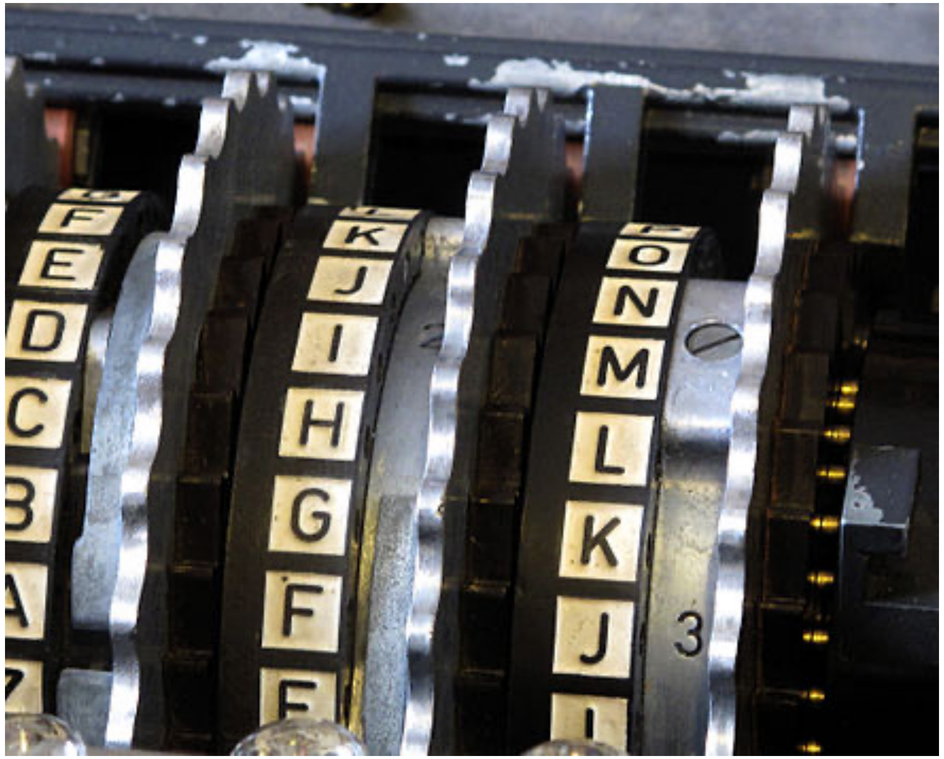
Cryptanalysis

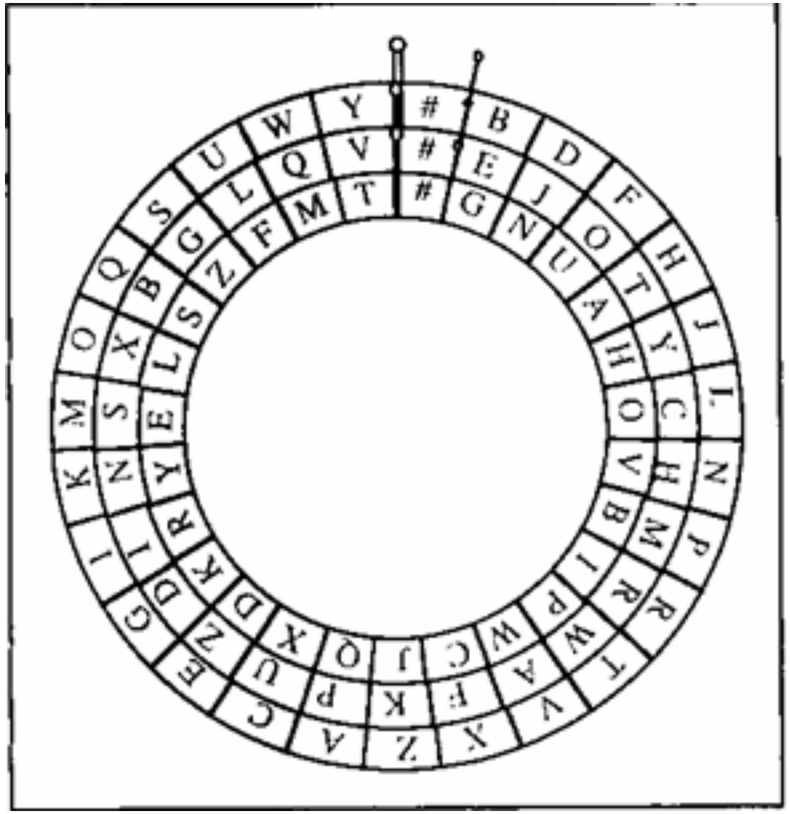
Z WFLEU KYV BVP

Enigma

https://en.wikipedia.org/wiki/Enigma_machine

Enigma





https://en.wikipedia.org/wiki/Enigma_machine

https://www.imdb.com/title/tt2084970/

Encipher

plaintext → ciphertext

Decipher

