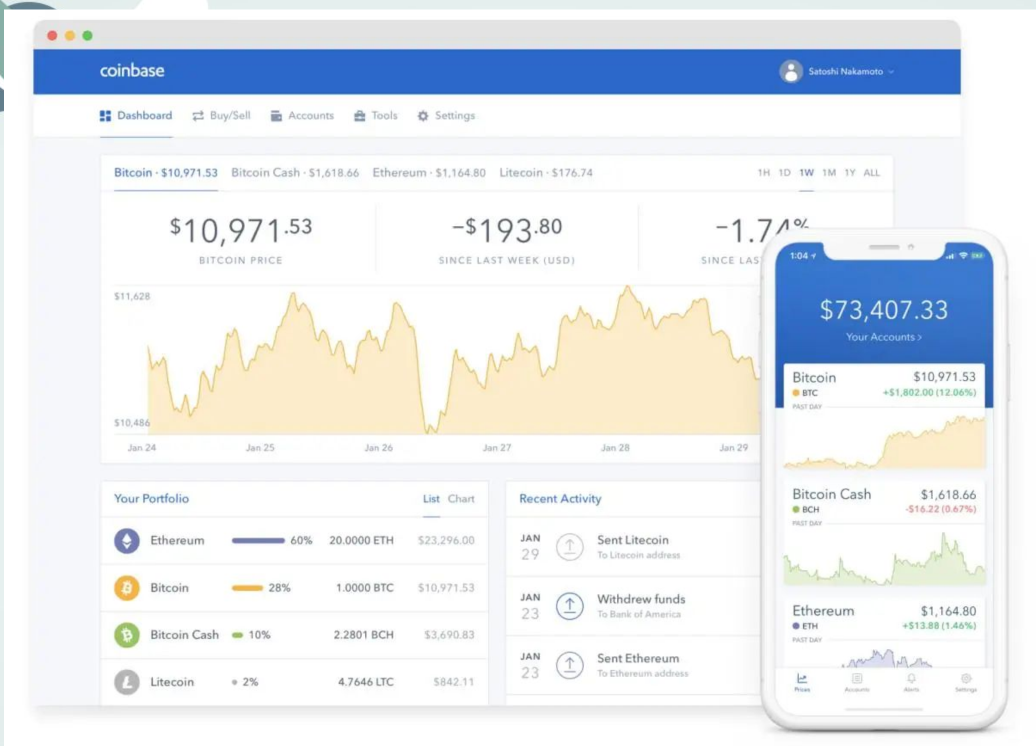
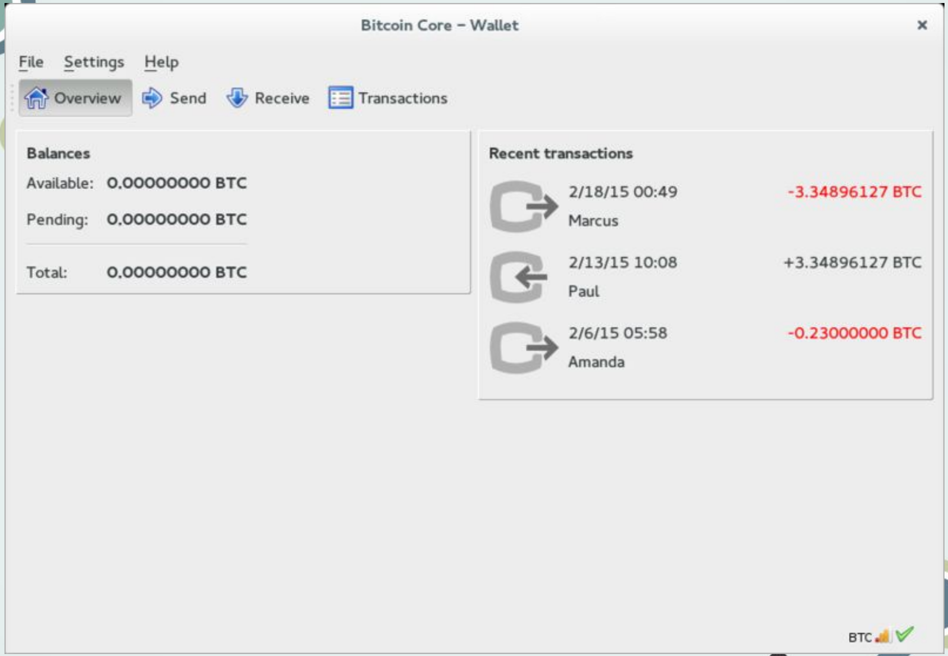
Elliptic Curve Cryptography and Its Application in Bitcoin Wallet

