

# Sistemas Numericos

Entendiendo sistema binario

# Sistema decimal

¿Cual es el último numero?

9

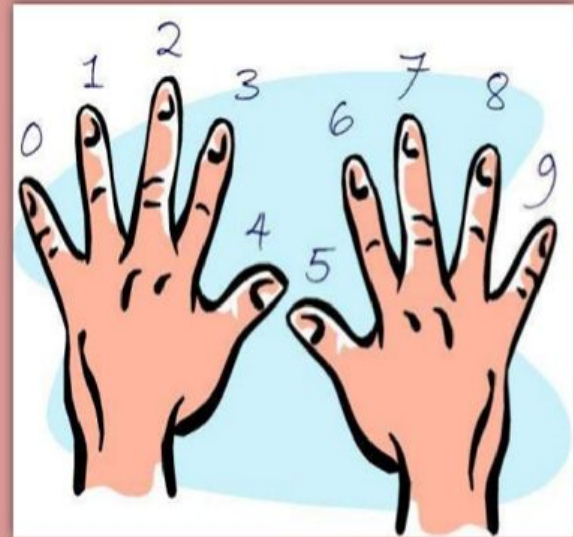
¿Con que numero sigo cuando llego al último?

10

¿A que valor representa el 10?

2

## Sistema de Numeración Decimal



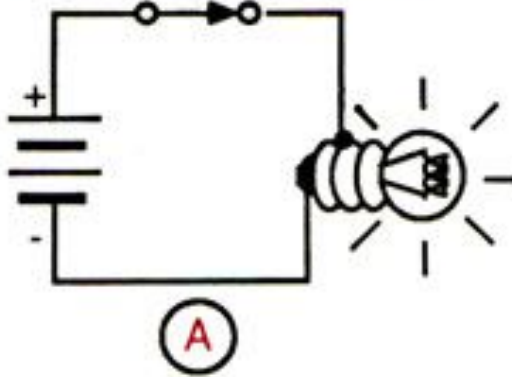
# Sistema Binario

Ultimo numero= 1

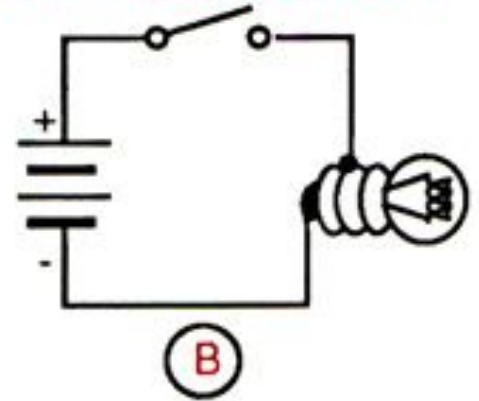
Siguiente = 10

BIT= **B**inary dig**IT**

Interruptor cerrado = 1



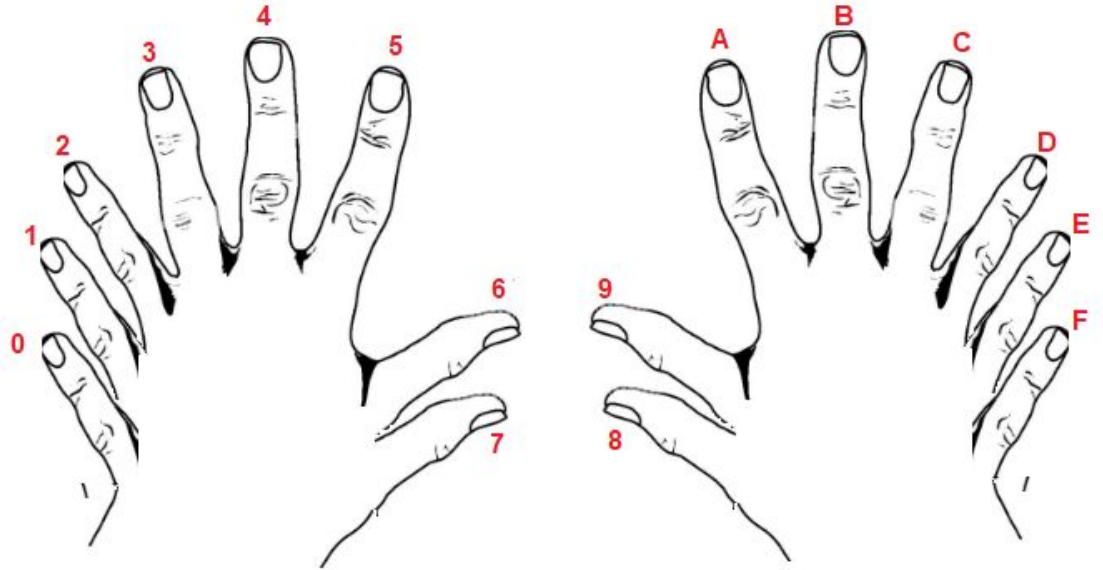
Interruptor abierto = 0



# Sistema Hexadecimal

¿Como seguimos contando más allá de F?

