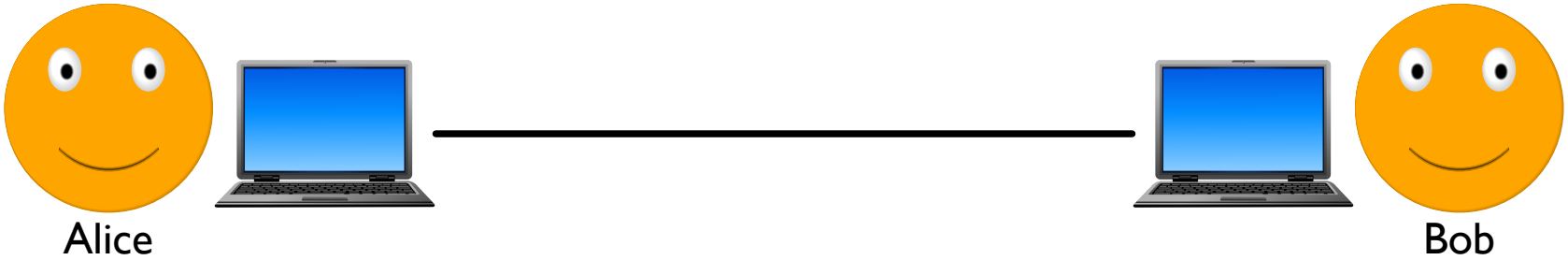
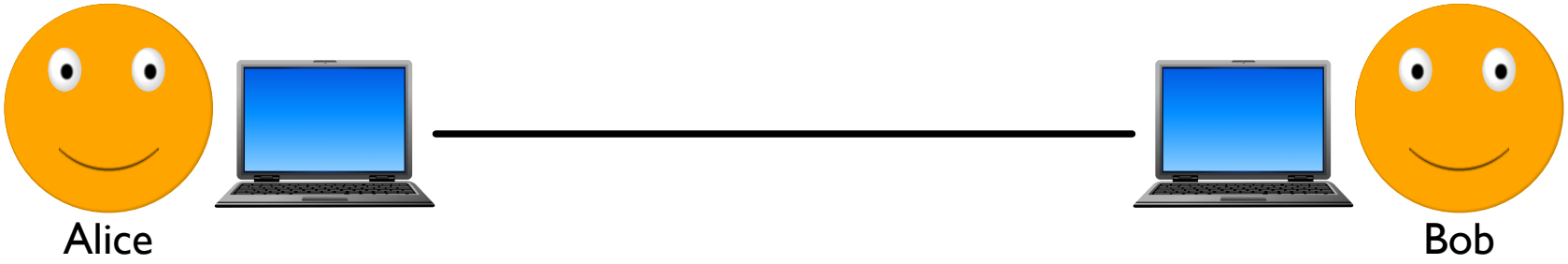
Cryptography

Cryptography: secure communication in the presence of adversaries



Cryptography

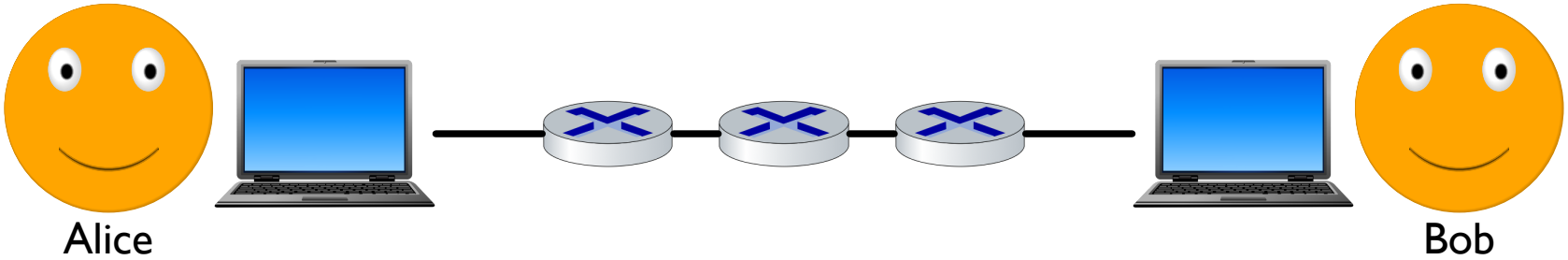
Cryptography: secure communication in the presence of adversaries



In this class, *crypto* is short for *cryptography*, not *cryptocurrency* or *cryptoanalysis*