Jin Lou

https://www.nbcnews.com/news/latino/el-salvador-hits-snags-adopts-bitcoin-official-currency-first-country-rcna1910



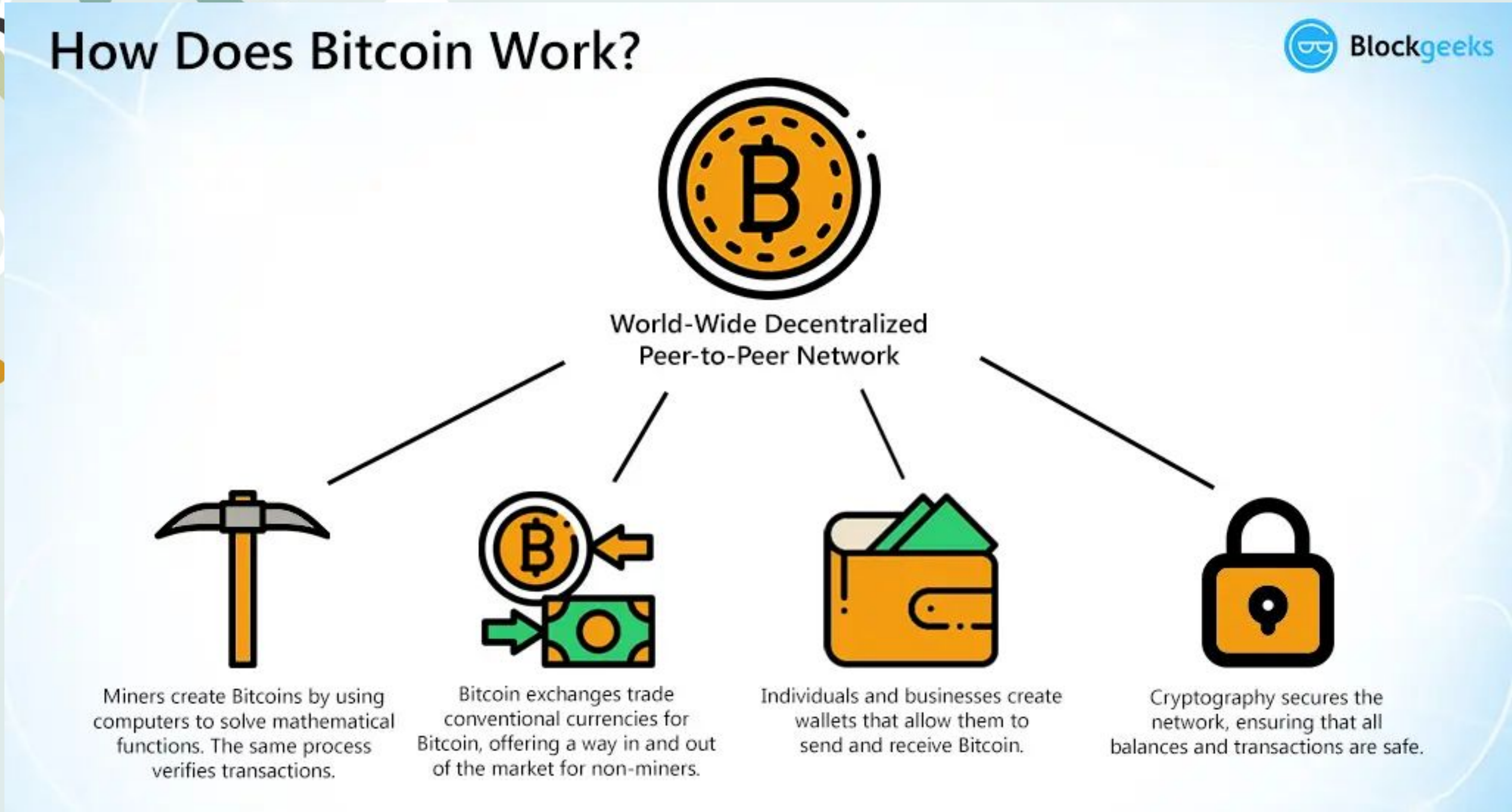
https://www.computerworld.com/article/3389678/whats-a-crypto-wallet-and-does-it-manage-digital-currency.html



https://commons.wikimedia.org/wiki/File:Bitcoin-core-v0.10.0.png

What is Bitcoin?