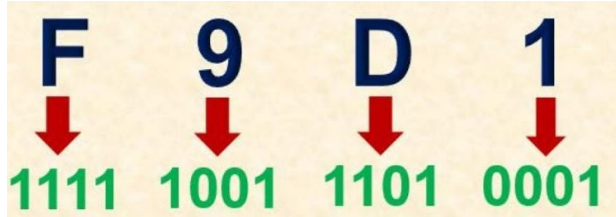
10



# Porque sistema octal y sistema hexadecimal

Es una forma de compactar la expresión binaria. Cualquier número exponencial de 2 (binario) serviría  $\Rightarrow$  2 (binario), 4(muy corto y no se usa), 8(sistema octal), 16(sistema hexadecimal 0-9+A-F), 32 (Deberíamos usar todo el abecedario y la compactación sería minima, solo 1bit más que el hexadecimal, por lo que no se usa)

Hexadecimal-Binario (nibble)



Octal Binario



Codificación ascii  $\Rightarrow$  C  $\Rightarrow$  0100 0001 = 0x41 = 65



# Matemática Binaria

Función	Tabla de verdad	Con interruptores	Con compuertas															
NOT $Z = \bar{A}$	<table><tr><th>A</th><th>Z</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	Z	0	1	1	0											
A	Z																	
0	1																	
1	0																	
OR $Z = A + B$	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Z	0	0	0	0	1	1	1	0	1	1	1	1		
A	B	Z																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
AND $Z = A \cdot B$	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Z	0	0	0	0	1	0	1	0	0	1	1	1		
A	B	Z																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
XOR $Z = A \oplus B$ $Z = \overline{A \cdot B}$	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Z	0	0	0	0	1	1	1	0	1	1	1	0		
A	B	Z																
0	0	0																
0	1	1																
1	0	1																
1	1	0																

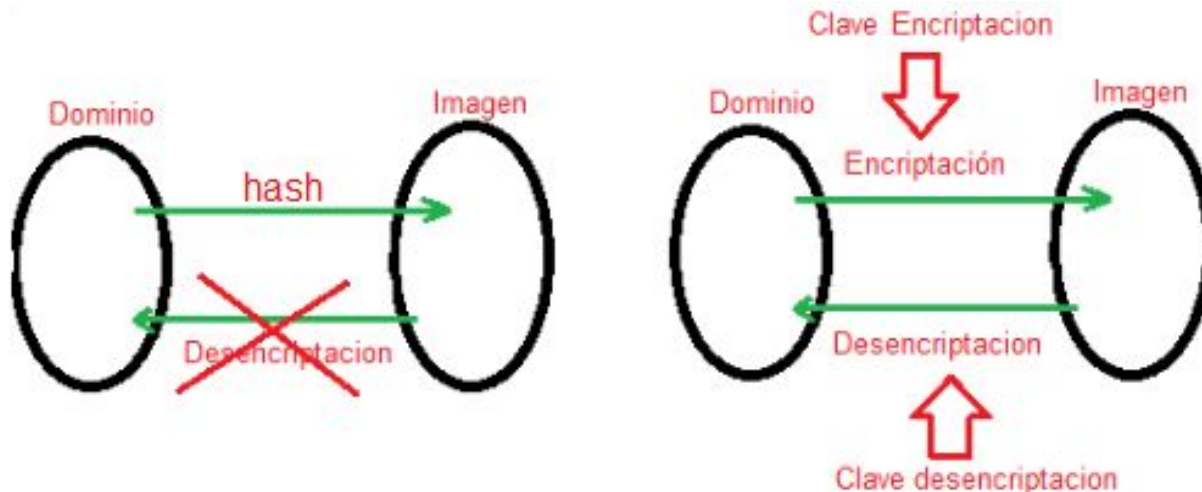
Función	Tabla de verdad	Con interruptores	Con compuertas															
<p>NOR</p> <p><math>Z = \overline{A + B}</math></p>	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Z	0	0	1	0	1	0	1	0	0	1	1	0		
A	B	Z																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
<p>NAND</p> <p><math>Z = \overline{A \cdot B}</math></p>	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Z	0	0	1	0	1	1	1	0	1	1	1	0		
A	B	Z																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
<p>XNOR</p> <p><math>Z = A \odot B</math></p> <p><math>Z = \overline{A \oplus B}</math></p>	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table> <p><math>Z = A \cdot B + \overline{A} \cdot \overline{B}</math></p>	A	B	Z	0	0	1	0	1	0	1	0	0	1	1	1		
A	B	Z																
0	0	1																
0	1	0																
1	0	0																
1	1	1																

# Quiz

Si les digo que estoy pensando un numero del 1 al 999 y que tienen solo una oportunidad para adivinarlo. ¿Que número es?