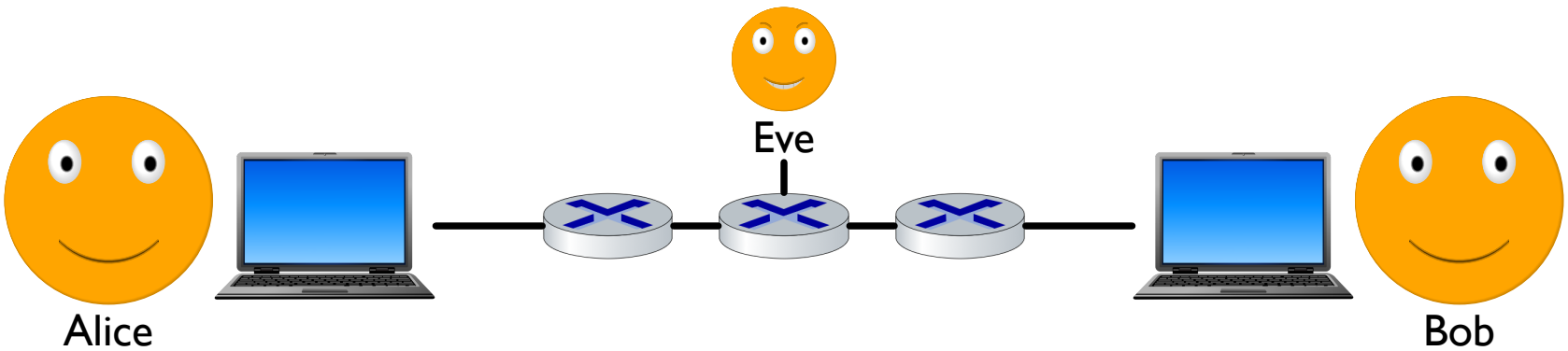
Cryptography

Cryptography: secure communication in the presence of adversaries



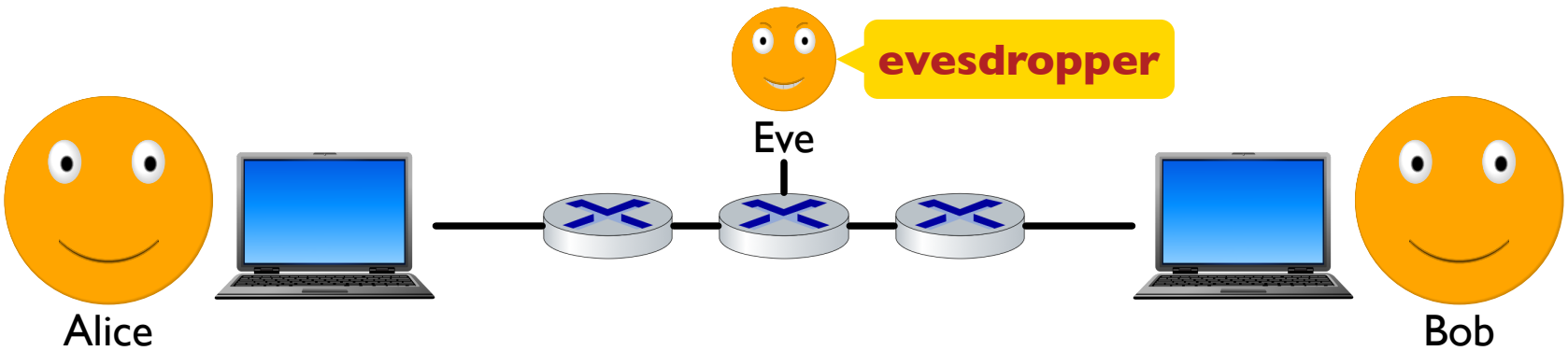
Cryptography

Cryptography: secure communication in the presence of adversaries



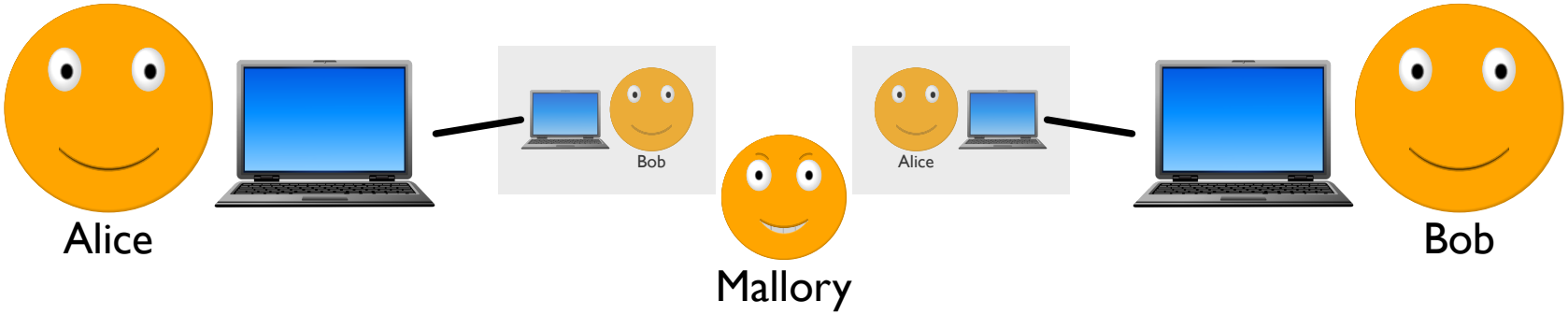
Cryptography

Cryptography: secure communication in the presence of adversaries



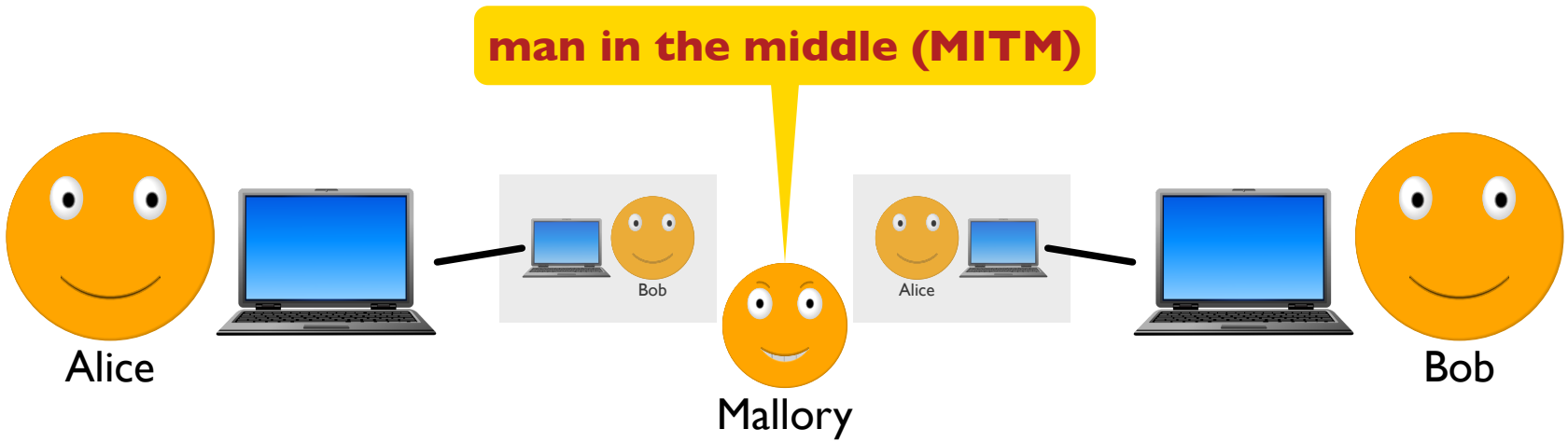
Cryptography

Cryptography: secure communication in the presence of adversaries

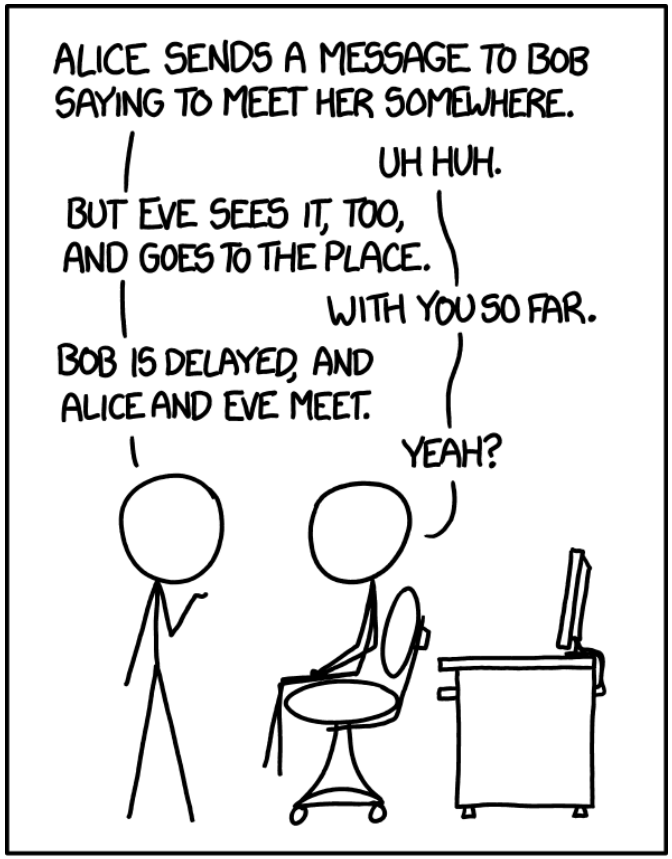


Cryptography

Cryptography: secure communication in the presence of adversaries



Alice and Bob



I'VE DISCOVERED A WAY TO GET COMPUTER SCIENTISTS TO LISTEN TO ANY BORING STORY.