It is a decentralized digital currency that is independent of banks and can be sent from user to user on the peer-to-peer bitcoin blockchain network without the need for intermediaries.


How Does Bitcoin Work?
Blockgeeks
World-Wide Decentralized Peer-to-Peer Network
Miners create Bitcoins by using computers to solve mathematical functions. The same process verifies transactions.
Bitcoin exchanges trade conventional currencies for Bitcoin, offering a way in and out of the market for non-miners.
Individuals and businesses create wallets that allow them to send and receive Bitcoin.
Cryptography secures the network, ensuring that all balances and transactions are safe.

The Basics

Private key

It is the password to your bitcoin account. You do not want to share it with anyone!

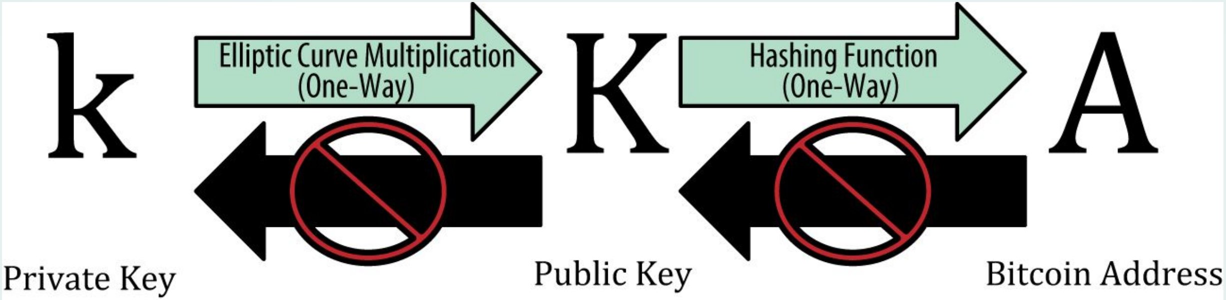
Public key

It is used to prove that the digital signature came from the private key. The signature proves ownership of the private key.

Address

The bitcoin address is visible to everyone on the blockchain. Just like your bank account number, the sender needs that to transfer money to you.

The Basics



Trapdoor functions

https://www.oreilly.com/library/view/mastering-bitcoin-2nd/9781491954379/ch04.html

JUMP IN DEPTH

How do we obtain those three things?

How do we create a private key (k)?

Hex: DD5113FEDED638E5500E65779613BDD3BDDBEB8EB5D86CDD3370E629B02E92CD

Base64: 3VET/t7WOOVQDmV3lhO9073b64612GzdM3DmKbAuks0=

WIF: 5KVkpWGfDQGJAUEEDUFbrFxwNPjmXy5kBBmRzzBDf4JkgFXqXTa

Binary:
1101110101010001000100111111111011011110110101100011100011100101010100
0000001110011001010111011110010110000100111011110111010011101111011101
1011111010111000111010110101110110000110110011011101001100110111000011
1001100010100110110000001011101001001011001101

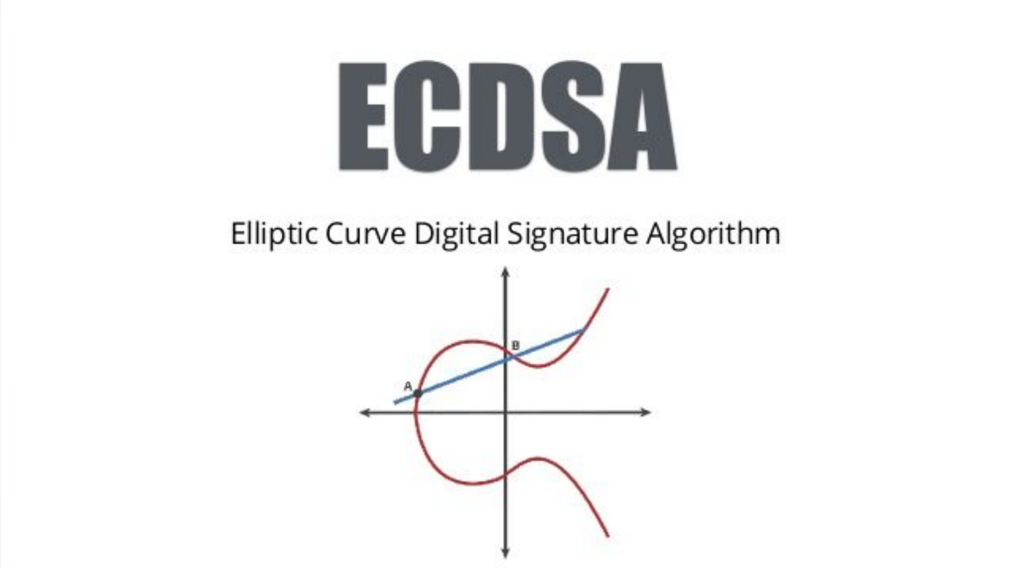
https://www.freecodecamp.org/news/how-to-generate-your-very-own-bitcoin-private-key-7ad0f4936e6c/