ciphertext → plaintext

Encrypt

plaintext → ciphertext

Decrypt

ciphertext → plaintext

Securing Data

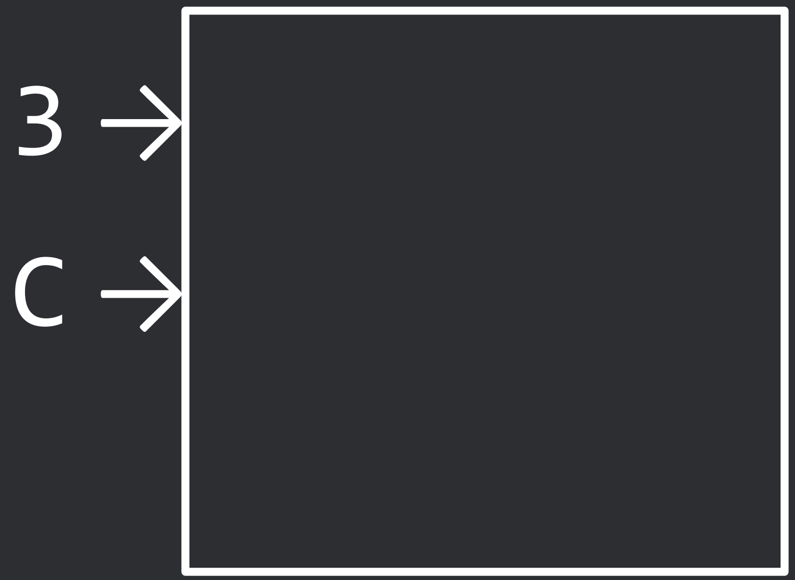
1. Codes
2. Ciphers
3. Symmetric-Key Encryption
4. Public-Key (Asymmetric) Cryptography

In-class demo

- Send a secure message across the room

AES
Triple DES
...