# Criptografía

## Introducción



simetrica => Clave Encriptacion= Clave desencriptacion

Asimetrica => Clave Encriptacion != Clave desencriptacion (ECDSA)

Hash => Salida fija (Normalmente) usado como firma (Keccak)

# Historia

MD4(1980)  $\Rightarrow$  MD5  $\Rightarrow$  SHA1  $\Rightarrow$  SHA2  $\Rightarrow$  SHA3 (2012)

SHA= secure hash algorithm

En la competencia de NIST del 2008 al 2012 sale elegida el Keccak como estandar SHA3. Al estandar actual SHA3 se le hicieron algunas modificaciones con el Keccak original planteado pero ethereum sigue utilizando el Keccak original (modificaciones en el b que usan). Esto es importante porque cuando ethereum hace alusión a SHA3 en códigos viejos no se refiere al estandar FIPS202 SHA3 de la NIST sino al Keccak original. (ERC59 corrige esto en el código)

Test vectors

Keccak256("") = c5d2460186f7233c927e7db2dcc703c0e500b653ca82273b7bfad8045d85a470

SHA3("") = a7ffc6f8bf1ed76651c14756a061d662f580ff4de43b49fa82d80a4b80f8434a

# Keccak

Funcionamiento del algoritmo

# Keccak

$$b=r+c$$

$$L=0,\dots,6$$

$$b=25.2^{**}L=1600$$

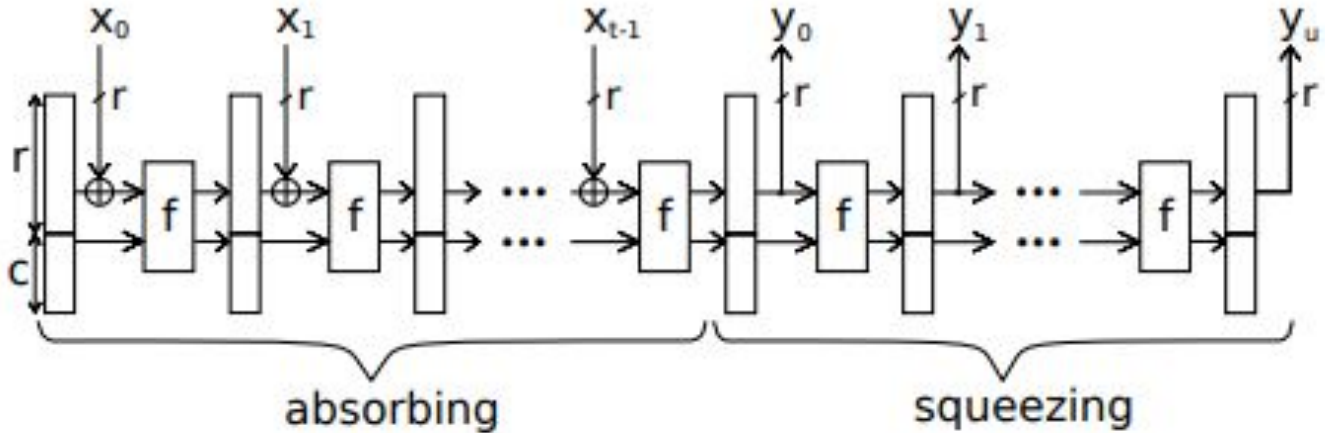
$$Nr=12+2.L=24$$

output:

224/256/384/512

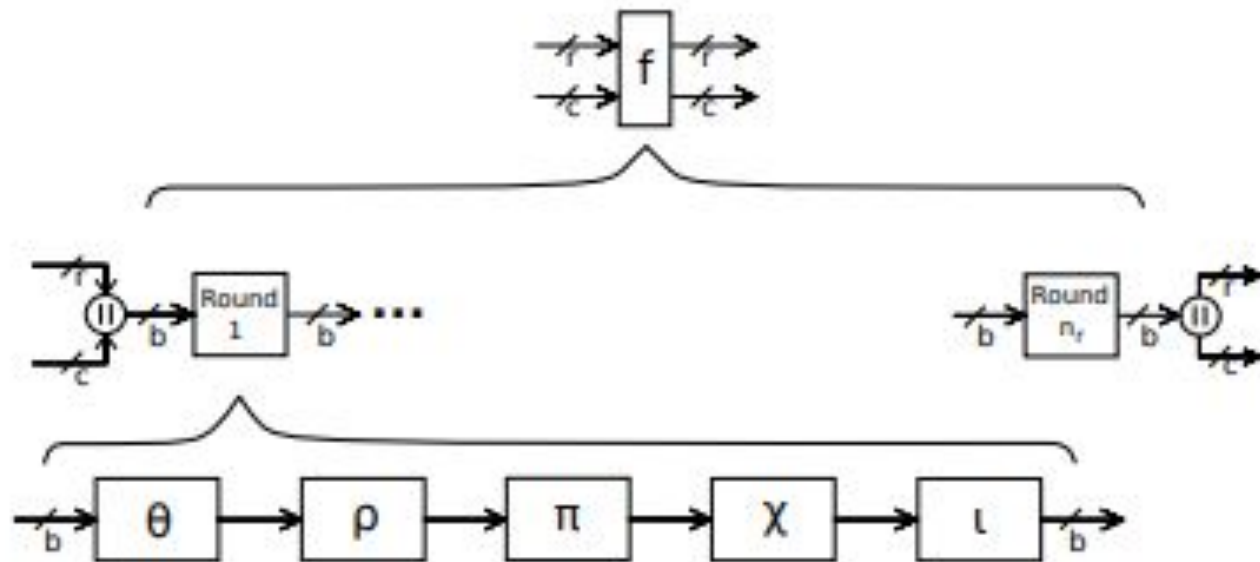
resistencia de tecnología= $2^{**}(n/2)$