Bitcoin uses the SHA-256 hash algorithm to generate a secret number k that is 256 bits long.

Would there be duplicate private keys?

We don’t need to concern about that right now because 2^{256} or 10^{77} is a very big integer.

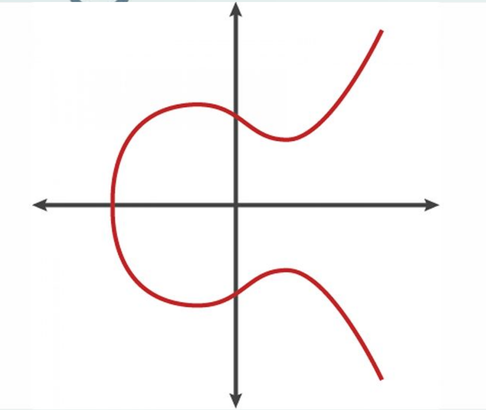
How do we create a public key(p)?

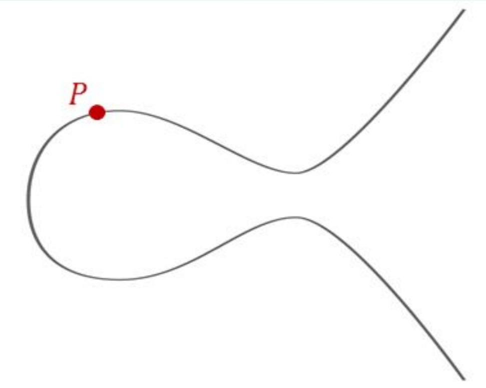


https://metamug.com/article/security/sign-verify-digital-signature-ecdsa-java.html

ECC algorithm is the base of ECDSA

(Elliptic Curve Cryptography)





The function: $y^2 = x^3+ax+b$ (a=0,b=7 in bitcoin blockchain)

Let's say:

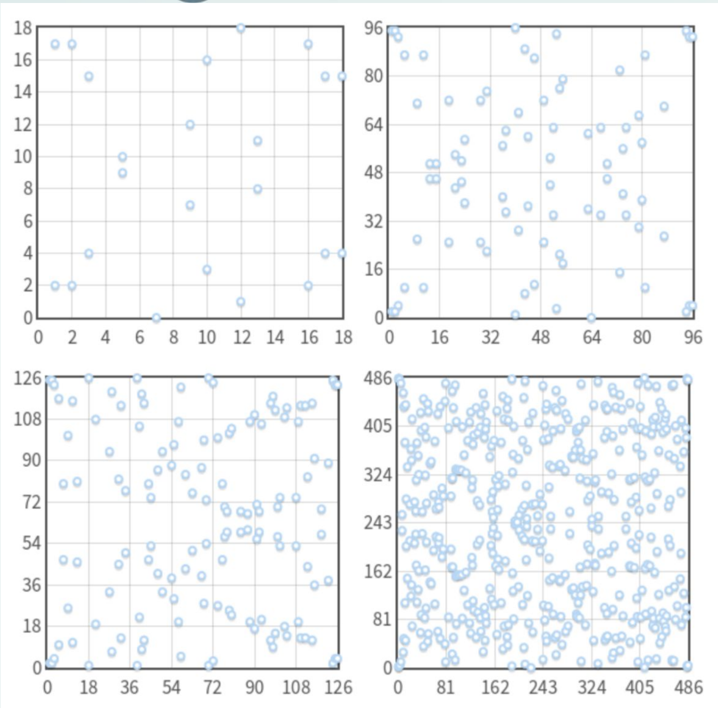
- k is private key
- p is public key
- G(x,y) is a point on the curve, we call it the "generator"
- '*' is a group operation such that k * G = (kx, ky)= p (In other words, p is a point)

k * G = G+G+....+G

Point and point addition:
A + B -> C

What if it is A + A?
A + A -> use tangent line

The curve is actually a finite field!



https://medium.com/@_ericprice/bitcoins-signing-algorithm-elliptic-curves-with-finite-fields-e386f8d0c05

How do we pick a G?