key
plaintext
ciphertext


3
C


3
C
F


3
A
D


3
T
W

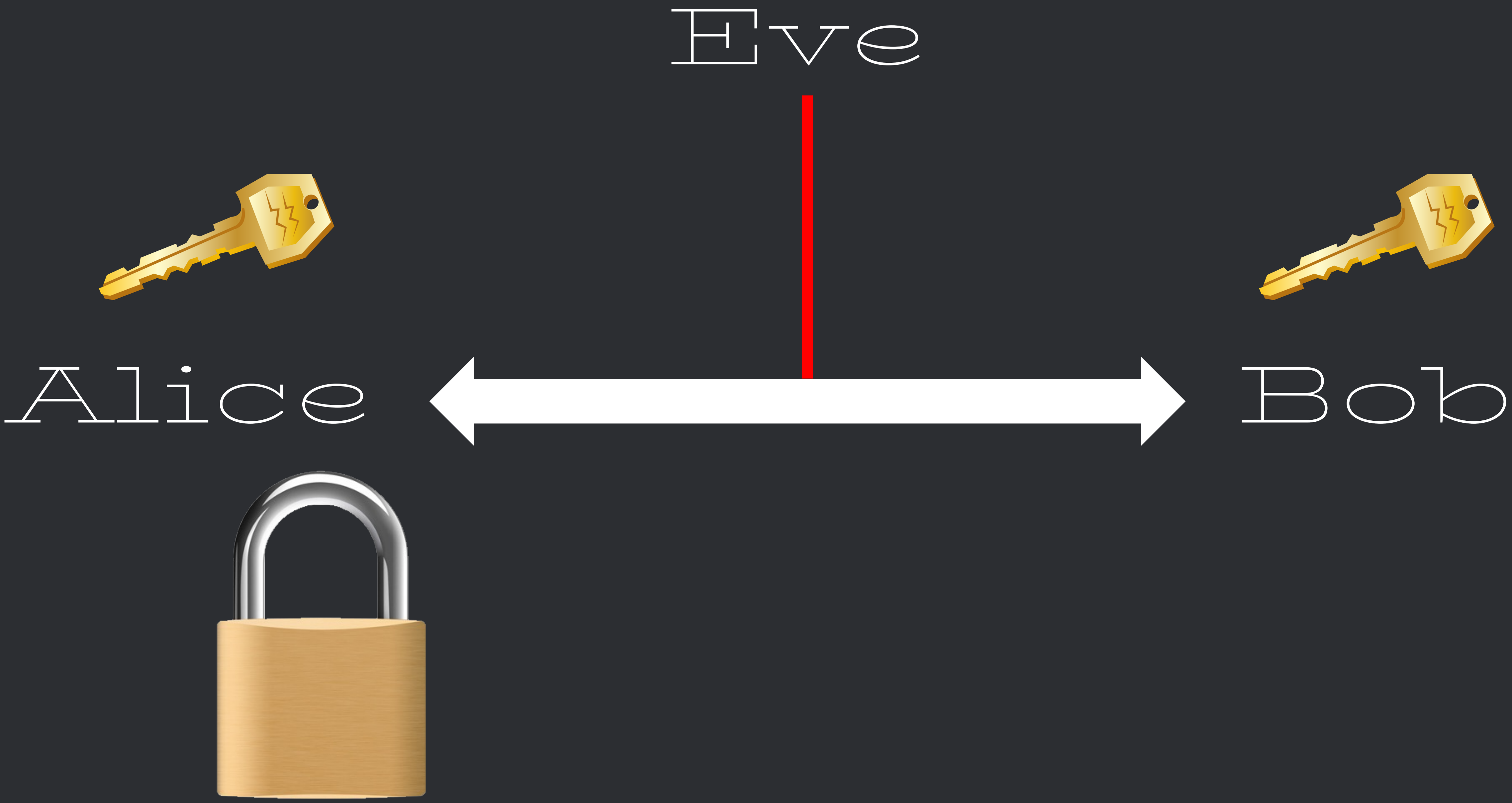
Decrypting


key
ciphertext
plaintext


3
F
C


3
D
A


3
W
T


Eve
Alice
Bob

Public-Key Cryptography

Asymmetric-key encryption

“Can the reader say what two numbers multiplied together will produce the number 8616460799? I think it unlikely that anyone but myself will ever know.”

-- William Stanley Jevons - T*he Principles of Science (1874)*

Diffie-Hellman
RSA
...

RSA (Rivest – Shamir – Adleman)

- One of the oldest (1977) and most widely used public-key cryptosystems for secure data transmission
- Public-key cryptography: the encryption key is public and distinct from the decryption key, which is kept private
- RSA is one of the cryptosystems used in Transport Layer Security, which is used by HTTPS