r y c dependen de output y b. para sha3 y 256b  $\Rightarrow$  r=1088 y c=512



**Fig. 1.2** Absorbing and squeezing phases of the sponge construction

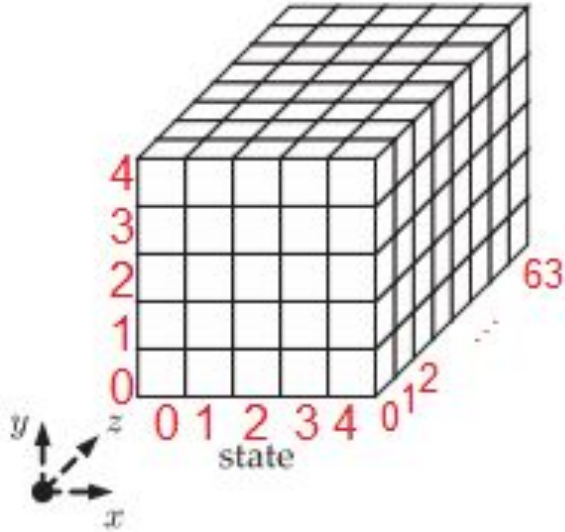
# Funcion F



**Fig. 1.3** Internal structure of function Keccak- $f$

# Preparando la matemática

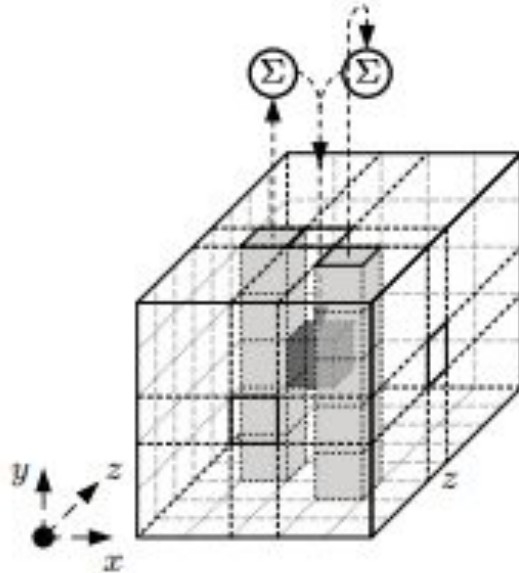
Representación de los bits B



Una ventaja de esto es que podemos usar 25 uint64  
(A veces)

# Funcion Tita

cada bit es una XOR de si mismo, de los 5bits en una posicion a la izquierda y de los 5 bits una posicion a la derecha y adelante. (XOR de 11 bits). Si desborda realiza un shift circular.



# Funcion RO

25 registros de 64 bits  $\Rightarrow$  Input=A[x,y]    Output=B[x,y]    con  $x,y=0,1,\dots,4$

Iteramos sobre los 25 registros

$\text{temp}[x,y] = \text{rot}(A[x,y], \text{rt}(x,y))$

$\text{rt}(x,y)$  es una tabla de doble entrada con valores constantes a rotar cada posicion.

	x = 3	x = 4	x = 0	x = 1	x = 2
y=2	25	39	3	10	43
y=1	55	20	36	44	6
y=0	28	27	0	1	62
y=4	56	14	18	2	61
y=3	21	8	41	45	15

# Funcion PI

Cambio el orden de esos 25 uint64 utilizando la función siguiente:

$$B[y, 2x+3y] = \text{temp}[x, y]$$

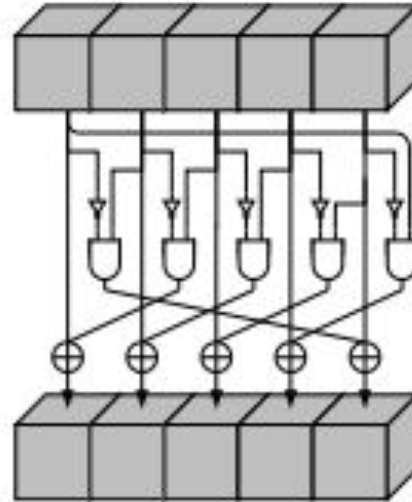
En el caso de que  $2x+3y$  se pase de 4, debo tomar el modulo 5 de ese valor.

Nota: Esta función pi suele ir conjunta con la función ro antes explicada.

# Funcion Chi

Recorro cada uno de los uint64 realizando la siguiente lógica:

$$B[x,y] = A[x,y] \text{ XOR } (!A[x+1,y] + A[x+2,y])$$



# Funcion iota

Esta funcion solo trata de una funcion por ronda en la que agrega una constante dependiente de la ronda al uint64 correspondiente a  $x=0$  e  $y=0$ .

