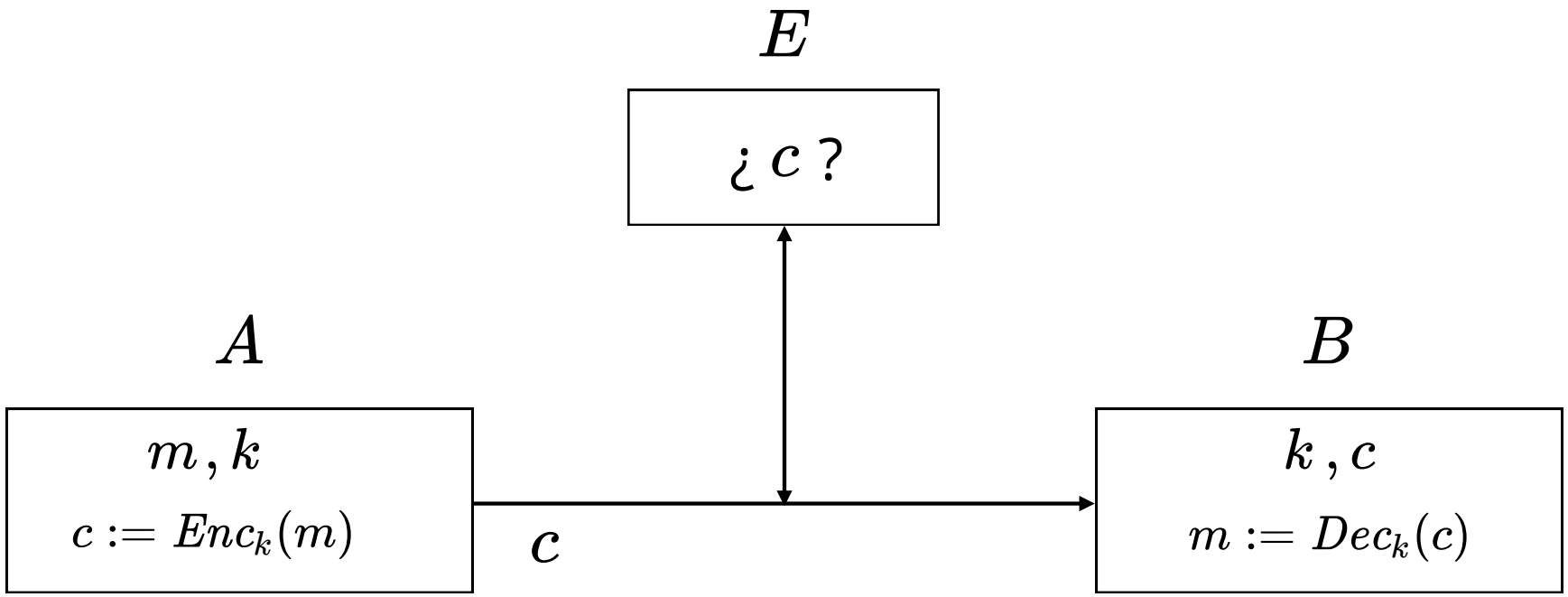
IIC3253

OTP y perfect secrecy

Cifrado (simétrico)



Cifrado del César

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

HOLA MUNDO
EMIX JRKAM

MANDEN BITCOINS A UCRANIA
JXKABK YFQZMFKP X RZOXKFX

¿Problemas?

Cifrado del César + llave

Llave = shift **7**

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

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

HOLA MUNDO
AIET FÑGWI

MANDEN BITCOINS A UCRANIA

FTGWXG UBNVIBGM T ÑVLTGBT

¿Problemas?

La probabilidad de que un atacante "seleccione" o "adivine" la llave correcta debe ser muy baja.

$\Rightarrow$ El espacio de llaves posibles debe ser muy (muy) grande

¿Cómo podríamos agrandar el espacio de llaves siguiendo la idea de "sustituir"?