- Any point in the block can be used as a generator.
- Everyone uses the same G in bitcoin:

$p=2^{256}-2^{32}-2^{9}-2^{8}-2^{7}-2^{6}-2^{4}-2^{0}$

=0279be667ef9dcbbac55a06295ce870b07029bfcdb2dce28d959f2815b16f81798

How many G are there in the block?

- Group size - 1 since it is a cyclic group.

Why using a field?

- To avoid numerical precision issues, we use finite fields because the number of elements is finite and known exactly.

An example: $y^2 = x^3 + 2x + 3 \mod 97$
With G=(3,6)

0*G = (infinity)
1*G = (3,6)
2*G = (80,10)
3*G = (80,87)
4*G = (3,91)
5*G = (infinity)
6*G = (3,6)
7*G = ...

The order of this cyclic group is 5, much less than 97.

The points have been partitioned into separate cyclic groups, all of the same size.

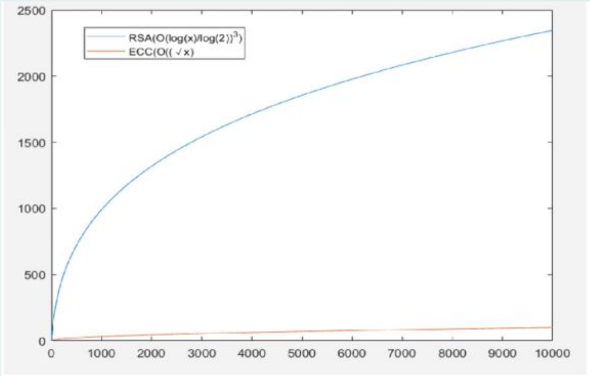
# of such groups is call "cofactor", h=n/r

How fast can we compute the public key p

Do we have to do calculations k times?

- No! The time complexity of ECC algo is $O(\log_2 n)$

  why?
  G+G=2G, 2G+2G=4G, 4G+4G=8G
  ...
  If n = 16, we need to do sqrt(16)=4 calculations.

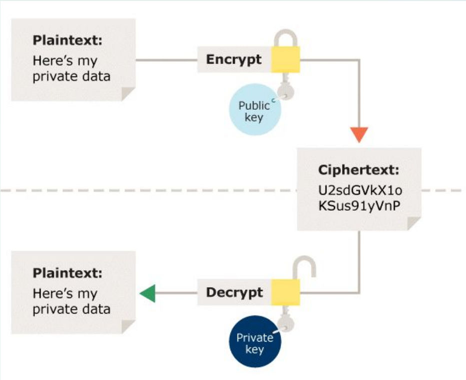


https://www.researchgate.net/figure/The-time-complexity-of-RSA-and-ECC_fig4_330832141

We love ECC because it is much faster than RSA with the same key size.

(1) Data Encryption: If we know p, we can hardly compute k.

Alice: k_a, p_a -> k_a*p_b = G * k_a * k_b
Bob: k_b, p_b -> k_b*p_a = G * k_a * k_b

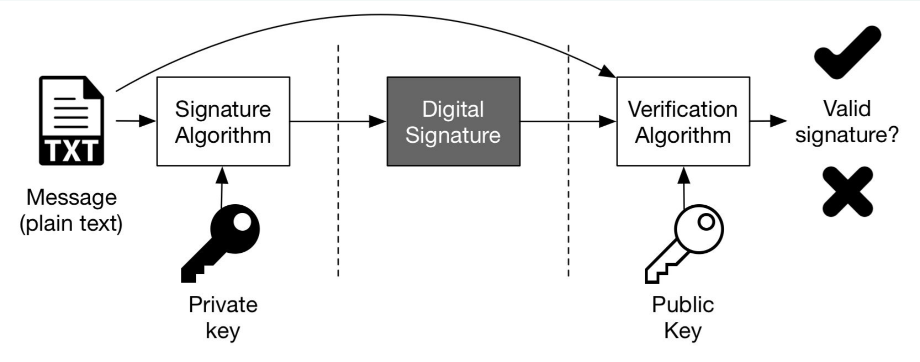


https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/encryption/what-types-of-encryption-are-there/

(2) Digital Signature: If we are given k, we can compute whether k*G matches p.