Matemáticamente:

$$B[0,0] = A[0,0] + RC(i)$$

Donde RC es la función dada por la tabla,

$i$  es la variable para ingresar a la tabla y

$i$  es la ronda en la que nos encontramos

en nuestra funcion F.

RC[ 0] = 0x0000000000000001  
RC[ 1] = 0x0000000000008082  
RC[ 2] = 0x800000000000808A  
RC[ 3] = 0x8000000080008000  
RC[ 4] = 0x000000000000808B  
RC[ 5] = 0x0000000080000001  
RC[ 6] = 0x8000000080008081  
RC[ 7] = 0x8000000000008009  
RC[ 8] = 0x000000000000008A  
RC[ 9] = 0x0000000000000088  
RC[10] = 0x0000000080008009  
RC[11] = 0x000000008000000A

RC[12] = 0x000000008000808B  
RC[13] = 0x800000000000008B  
RC[14] = 0x8000000000008089  
RC[15] = 0x8000000000008003  
RC[16] = 0x8000000000008002  
RC[17] = 0x8000000000000080  
RC[18] = 0x000000000000800A  
RC[19] = 0x800000008000000A  
RC[20] = 0x8000000080008081  
RC[21] = 0x8000000000008080  
RC[22] = 0x0000000080000001  
RC[23] = 0x8000000080008008

# ECDSA

Curva Eliptica

# Encriptación asimétrica

Supongo clave privada= 3 y 33

Supongo clave publica= 7 y 33

Mensaje a encriptar=  $X = 9$

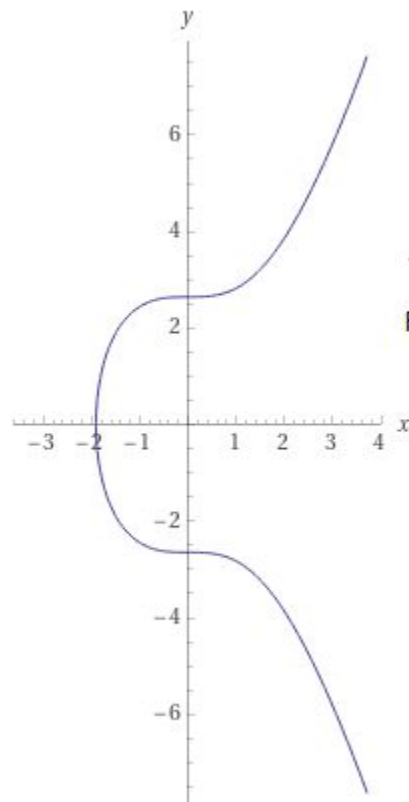
$$y = 9^3 \bmod 33 = 3 \text{ (3 es mi mensaje encriptado)}$$

Ahora para desencriptar no hacemos la inversa, Usamos la clave publica

$$X = 3^7 \bmod 33 = 9 \text{ (volvemos a obtener nuestro mensaje que era 9)}$$

Un procedimiento similar es ocurrirá con la curva elíptica, solo que partiremos de una clave privada y encontraremos una pública como la multiplicación de esta por un punto generador  $G$ . Las multiplicaciones encima no serán tradicionales sino elípticas.