One-Way Function

Trapdoor One-Way Function

EASY

$$m^e \bmod n \equiv c$$

HARD

$$m^e \bmod n \equiv c$$

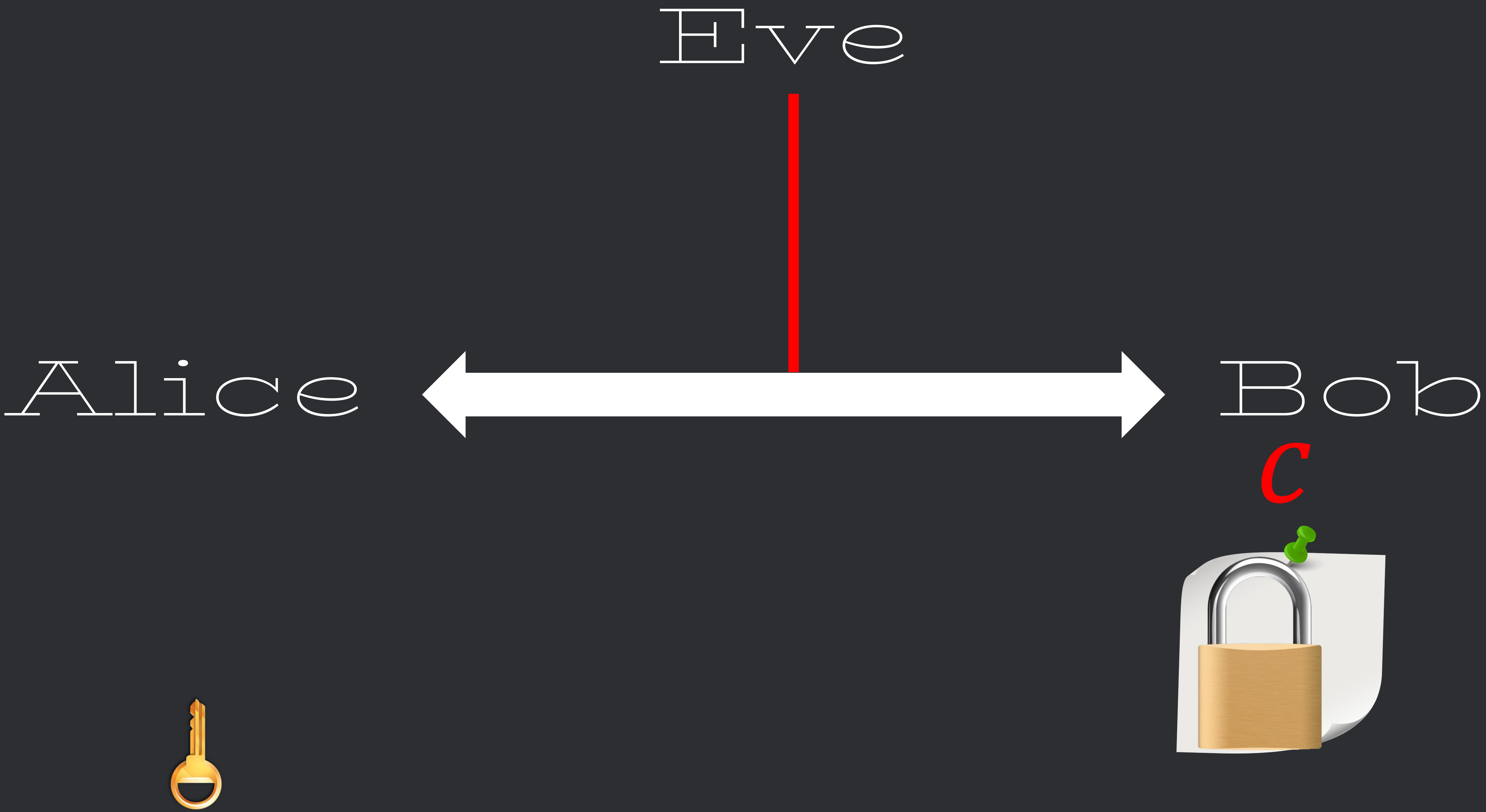

Eve
Alice
Bob
e
mod n
m

Eve

Alice

Bob

$$m^e \bmod n \equiv c$$


Eve
Alice
Bob
c

$$m^e \bmod n \equiv c$$

$$c^d \bmod n \equiv m$$

RSA (Rivest – Shamir – Adleman)

Public key $(n = 133, \ e = 29)$

Private key $(d = 41)$

Message: 99

Encrypt with: $m^e mod\ n \equiv c$

$99^{29}\ mod\ 133 = 92$

92 is the **ciphertext** message

Decrypt with: $c^d mod\ n \equiv m$

$92^{41}\ mod\ 133 = 99$

We recovered the **plaintext** message!

$$m^e \bmod n \equiv c$$

$$c^d \bmod n \equiv m$$

$$(m^e \bmod n)^d \bmod n \equiv m$$

$$(m^e)^d \bmod n \equiv m$$

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

Prime numbers

A **prime number** (or a **prime**) is a natural number greater than 1 that is not a product of two smaller natural numbers