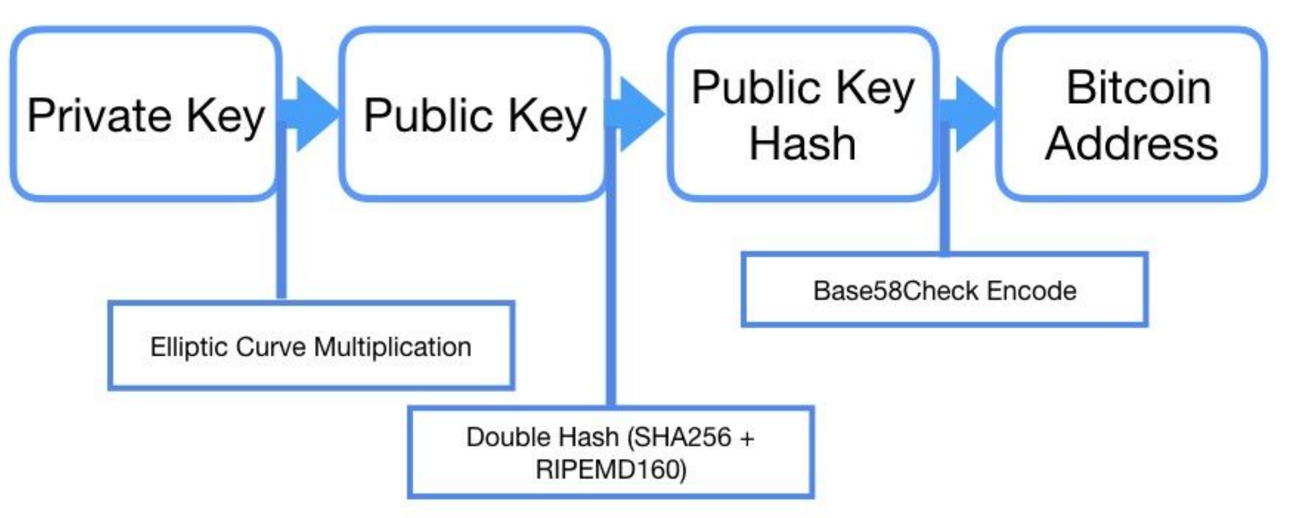
Suppose m is message
m*k = N(digital signature)
m*p = m*G*k = (m*k)*G=N*G

We can check whether signature matches to verify the integrity of the message m.



https://stakey.club/en/verifying-digital-signatures/

Now we know p and k, what about the address?



https://medium.com/coinmonks/what-is-a-bitcoin-address-6c822c857004

We can compute the Bitcoin address from Double Hash functions(SHA256 + RIPEMD160)

Issues With ECC Implementation

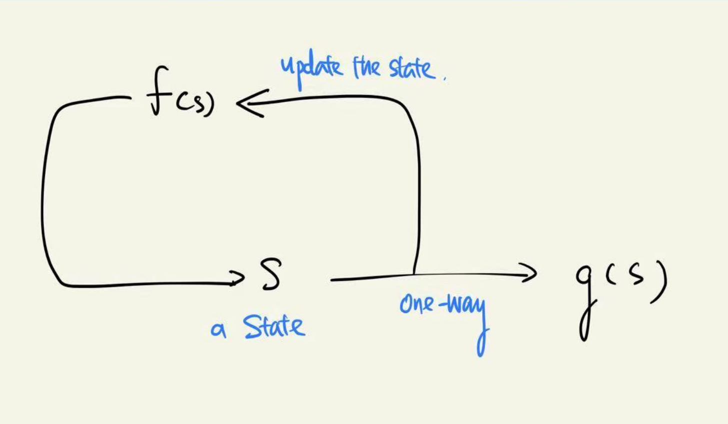
What if k is a static value? Disaster!

A secure implementation of the ECC curve is theoretically possible, it is not easy to achieve. There are numerous examples of how failed implementation of ECC algorithms resulted in significant vulnerabilities in the cryptographic software.

A great example is that of the Sony ECDSA security disaster. Although Sony used ECDSA to sign software for their PlayStation game console, they did not properly implement the algorithm. Using static parameters instead of random ones made Sony's implementation of the algorithm solvable and subsequently useless.

Can we obtain truly randomness?

AKA can we build a random number generator
that guarantees to create a random number?



Typical workflow of a random number generator

Thank you!