https://xkcd.com/1323/

Cryptography Application Goals

Confidentiality

Cryptography Application Goals

Confidentiality: only intended recipient can read a message

Cryptography Application Goals

Confidentiality: only intended recipient can read a message

Integrity

Cryptography Application Goals

Confidentiality: only intended recipient can read a message

Integrity: received message is unchanged from sender

Cryptography Application Goals

Confidentiality: only intended recipient can read a message

Integrity: received message is unchanged from sender

Authenticity

Cryptography Application Goals

Confidentiality: only intended recipient can read a message

Integrity: received message is unchanged from sender

Authenticity: identity of each communicating party can be confirmed

Cryptography Application Goals

Confidentiality: only intended recipient can read a message

Integrity: received message is unchanged from sender

Authenticity: identity of each communicating party can be confirmed

Non-repudiation

Cryptography Application Goals

Confidentiality: only intended recipient can read a message

Integrity: received message is unchanged from sender

Authenticity: identity of each communicating party can be confirmed

Non-repudiation: parties cannot deny previous commitments

Cryptography Application Goals

Confidentiality: only intended recipient can read a message

Integrity: received message is unchanged from sender

Authenticity: identity of each communicating party can be confirmed

Non-repudiation: parties cannot deny previous commitments

Assume that attackers are capable of evesdropping,
are capable of MITM,
know your algorithms, and
have NSA-scale compute power

Ciphers, Algorithms, Keys

Alice

Bob

```
MEET ME AT THE CLOCK TOWER
```

Ciphers, Algorithms, Keys

Alice

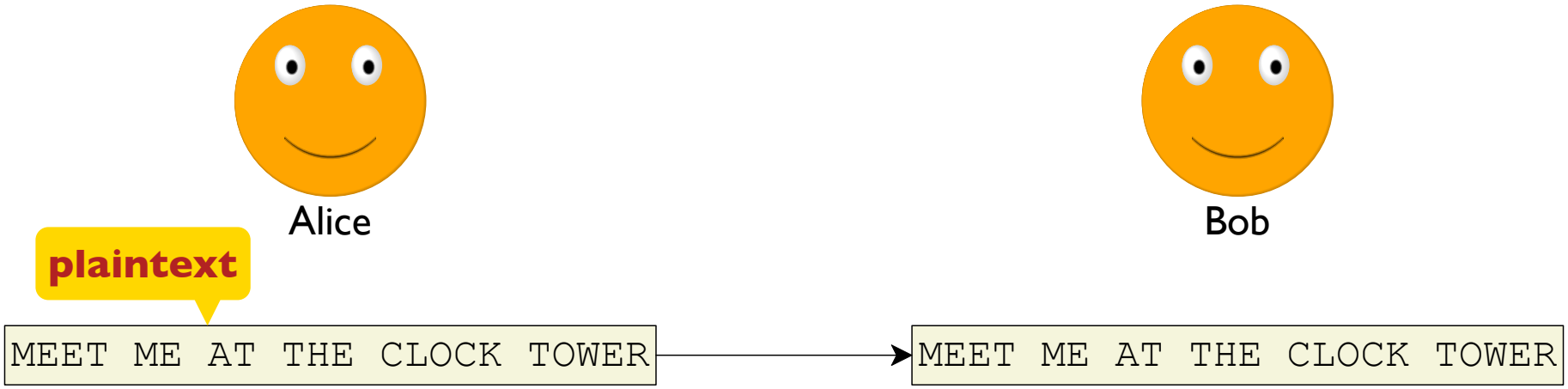
Bob

```
MEET ME AT THE CLOCK TOWER
```

→

```
MEET ME AT THE CLOCK TOWER
```

Ciphers, Algorithms, Keys



Ciphers, Algorithms, Keys

Alice

Bob

plaintext

```
MEET ME AT THE CLOCK TOWER
```

```
ZRRG ZR NG GUR PYBPX GBJRE
```

```
MEET ME AT THE CLOCK TOWER
```

Ciphers, Algorithms, Keys

Alice

Bob

plaintext

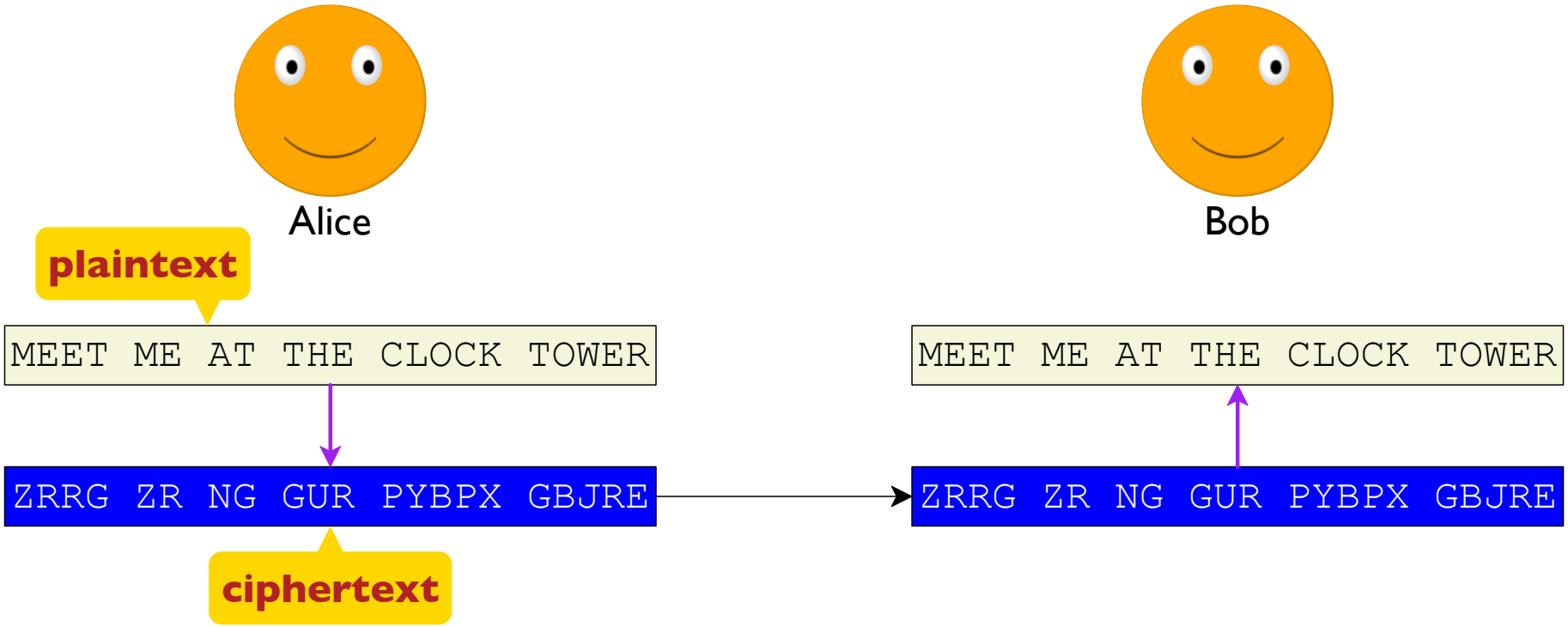
```
MEET ME AT THE CLOCK TOWER
```

```
ZRRG ZR NG GUR PYBPX GBJRE
```