Shift $\rightarrow$ Permutación

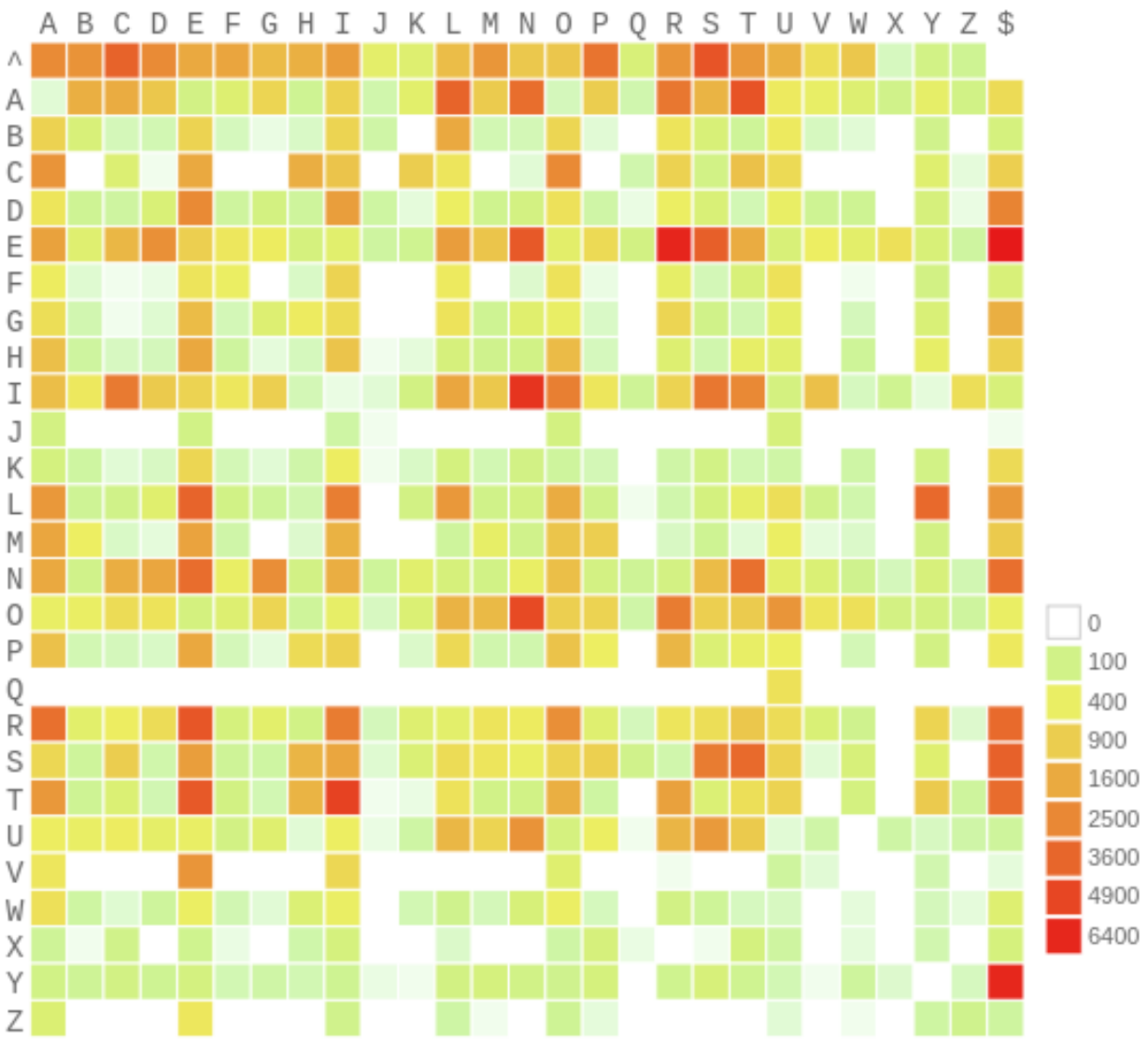
```
A B C D E F G H I J K L M N Ñ O P Q R S T U V W X Y Z
P Q O W I E U R Y T L K A J S H D F G Ñ M Z N X B C V
```

```
HOLA MUNDO
RHKP AZJWH
```

¿Cuántas llaves posibles?

$$27! = 10,888,869,450,418,352,160,768,000,000$$

¿Es este un buen cifrado?


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 $
^ 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
0
100
400
900
1600
2500
3600
4900
6400

Un espacio de llaves grande es necesario, no suficiente.

ONE-TIME PAD (OTP)

Operación Módulo

(Recordatorio)

Dados $a, n \in \mathbb{Z}$, existe un único par de elementos $(q, r) \in \mathbb{Z}^2$ tal que:

$$0 \leq r < |n|$$

$$a = q \cdot n + r$$

Cuociente

Resto

Decimos entonces que $a \bmod n = r$ y que $a \equiv r \bmod n$

Operación Módulo

(Ejemplos)

$$10 \mod 3 = 1$$

$$28 \mod 8 = 4$$

$$6 \mod -20 = 6$$

$$-6 \mod -20 = 14$$

Siempre esperaríamos que

$$n \cdot \left\lfloor \frac{a}{n} \right\rfloor + (a \bmod n) = a$$

Programando, esto se ve como

```
n * (a / n) + a % n = a
```

División entera

```python
# Python
print("La división entera entre 6 y -20 es:")
print(6 // -20)
```

Output: -1

```cpp
// C++
#include <iostream>
using namespace std;

int main() {
    cout << "La división entera entre 6 y -20 es: " << endl;
    cout << (6 / 20) << endl;
    return 0;
}
```

Output: 0

Esperamos que

```
n * (a / n) + a % n = a
```

```
-20 * (6 / -20) + 6 % -20 = 6
```