A **composite number** is a natural number greater than 1 that is not prime

Integer factorization

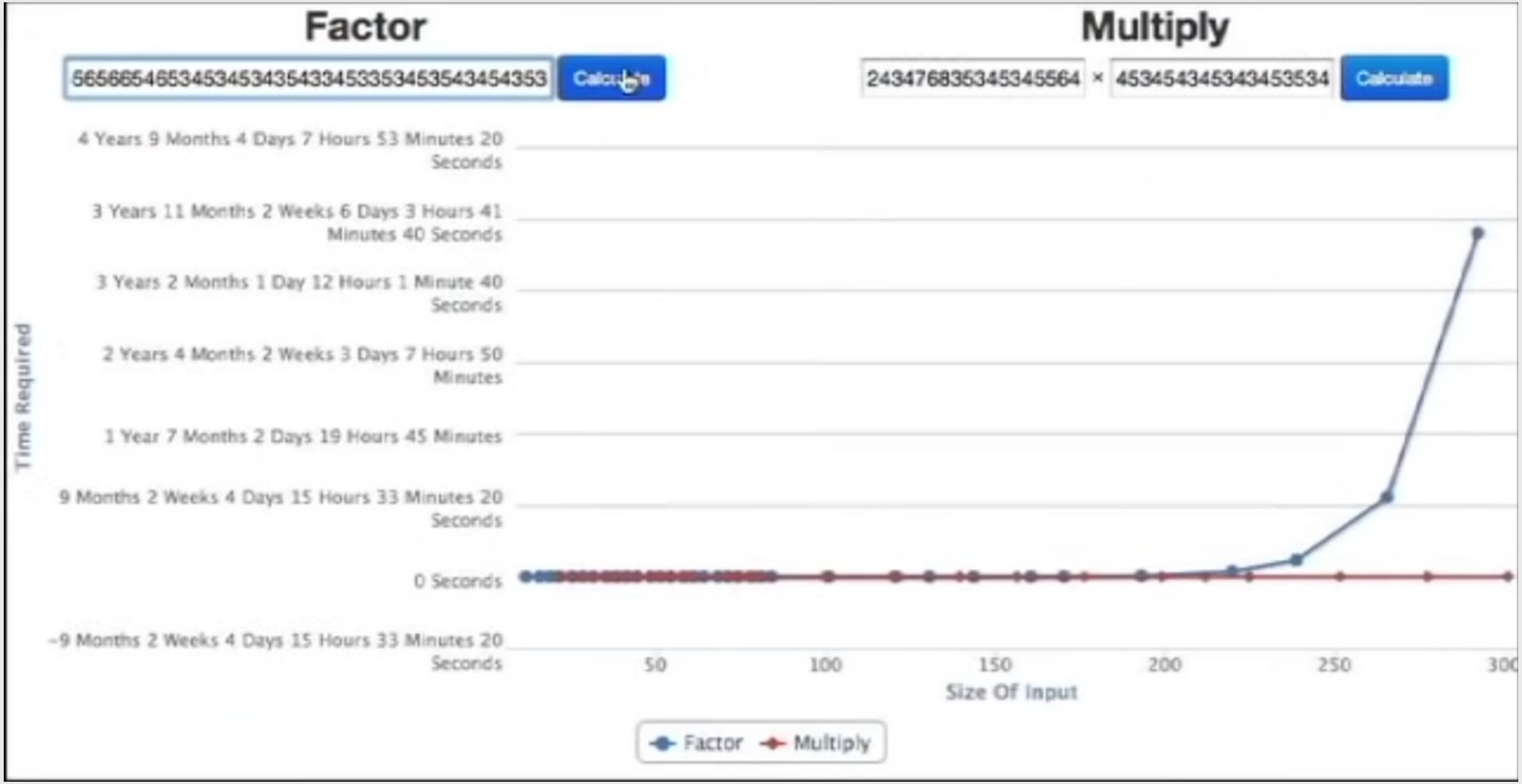
Decomposition of a positive integer into a product of integers

Example: 3×5 is an integer factorization of 15

When the numbers are sufficiently large, no efficient *non-quantum* integer factorization algorithm is known

The difficulty of this problem is important for the algorithms used in cryptography such as RSA public-key encryption

Integer factorization



Coprime

Two integers a and b are **coprime** if the only positive integer that is a divisor of both is 1

Example: 8 and 9 are since 1 is their only common divisor

modulo (or "mod") is the remainder after dividing one number by another

- Two integers a and b are **congruent** modulo n, if n is a divisor of their difference