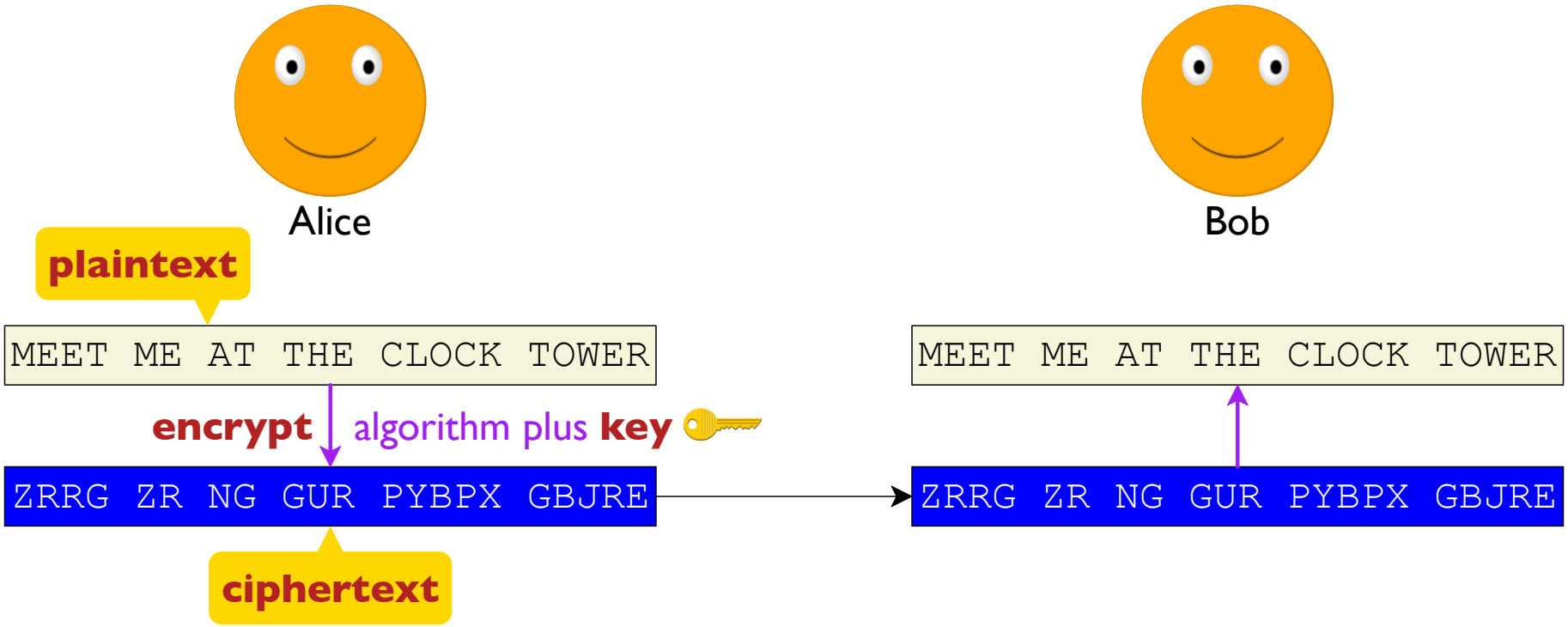
ciphertext

```
MEET ME AT THE CLOCK TOWER
```