Python: `-20 * -1 + 6 % -20 = 6` $\Rightarrow$ `6 % -20 = -14`

C++: `-20 * 0 + 6 % -20 = 6` $\Rightarrow$ `6 % -20 = 6`

Operación Módulo

Dados $a, n \in \mathbb{Z}$, existe un único par de elementos $(q, r) \in \mathbb{Z}^2$ tal que:

$$0 \leq r < |n|$$

$$a = q \cdot n + r$$

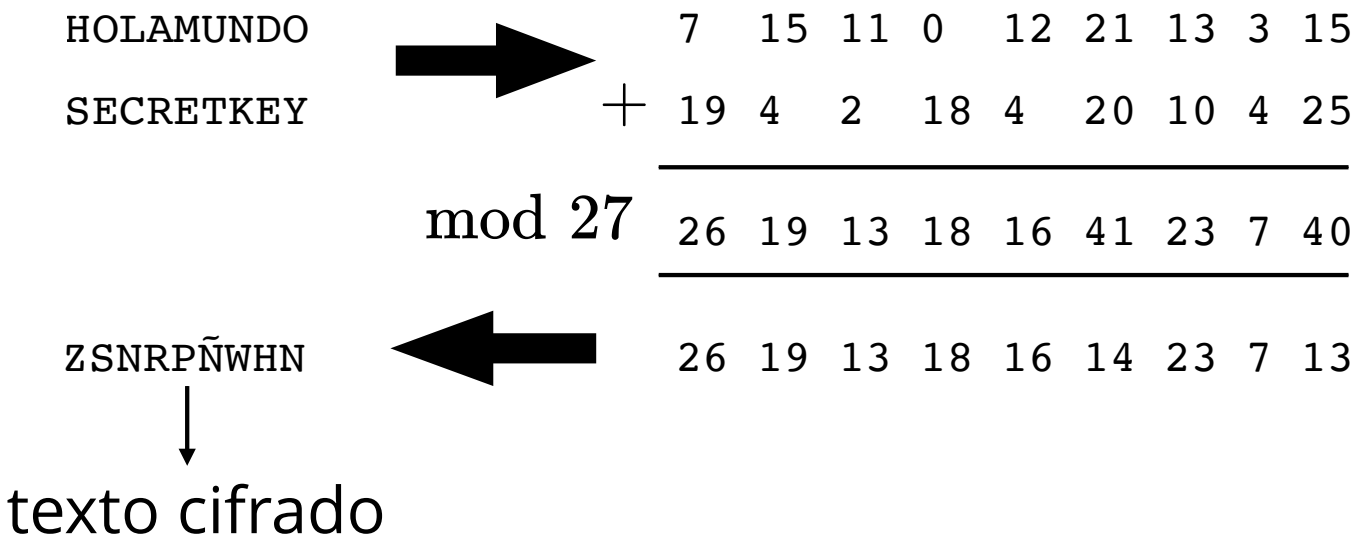
Decimos entonces que $a \bmod n = r$ y que $a \equiv r \bmod n$

ONE-TIME PAD (OTP)

Partimos enumerando las letras

A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
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	26

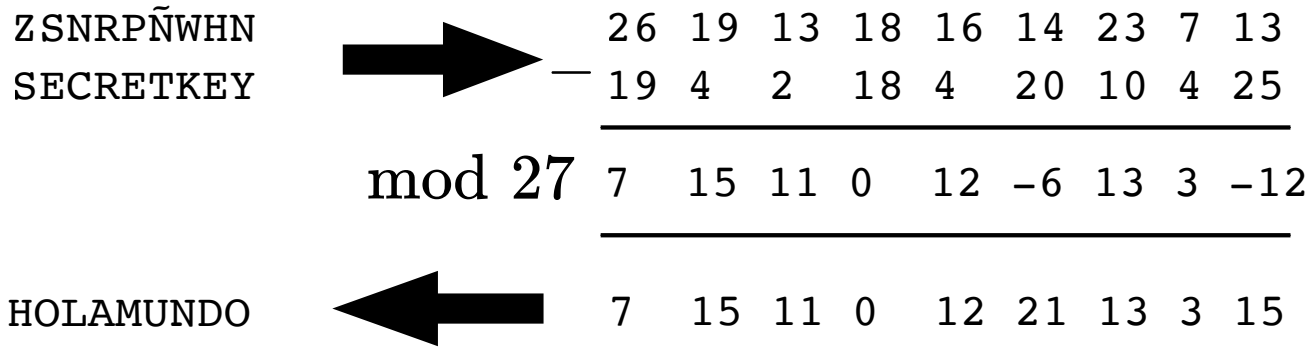
Para enviar un mensaje de largo ℓ
necesitaremos una llave de largo ℓ



$$Enc_{\text{SECRETKEY}}(\text{HOLAMUNDO}) = \text{ZSNRPÑWHN}$$

¿Cómo decriptar?

$$Dec_{\text{SECRETKEY}}(\text{ZSNRPÑWHN}) = \text{HOLAMUNDO}$$



A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
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	26