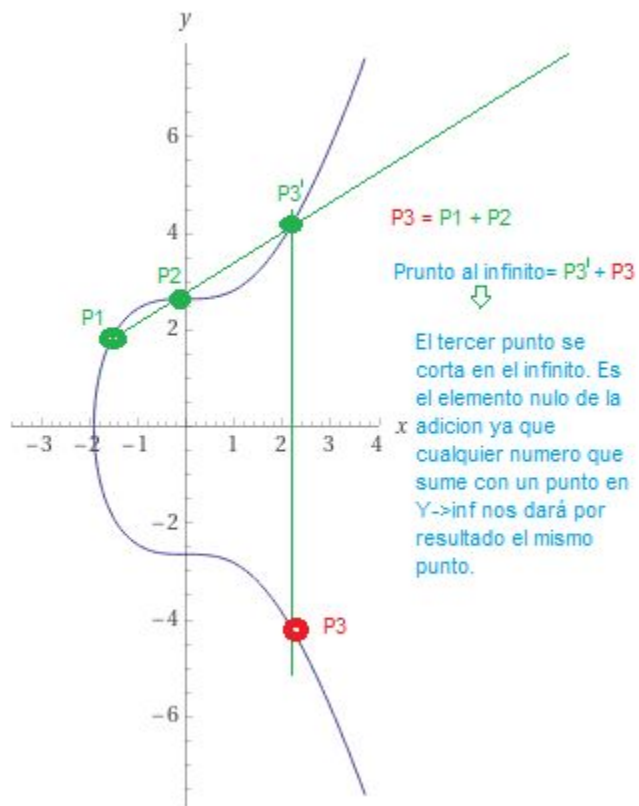
# Curva secp256k1



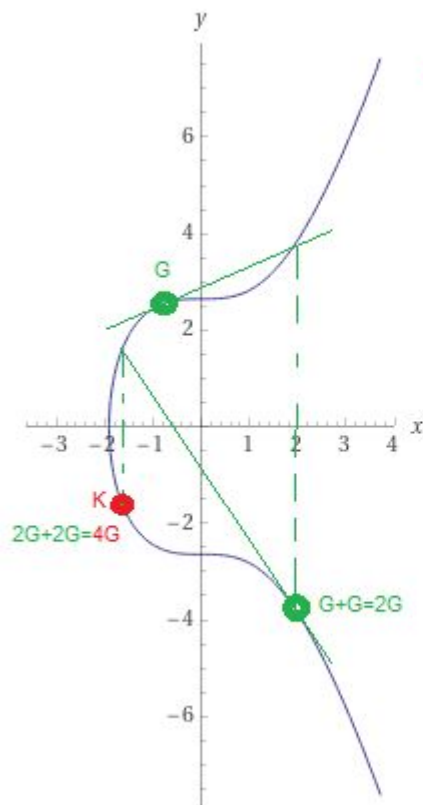
$$y^2 \% F_p = (x^3 + 7) \% F_p$$

$$F_p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$$

# Adición elíptica



# Multiplicación elíptica



$$K = 4 * G \Rightarrow$$

Siendo G un punto generador puedo multiplicar haciendo la suma de los sucesivos G.  $G+G$  como es el mismo punto, trata de una derivada para la pendiente. luego ese puntos resultante por si mismo será  $4G$ , luego  $8G$  y así sucesivamente. Para encontrar un numero de  $2^{**}256$  me llevaría como máximo 512 calculos de esta forma. En vez de  $2^{**}256$  que me llevaría si lo computo siempre contra G.

# Pregunta

Cuál es la diferencia entre un hash (keccak) y una encriptación asimétrica (ECDSA).

# Wallet y cuentas

¿que son y como se crean?

# ¿Que es una wallet?

Es un programa de software o hardware que contiene claves públicas y privadas que son únicas para el propietario del monedero y te permiten interactuar con la blockchain.

Esto te va a permitir realizar transacciones en la blockchain, así como recibir valor de una cierta criptomoneda/token y ver sus montos de una forma sencilla y amena.

Es importante resaltar que la wallet no contiene tokens, solo las claves privadas para mover lo que una determinada dirección publica posee.

# Categorías de wallets

Hot wallets: Están en línea, siempre conectadas a internet, son más cómodas de utilizar para los movimientos más cotidianos.

Cold wallets: El almacenamiento de estas está fuera de internet. En papel, hardware, una computadora, pero desconectada. No es cómodo para las transacciones diarias pero la seguridad es mucho mayor.

# Tipos de wallets

- Paper wallets: Está dentro de las cold wallets y los datos de la misma estan, como su nombre lo indica, en papel.
- Web wallets: Tu wallet está en internet, no está custodiada por tí sino que lo hace un tercero como un exchange.
- Desktop Wallet: Son wallets en tu ordenador. Tu eres custodio de tus datos por lo que debes cuidarlos bien. Igualmente estos siguen conectados a internet y no son tan seguros como los hardware wallets o paper wallets.
- Mobile wallets= Son wallets sobre todo para celular hechas como aplicaciones Dapps. (Electrum)
- Hardware wallets: Cold wallets parecidas a un pendrive que al realizar una transacción, estas mandan la firma y no la clave para impedir que un dispositivo online las registre. Son mucho más seguras pero más caras.

# Funcionamiento

Estas son un **conjunto de 3 numeros** que nos permitiran guardar valor en ella, así como realizar transferencia, haciendo uso de la criptografía estudiada.

Los 3 valores son:

- **Clave Privada** = Me permite crear transacciones
- **Clave Pública** = Me permite verificar la autenticidad de la transacción.
- **Dirección**= Me permite recibir transacciones.

Vamos a estudiar un poco mejor esto en las tecnologías basadas en la blockchain de ethereum. Para Bitcoin es conceptualmente igual pero los cálculos son más complicados.

# Clave privada

No es más que simplemente un **número aleatorio de 256 bits**. 256 bits corresponden a 32bytes o 64 números hexadecimales.

Es la única clave que como vimos en criptografía asimétrica **permite** crear la **encriptación** para crear una transacción desde mi billetera y **me identifica como** el único **dueño** de ella.

Si me la roban, o la pierdo, estoy frito.

Generador aleatorio= <https://www.browserling.com/tools/random-hex>

# Clave publica

Esto son 512 bits obtenidos de la **encriptación de la clave privada**, por lo tanto, sin importar que la gente conozca la clave pública, no podrán conocer la privada.

Suele agregar un 04 adelante marcando que la misma no está comprimida (04+x+y siendo el + una concatenación).

Esta clave pública, como se vio en criptografía asimétrica me **permitirá desencriptar el mensaje** de transacción que el dueño haya mandado usando su clave privada, **pero no** me dejará **encriptar** el mensaje haciéndome pasar por dicha persona.