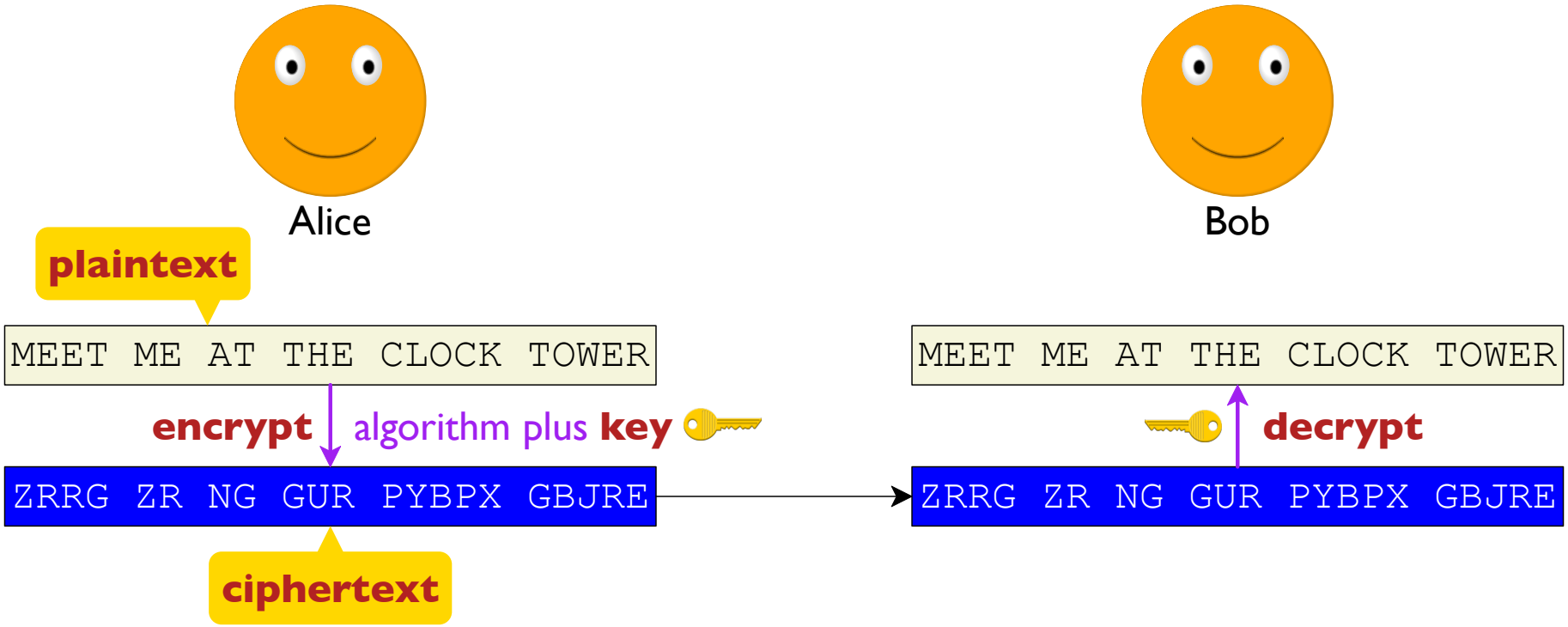
Ciphers, Algorithms, Keys



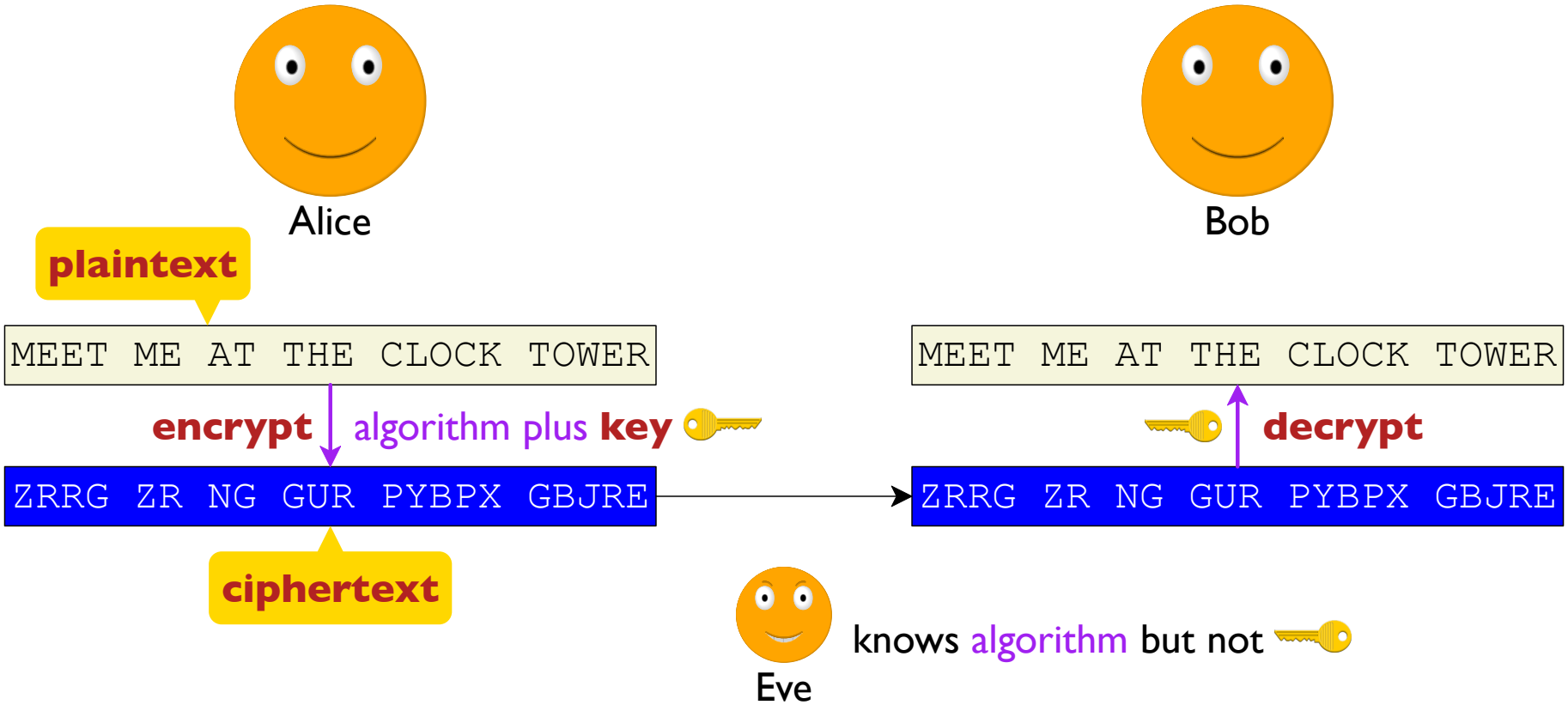
Ciphers, Algorithms, Keys



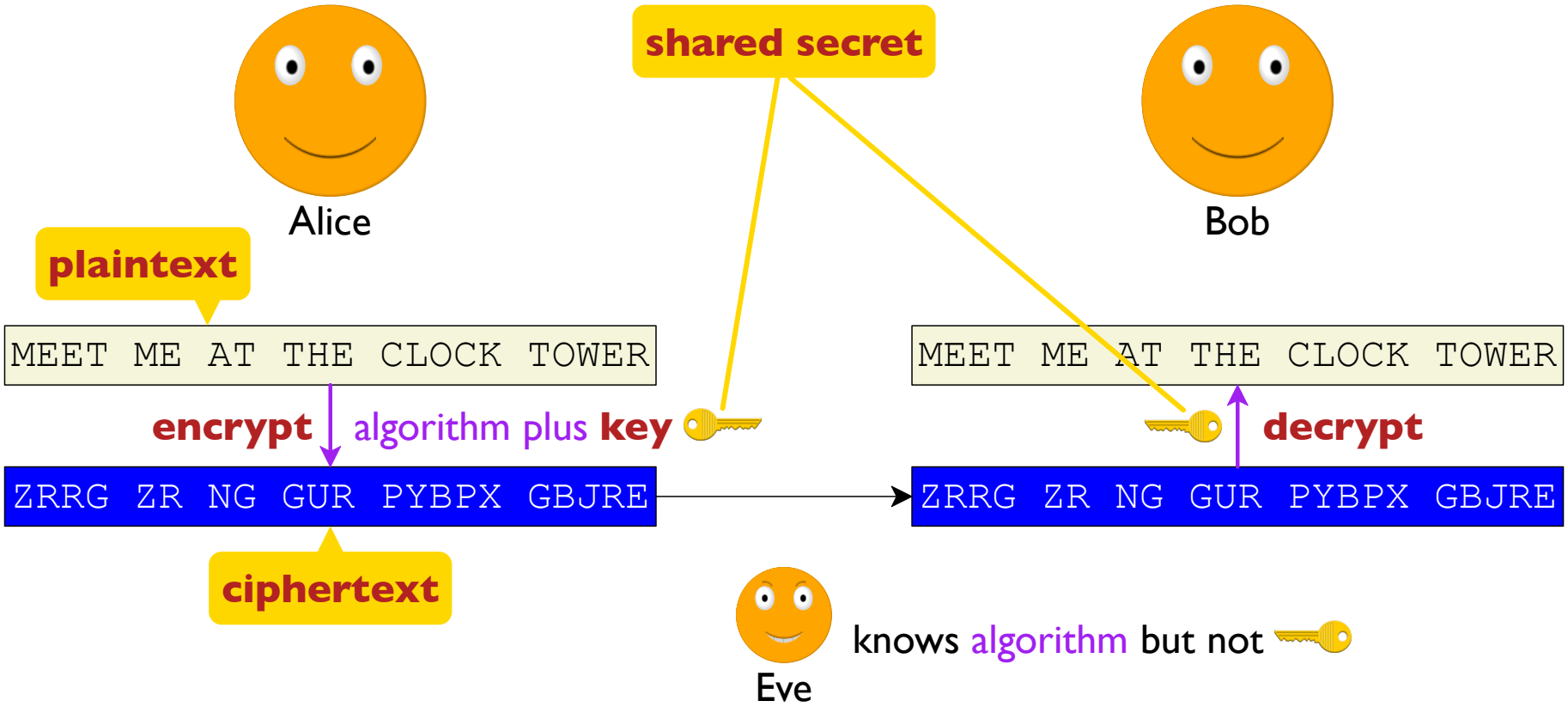
Ciphers, Algorithms, Keys



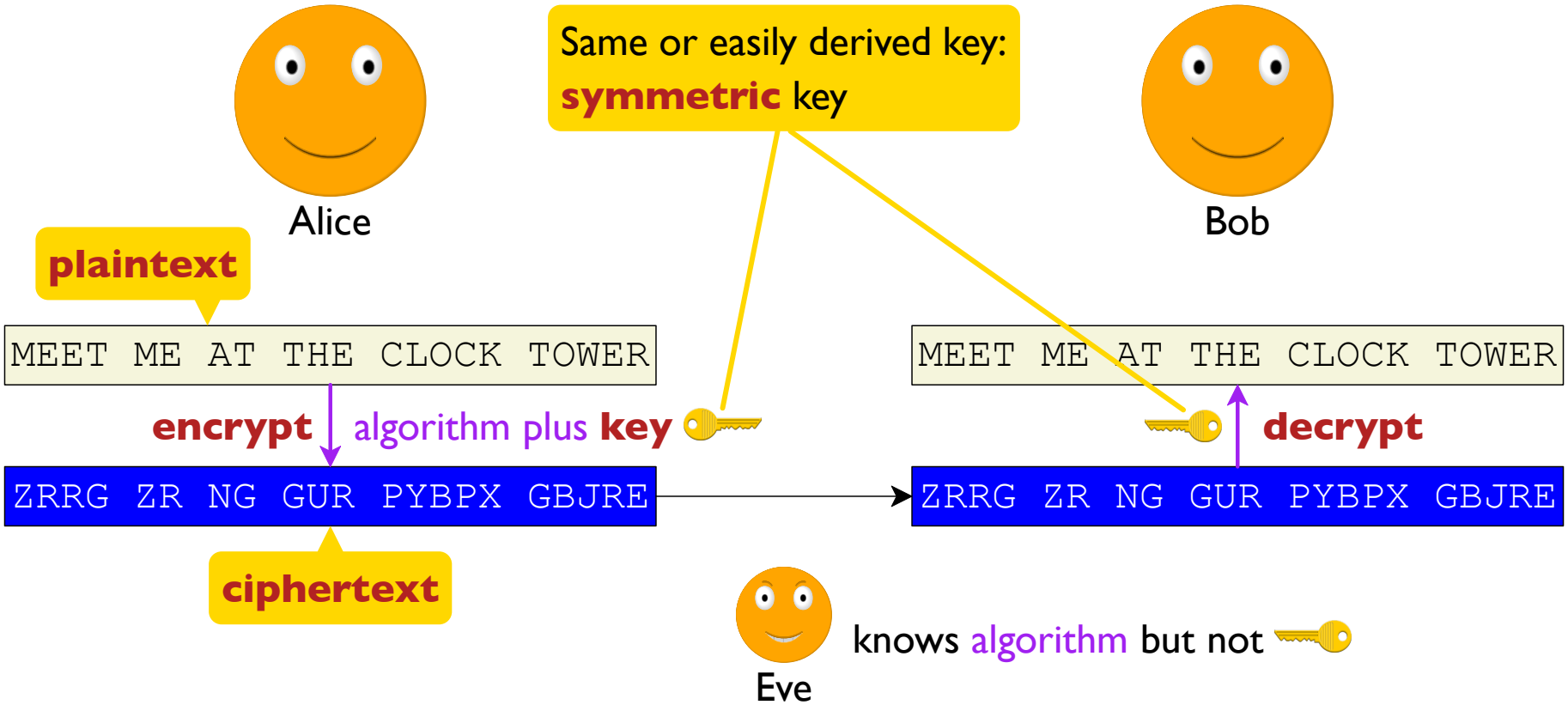
Ciphers, Algorithms, Keys



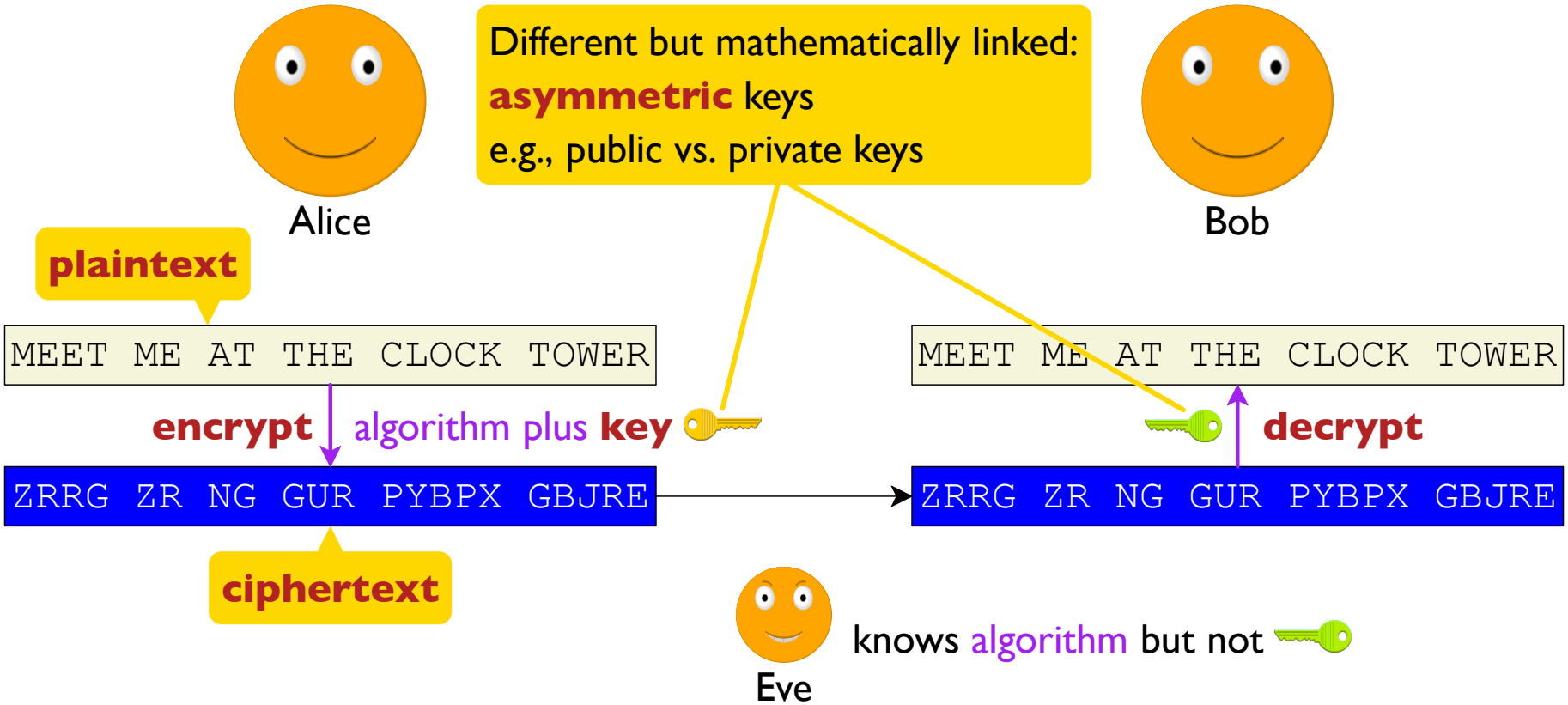
Ciphers, Algorithms, Keys



Ciphers, Algorithms, Keys



Ciphers, Algorithms, Keys



Encrypt and Decrypt

Encryption can use any function

Enc(plaintext, ) = ciphertext

that can be reversed by a decryption function

Dec(ciphertext, ) = plaintext

Encrypt and Decrypt

Encryption can use any function

function in the mathematical sense
i.e., deterministic

Enc(plaintext, ) = ciphertext

that can be reversed by a decryption function

Dec(ciphertext, ) = plaintext

Encrypt and Decrypt

Encryption can use any function

function in the mathematical sense
i.e., deterministic

Enc(plaintext, ) = ciphertext

that can be reversed by a decryption function

Dec(ciphertext, ) = plaintext

Goals:

functions that make ciphertext look random

functions with enough  s to making guessing impractical

Encrypt and Decrypt

Encryption can use any function

function in the mathematical sense
i.e., deterministic

Enc(plaintext, ) = ciphertext

that can be reversed by a decryption function

Dec(ciphertext, ) = plaintext

Goals:

functions that make ciphertext look random

functions with enough  s to making guessing impractical

A good algorithm is one where this **brute force** strategy is the only one