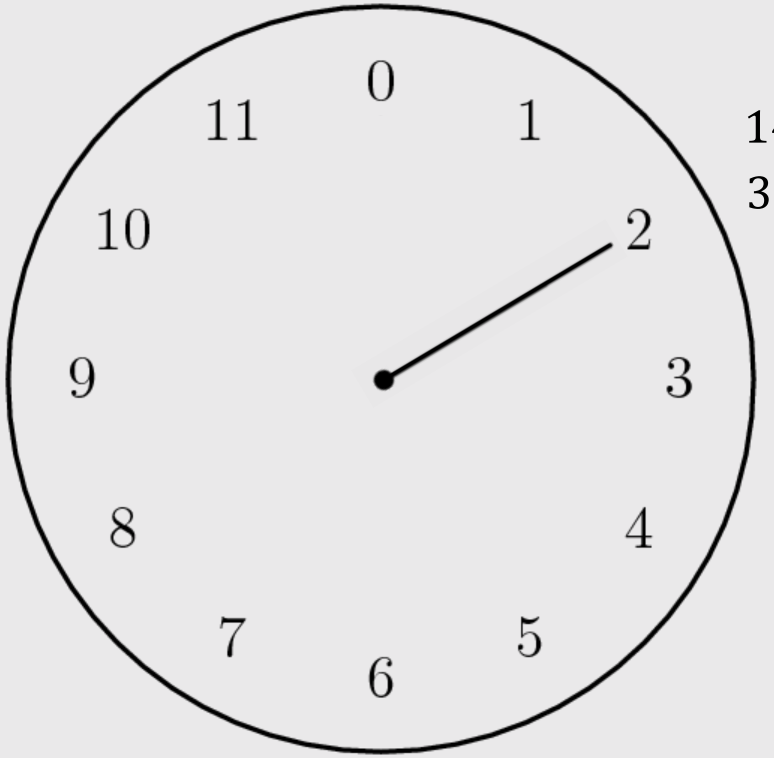
If there is an integer k such that

$$a - b = kn$$

$$38 - 14 = 2 \times 12$$

$$38 \equiv 14 \ (mod\ 12)$$

$$38\ mod\ 12 = 14\ mod\ 12$$



$14\ mod\ 12 = 2$

$38\ mod\ 12 = 2$

Multiplicative inverse

For a number x, there is a number $\frac{1}{x}$ which when multiplied by x yields 1

Example:

The multiplicative inverse of 8 is $\frac{1}{8}$

$$8 \times \frac{1}{8} = 1$$

Modular multiplicative inverse

For in integer a there is an integer x such that the product ax is congruent to 1 with respect to the modulus n

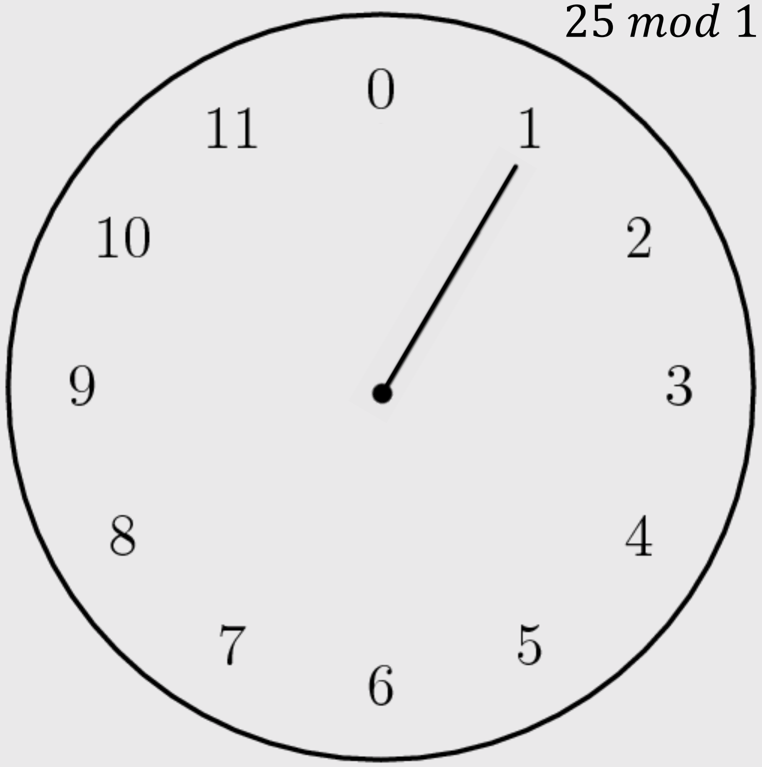
$$ax \equiv 1 \ (mod\ n)$$

Example:

$$a = 5 \quad n = 12 \quad x = ?$$

$$5 \times 5 \equiv 1 \ (mod\ 12)$$

$25\ mod\ 12 = 1$



Modulo operations

$$(a + b) \; mod \; n = [(a \; mod \; n) + (b \; mod \; n)] \; mod \; n$$

$$ab \; mod \; n = [(a \; mod \; n)(b \; mod \; n)] \; mod \; n$$

$$a^x \; mod \; n = (a \; mod \; n)^x \; mod \; n$$

Euler's totient function

aka. Euler's **phi** function

$\varphi(n)$ Counts the positive integers less than n that **do not share** a common factor greater than 1 with n

Example:

$\varphi(8) = ?$

1
2
3
4
5
6
7

These integers are **coprime** to n

$\varphi(8) = 4$

Euler's totient function

aka. Euler's **phi** function

$\varphi(n)$ Counts the positive integers less than n that do not share a common factor greater than 1 with n

Example:

$\varphi(7) = ?$

1
2
3
4
5
6

$\varphi(7) = 6$

Euler's totient function

aka. Euler's **phi** function

$\varphi(n)$ Counts the positive integers less than n whose gcd with n are equal to 1

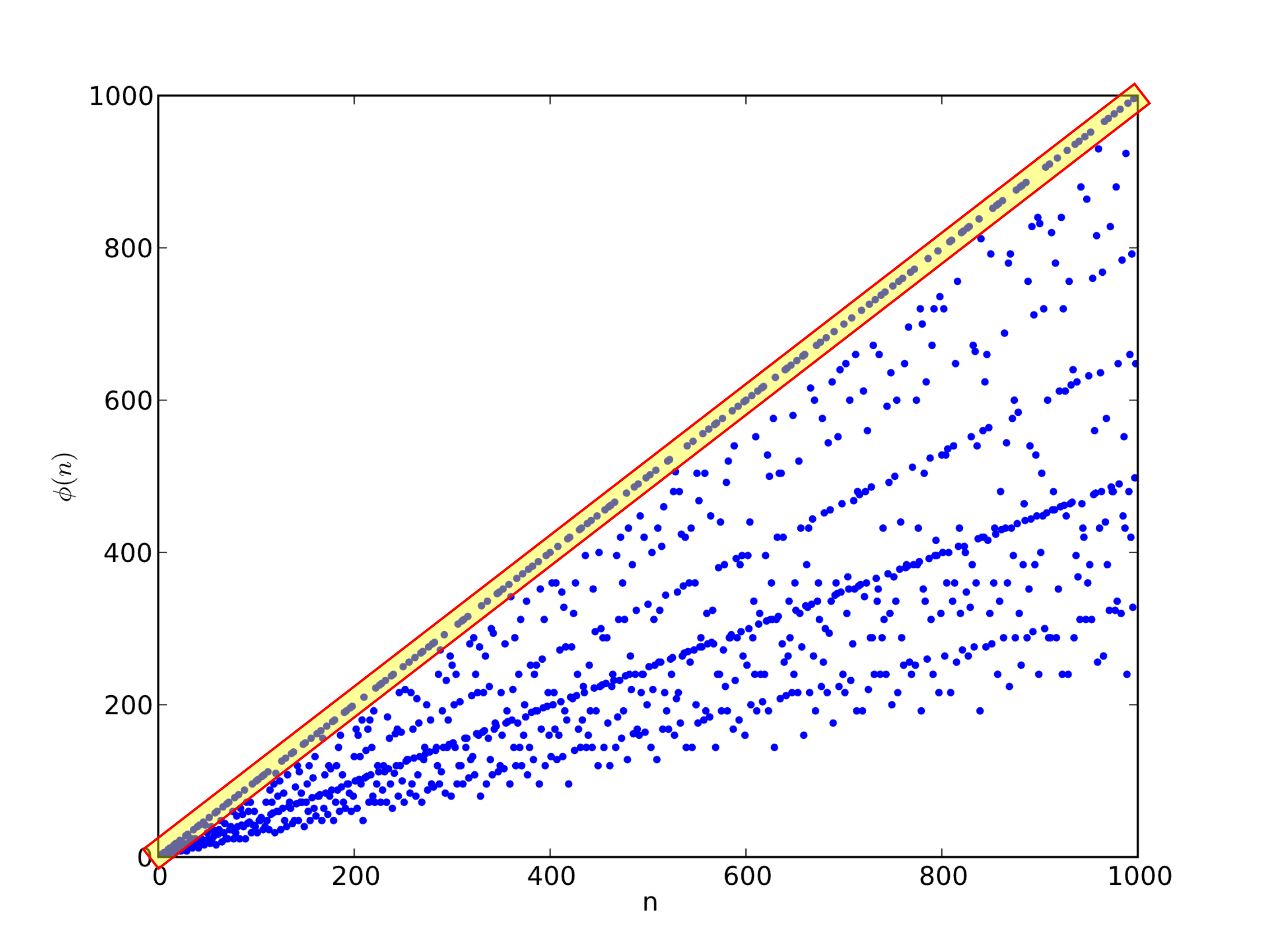
Example:

$$\varphi(7) = ?$$

1	$\gcd(1,7)$	$= 1$
2	$\gcd(2,7)$	$= 1$
3	$\gcd(3,7)$	$= 1$
4	$\gcd(4,7)$	$= 1$
5	$\gcd(5,7)$	$= 1$
6	$\gcd(6,7)$	$= 1$

$$\varphi(7) = 6$$

Euler's totient function



Euler's totient function

aka. Euler's **phi** function

$\varphi(n)$ of any prime number n is equal to $n - 1$

Example:

$$\varphi(13) = 13 - 1 = 12$$
$$\varphi(17) = 17 - 1 = 16$$
$$\varphi(31) = 31 - 1 = 30$$
$$\varphi(21377) = 21377 - 1 = 21376$$

phi of any **prime** is **EASY** to compute

Euler's totient function

Euler's totient function is a **multiplicative function**, meaning that if two numbers m and n are coprime, then

$$\varphi(mn) = \varphi(m)\,\varphi(n)$$

coprime: the number m and n do not share a common factor

Euler's totient function

Given **two prime** numbers p and q

$$n = pq$$

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

Example:

$$91 = 7 \times 13$$
$$\varphi(91) = (7-1)(13-1) = 6 \times 12 = 72$$

Euler's theorem

A relationship between the **phi** function and modular exponentiation

Euler's theorem states that if a and n are coprime then

$$a^{\varphi(n)} \equiv 1 \ (mod\ n)$$

Example:

$$a = 5 \ , \ n = 8$$

$$5^{\varphi(8)} \equiv 1 \ (mod\ 8)$$

$$5^4 \equiv 1 \ (mod\ 8)$$

$$625 \equiv 1 \ (mod\ 8)$$

Solving for the private key d (trap door!)

- Given that $1^k = 1$
- $m^{\varphi(n)} \equiv 1 \ (mod\ n)$
- $(m^{\varphi(n)})^k \equiv 1 \ (mod\ n)$
- $m^{k\varphi(n)} \equiv 1 \ (mod\ n)$
- Multiply both sides by m
- $m \cdot m^{k\varphi(n)} \equiv m \ (mod\ n)$
- $m^{k\varphi(n)+1} \equiv m \ (mod\ n)$
- $m^{ed} \equiv m \ (mod\ n)$
- $ed = k\varphi(n) + 1$
- $d = \frac{k\varphi(n)+1}{e}$

RSA (Rivest – Shamir – Adleman)

Generate the public key (e, n):

1. Select two large prime numbers p and q
2. Calculate $n = pq$
3. Calculate $\varphi(n) = (p - 1)(q - 1)$
4. Chose e such that
    1. Must be prime
    2. $1 < e < \varphi(n)$
    3. Must be coprime with $\varphi(n)$

Generate the private key (d)

1. Calculate d such that $d = \frac{k\varphi(n)+1}{e}$

Is RSA Safe?

- RSA Factoring Challenge
- https://en.wikipedia.org/wiki/RSA_numbers

RSA-260 [edit]

RSA-260 has 260 decimal digits (862 bits), and has not been factored so far.

```
RSA-260 = 22112825529529666435281085255026230927612089502470015394413748319128822941402
          001986512729726569746599085900330031400051170742204560859276357953757185954
          298838958709229238491006703034124620545784566413664540684214361293017694020
          8463910658759147942514351444581 99
```