Matemáticamente se calcula como:

Clave\_Publica= ECDSA (Clave\_Privada)

(Usar el algoritmo de python para mostrarlo)

# Direccion de la wallet

Muchas veces se la usa como sinónimo de la clave pública pero no lo son, ya que teniendo la clave pública se puede calcular la dirección de la wallet (La recíproca no es válida).

Está me sirve para yo pasarla a cualquiera y que la gente me mande y yo **recibir transacciones**.

Matemáticamente=

(314)

$$A(p_r) = \mathcal{B}_{96..255}(\text{KEC}(\text{ECDSAPUBKEY}(p_r)))$$

Para el Keccak debo omitir el 04 que me indica que x e y de ecdsa no fueron comprimidos.

([https://emn178.github.io/online-tools/keccak\\_256.html](https://emn178.github.io/online-tools/keccak_256.html))

# Ataques a claves privadas

Al no poder desencriptar hay 3 ataques que podríamos usar para averiguar una clave privada:

- Ataque de **fuerza bruta**= Consiste en probar toda la combinatoria y encriptarla hasta que los hashes coincidan, de esta forma sabremos cual clave privada corresponde a la clave pública.
- Ataque de **Rainbow tables**= Son bases de datos que ya contienen billones de pares mensaje/encriptación. Son las más usadas para romper passwords comunes como 1234, admin, probando, etc.
- Robarlas usando **Ingeniería social**= páginas de phishing, dust tokens o cualquier tecnica que al hacker se le ocurra para explotar al eslabón débil de la cadena “el usuario”.

# Por que Fuerza bruta y Rainbow son inefectivas

Combinatoria=  $2^{256}$  valores =  $10^{77}$

Cantidad de granos de arena en el mundo=  $0,56 \cdot 10^{23}$

Ordenador más potente del mundo= Fugaku => 442 petaflops/s

Fugaku realiza =  $442 \cdot 10^{15}$  operaciones por segundo.

Suponiendo que calcular la encriptación es una operación y que cada grano de arena en el mundo es una fugaku. (sabiendo que la colision es  $2^{(256/2)}$  )

Años en descubrir una clave=  $0,128 \cdot 10^{15}$  años (**128 billones de años**)

Suponiendo computadoras cuanticas que calculen en 200s lo que esta computadora calcula en 10mil años. aun estamos hablando de **81 mil años**.

# Mnemónico

Se podrán estar preguntando que es un mnemónico entonces ya que no lo mencioné antes. Esto no es más que una **codificación de la clave privada**. Existe un algoritmo conocido al cual le ingresamos el mnemónico (12 palabras) y nos da por resultado la clave privada. La ventaja es que si te equivocaste en alguna letra al copiar podrás darte cuenta por ortografía en cambio si copiamos mal uno de los 256 bits de la clave privada será imposible saber el problema.

Pueden verlo en el siguiente enlace: <https://iancoleman.io/bip39/>

Estandar= <https://github.com/bitcoin/bips/blob/master/bip-0039.mediawiki>

Si hay tiempo mostrar cómo funciona en un exchange centralizado.

# Estandar EIP55

Las wallets de ethereum no incluyen como bitcoin un checksum para proteger al usuario de tipear mal un address. Originalmente tampoco distinguen entre minúsculas y mayúsculas, pero habrán notado que a veces tienen letras en mayúscula y otras en minúscula. Esto se debe a un checksum dado por el estandar EIP55. Cualquier billetera que lo integre, al chequear las letras en minúsculas y mayúsculas podrá ver si se equivocaron o no al ingresar una wallet.

¿Cómo funciona?

Dirección= 0x001d3F1ef827552Ae1114027BD3ECF1f086bA0F9

kd=keccak(001d3f1ef827552ae1114027bd3ecf1f086ba0f9)

kd=23a69c1653e4ebbb619b0b2cb8a9bad49892a8b9695d9a19d8f673ca991deae1

Alineamos dirección y kd. Analizamos línea por línea. si kd es mayor o igual a 8 y en la línea la dirección corresponde a una letra irá en mayúscula, sino en minúscula.

# Tipos de wallets según metodo de generación

- Aleatoria: Cada cuenta que se genera tiene sus llaves privadas totalmente aleatorias. Tambien se las conoce como JBOK wallet (just a bunch of keys). Practicamente no se las usa.
- Deterministicas: Cada cuenta generada es derivada de una clave maestra conocida como semilla. La mas conocidas son los estandares BIP32 y BIP44 que usan una derivación de claves en formato de arbol.

# Paper wallet para BTC

En el siguiente enlace se puede generar una y es de código abierto

<https://www.bitaddress.org/>

El código puede plagiarse de:

<https://github.com/pointbiz/bitaddress.org>

Otro enlace para generar paper wallets basado en el anterior:

<https://paperwallet.bitcoin.com/>

La entropía la genera con las coordenadas del mouse.