Attack Modes

Ciphertext only: attacker has only ciphertext to work from, but maybe many of them

Known plaintext: attacker has an example plaintext and matching ciphertext to work from

Chosen plaintext: attacker can get its own plaintext encoded to its ciphertext

IFMMP XPSME

Substitution

IFMMP XPSME

HELLO WORLD

Substitution

IFMMP XPSME

HELLO WORLD

A	B
B	C
C	D
D	E
E	F
F	G
G	H
H	I
…	…
X	Y
Y	Z
Z	A

Substitution

JGNNQ YQTNF

Substitution

JGNNQ YQTNF

HELLO WORLD

Substitution

JGNNQ YQTNF

HELLO WORLD

A	C
B	D
C	E
D	F
E	G
F	H
G	I
H	J
…	…
X	Z
Y	A
Z	B

Substitution

JGNNQ YQTNF

HELLO WORLD

A	C
B	D
C	E
D	F
E	G
F	H
G	I
H	J
…	…
X	Z
Y	A
Z	B

Ceasar cipher

 = 2

 = -2

Substitution

URYYB JBEYQ

HELLO WORLD

A	N
B	O
C	P
D	Q
E	R
F	S
G	T
H	U
...	...
X	K
Y	L
Z	M

Ceasar cipher

 = 13

 = -13

a.k.a. ROT13

Substitution

URYYB JBEYQ

HELLO WORLD

A	N
B	O
C	P
D	Q
E	R
F	S
G	T
H	U
...	...
X	K
Y	L
Z	M

Ceasar cipher

 = 13

 = -13

a.k.a. ROT13

With only 26 possible keys guessing is easy

Substitution

URYYB JBEYQ

HELLO WORLD

A	N
B	O
C	P
D	Q
E	R
F	S
G	T
H	U
...	...
X	K
Y	L
Z	M

Ceasar cipher

 = 13

 = -13

a.k.a. ROT13

Can treat N letters in a row as base-26 digits:

$$\mathtt{HEL} = 8 \times 26^2 + 5 \times 26 + 12$$

That gives us 26^N keys

Substitution

URYYB JBEYQ

Substitution by itself is weak, because it preserves patterns:

- Commonly used letters ⇒ commonly used replacements
- Local patterns like “ll” in “hello” ⇒ local patterns in ciphertext

Permutation

A permutation can break up local patterns:

```
MEET ME AT THE CLOCK TOWER
```

M	E	E	T
	M	E	
A	T		T
H	E		C
L	O	C	K
	T	O	W
E	R	□	□

```
M AHL EEMTEOTREE  COT TCKW
```

Permutation

A permutation can break up local patterns:

```
MEET ME AT THE CLOCK TOWER
```

M	E	E	T
	M	E	
A	T		T
H	E		C
L	O	C	K
	T	O	W
E	R	□	□

= number of columns

```
M AHL EEMTEOTREE  COT TCKW
```

Permutation

A permutation can break up local patterns:

```
MEET ME AT THE CLOCK TOWER
```

M	E	E	T
	M	E	
A	T		T
H	E		C
L	O	C	K
	T	O	W
E	R	□	□

= number of columns

```
M AHL EEMTEOTREE  COT TCKW
```

Other examples of permutations:
- shifting with wraparound
- shuffling deterministically

Substitution plus Permutation

Combining substitution and permutation is even better:

MEET ME AT THE CLOCK TOWER

NFFU NF BU UIF DMPDL UPXFS

N	F	F	U
	N	F	
B	U		U
I	F		D
M	P	D	L
	U	P	X
F	S	□	□

Z NUY RRZGRBGERR PBG GPXJ

Substitution plus Permutation

Combining substitution and permutation is even better:

```
MEET ME AT THE CLOCK TOWER
```

```
NFFU NF BU UIF DMPDL UPXFS
```

N	F	F	U
	N	F	
B	U		U
I	F		D
M	P	D	L
	U	P	X
F	S	□	□

is ⟨columns, rotation⟩

```
Z NUY RRZGRBGERR  PBG GPXJ
```

Substitution plus Permutation

Combining substitution and permutation is even better:

MEET ME IN THE CLOCK TOWER

NFFU NF BU UIF DMPDL UPXFS

N	F	F	U
	N	F	
J	O		U
I	F		D
M	P	D	L
	U	P	X
F	S	□	□

is ⟨columns, rotation⟩

Z VUY RRZARBGERR PBG GPXJ

Substitution plus Permutation

Combining substitution and permutation is even better:

MEET ME IN THE CLOCK TOWER

NFFU NF BU UIF DMPDL UPXFS

N	F	F	U
	N	F	
J	O		U
I	F		D
M	P	D	L
	U	P	X
F	S	□	□

Still, small changes in plaintext trigger only small changes in ciphertext

is ⟨columns, rotation⟩

Z VUY RRZARBGERR PBG GPXJ

Chaining

Avalance effect via running total mod 27 ⇒ each position affects all later

	M	E	E	T		M	E		A	T		T	H	E		C	L	O	C	K		T	O	W	E	R
5	+ 13	+ 5	+ 5	+ 20	+ 0	+ 13	+ 5	+ 0	+ 1	+ 20	+ 0	+ 20	+ 8	+ 5	+ 0	+ 3	+ 12	+ 15	+ 3	+ 11	+ 0	+ 20	+ 15	+ 23	+ 5	+ 18
	= 18	= 23	= 1	= 21	= 21	= 7	= 12	= 12	= 13	= 6	= 6	= 26	= 7	= 12	= 12	= 15	= 0	= 15	= 18	= 2	= 2	= 22	= 10	= 6	= 11	= 2
	R	W	A	U	U	G	L	L	M	F	F	Z	G	L	L	O		O	R	B	B	V	J	F	K	B

Chaining

Avalance effect via running total mod 27 ⇒ each position affects all later

	M	E	E	T		M	E		A	T		T	H	E		C	L	O	C	K		T	O	W	E	R
5	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
	13	5	5	20	0	13	5	0	1	20	0	20	8	5	0	3	12	15	3	11	0	20	15	23	5	18
	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=
	18	23	1	21	21	7	12	12	13	6	6	26	7	12	12	15	0	15	18	2	2	22	10	6	11	2
	R	W	A	U	U	G	L	L	M	F	F	Z	G	L	L	O		O	R	B	B	V	J	F	K	B

Can decrypt because + is reversible
The xor operation has the same property

Chaining

Avalance effect via running total mod 27 ⇒ each position affects all later

	M	E	E	T		M	E		A	T		T	H	E		C	L	O	C	K		T	O	W	E	R
5	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
	13	5	5	20	0	13	5	0	1	20	0	20	8	5	0	3	12	15	3	11	0	20	15	23	5	18
	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=
	18	23	1	21	21	7	12	12	13	6	6	26	7	12	12	15	0	15	18	2	2	22	10	6	11	2
	R	W	A	U	U	G	L	L	M	F	F	Z	G	L	L	O		O	R	B	B	V	J	F	K	B

Can decrypt because + is reversible
The xor operation has the same property

but needs to be combined
with other techniques

Chaining

Avalance effect via running total mod 27 ⇒ each position affects all later

	M	E	E	T		M	E		I	N		T	H	E		C	L	O	C	K		T	O	W	E	R
 5	+ 13	+ 5	+ 5	+ 20	+ 0	+ 13	+ 5	+ 0	+ 1	+ 20	+ 0	+ 20	+ 8	+ 5	+ 0	+ 3	+ 12	+ 15	+ 3	+ 11	+ 0	+ 20	+ 15	+ 23	+ 5	+ 18
	= 18	= 23	= 1	= 21	= 21	= 7	= 12	= 12	= 13	= 6	= 6	= 26	= 7	= 12	= 12	= 15	= 0	= 15	= 18	= 2	= 2	= 22	= 10	= 6	= 11	= 2
	R	W	A	U	U	G	L	L	U	H	H	A	I	N	N	Q	B	Q	T	D	D	X	L	H	M	D

Chaining

Could run it twice to make every position affect all positions...

	M	E	E	T		M	E		I	N		T	H	E		C	L	O	C	K		T	O	W	E	R
5	+ 13	+ 5	+ 5	+ 20	+ 0	+ 13	+ 5	+ 0	+ 1	+ 20	+ 0	+ 20	+ 8	+ 5	+ 0	+ 3	+ 12	+ 15	+ 3	+ 11	+ 0	+ 20	+ 15	+ 23	+ 5	+ 18
	+ 18	+ 23	+ 1	+ 21	+ 21	+ 7	+ 12	+ 12	+ 13	+ 6	+ 6	+ 26	+ 7	+ 12	+ 12	+ 15	+ 0	+ 15	+ 18	+ 2	+ 2	+ 22	+ 10	+ 6	+ 11	+ 2
	= 20	= 16	= 17	= 11	= 5	= 12	= 24	= 9	= 22	= 1	= 7	= 6	= 13	= 25	= 10	= 25	= 25	= 13	= 4	= 6	= 8	= 3	= 13	= 19	= 3	= 5
	V	R	S	M	G	N	Z	K	E	M	U	V	D	R	E	V	X	N	G	K	O	L	X	E	R	V

Chaining plus Substitution plus Permutation

MEET ME AT THE CLOCK TOWER

accumulate

TPQKELXIVAGFMYJYYMDFHCMSCE

substitute

GCDXRYKVINTSZLWLLZQSUPZFPR

permute

GRIZLUPCYNLZPRDKTWQZXVSLSF

Chaining plus Substitution plus Permutation

MEET ME IN THE CLOCK TOWER

accumulate

VRSMGNZKEMUVDREVXNGKOLXERV

substitute

IEFZTAMXRZHIQERIKATXBYKREI

permute

ITRQKBEEAZEAYIFMHRTKZXIIXR

Key Size

Substitution, permutation, and chaining are useful building blocks, and our example combination generates results that *look* random, but there's an easy way to see that it's insecure

 = ⟨rotation, columns, init⟩

Key Size

Substitution, permutation, and chaining are useful building blocks, and our example combination generates results that *look* random, but there's an easy way to see that it's insecure

 = ⟨rotation, columns, init⟩

Assuming that up to 32 columns makes sense:

$$26 \times 32 \times 27 \quad = \quad 22{,}464 \text{ possible keys}$$

Key Size

Substitution, permutation, and chaining are useful building blocks, and our example combination generates results that *look* random, but there's an easy way to see that it's insecure

 = ⟨rotation, columns, init⟩

Assuming that up to 32 columns makes sense:

$$26 \times 32 \times 27 \quad = \quad 22{,}464 \text{ possible keys}$$

So, **key size** is going to be an important metric

Block Size

For a long enough message, typically you want to encode only small parts at a time, as opposed to keeping the whole message in memory to rearrange all the bytes

Block Size

For a long enough message, typically you want to encode only small parts at a time, as opposed to keeping the whole message in memory to rearrange all the bytes

As our permutation example shows, though, it's useful to be able to mix large chunks to create confusion

Block Size

For a long enough message, typically you want to encode only small parts at a time, as opposed to keeping the whole message in memory to rearrange all the bytes

As our permutation example shows, though, it's useful to be able to mix large chunks to create confusion

So, **block size** is going to be an important metric