# Paper wallets para varias criptomonedas y tokens

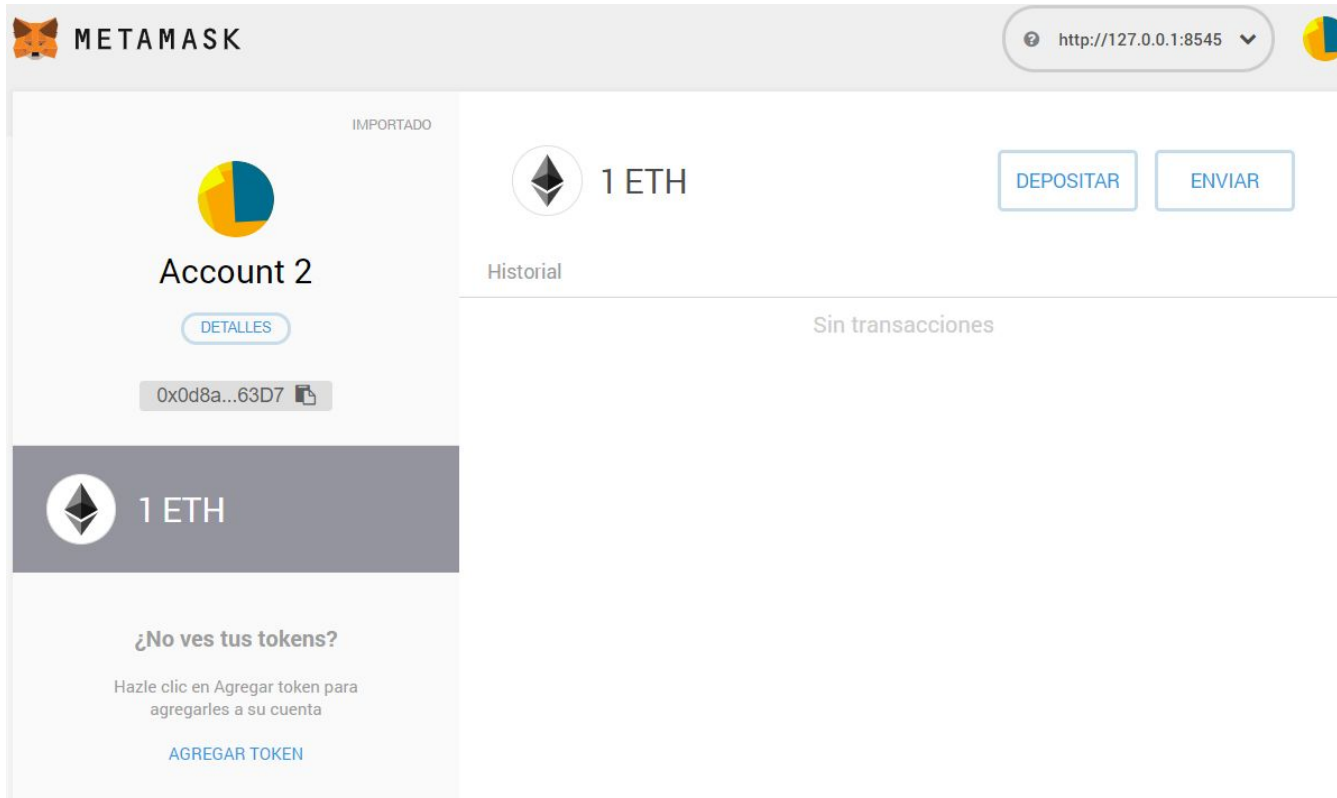
<https://github.com/hubernautmartin/generator/>

# Cuidado de las paper wallets

- En criptomonedas que usen UTXO como BTC Tener cuidado con los change address. Para solucionar esto debemos establecer cual será nuestra change wallet desde el principio, o lo que es más fácil mandar todos los fondos a otra billetera (trustwallet por ejemplo) y de ahí enviar los btc parciales. Ethereum o similares no tienen problema porque sus balances se basan en estados.
- Al conectarlas perdemos la ventaja de cold wallet. Por esto se recomienda varios ingresos, pero una sola extracción. Una alternativa es firmar offline y mandar solo la firma online.
- Según binance su auge cayó en el 2016 por la dificultad de uso para el UTXO mencionado. Pero en latinoamérica donde es difícil conseguir hardware wallets y que estas vengan íntegras, sigue siendo una muy buena opción y más barata.
- El papel es frágil y puede borrarse o romperse pero las mismas también pueden guardarse en un formato digital en un pendrive.

# Pregunta conceptual

¿Cuántos Eth  
tiene mi  
Wallet?



# Bibliografía

<https://ethereum.github.io/yellowpaper/paper.pdf>

mastering ethereum (Andreas antonopulus)

Binance Academy

Understanding Cryptography (Christof Paar)

SHA3 and the hash function keccak (Christoph Paar)

<https://impulsomatematico.com/2019/10/16/numeros-primos-como-enciptar-y-des-enciptar-mensajes-con-ellos-y-algunas-otras-curiosidades/>

# Consultas

¿?

# Reto

<https://curso-blockchain.tk/tarea2/>