

01000101001001010  
101010101001010100111101110  
0101100101000101001010100010111110  
101010100010100100101010101010100  
01010100010100100101010101010100  
010011110100010010010101001010100  
100010110101110101111011100  
101100101010101010101111101  
101010001010101010101111110  
10101010001010101010100100100  
101010101010101010101001111011100101  
10010100010100101010001011111101101  
0100010100100101010101010101



# 比特幣介紹

# 比特幣

- 2008 年 10 月
  - Satoshi Nakamoto (中本聰) 提出了比特幣設計白皮書
  - Bitcoin: A Peer-to-Peer Electronic Cash System
- 2009 年
  - 開源
  - 比特幣軟體
  - 以 10,000 比特幣兌換一個披薩
- 2010 年開始流行
  - <https://bitcoin.org/>



# 電子貨幣

- 如何鑄幣
- 如何確保錢幣無法被偽造
- 如何確保錢幣無法二次消費
- 如何運營且不需要第三方管理
  - PayWord<sup>1</sup>
  - MicroMint<sup>1</sup>
  - HashCash<sup>2</sup>

<sup>1</sup> <https://people.csail.mit.edu/rivest/RivestShamir-mpay.pdf>

<sup>2</sup> <http://www.hashcash.org/papers/hashcash.pdf>

# 系統設計挑戰

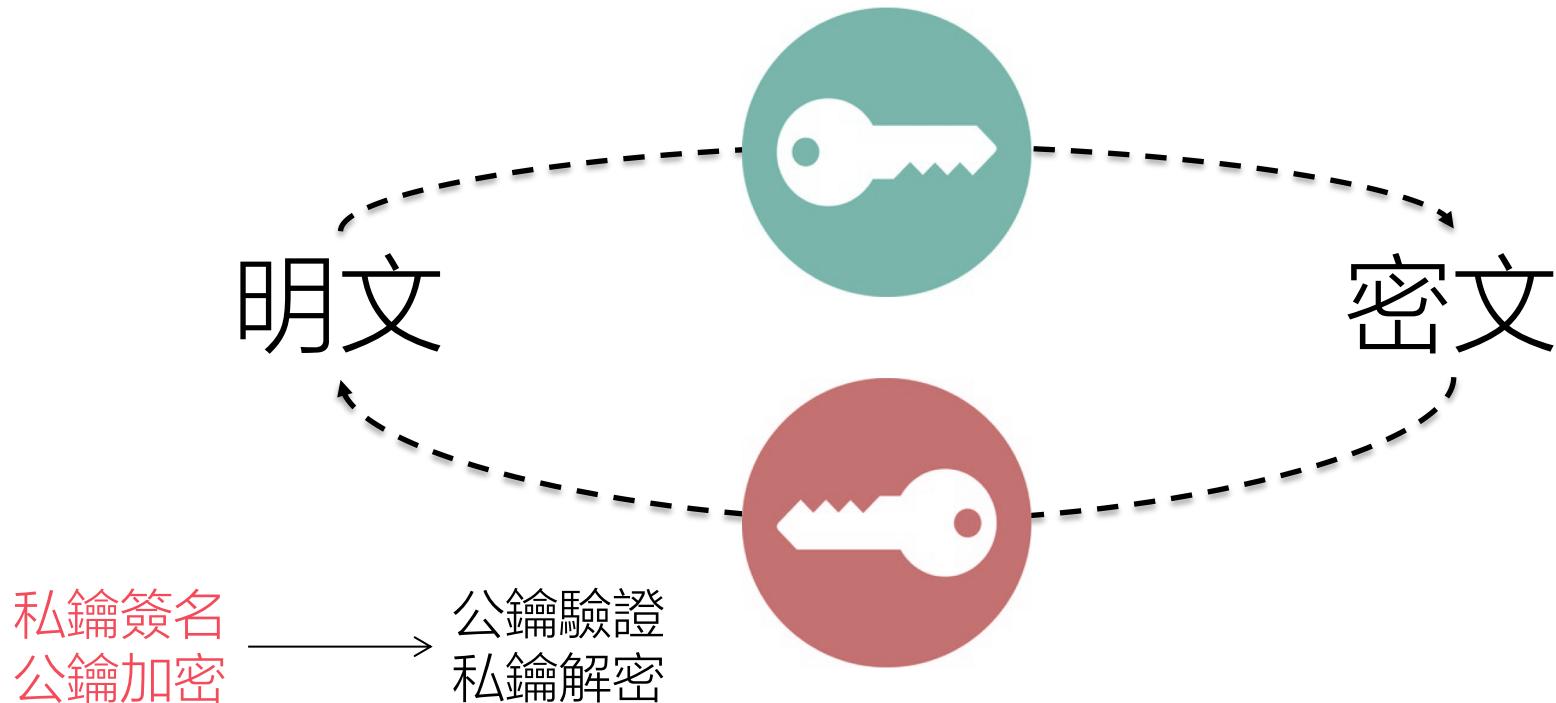
- 如果要透過電子貨幣進行交易
  - 如何避免電子貨幣被複製？
  - 如何避免電子貨幣被偽造？



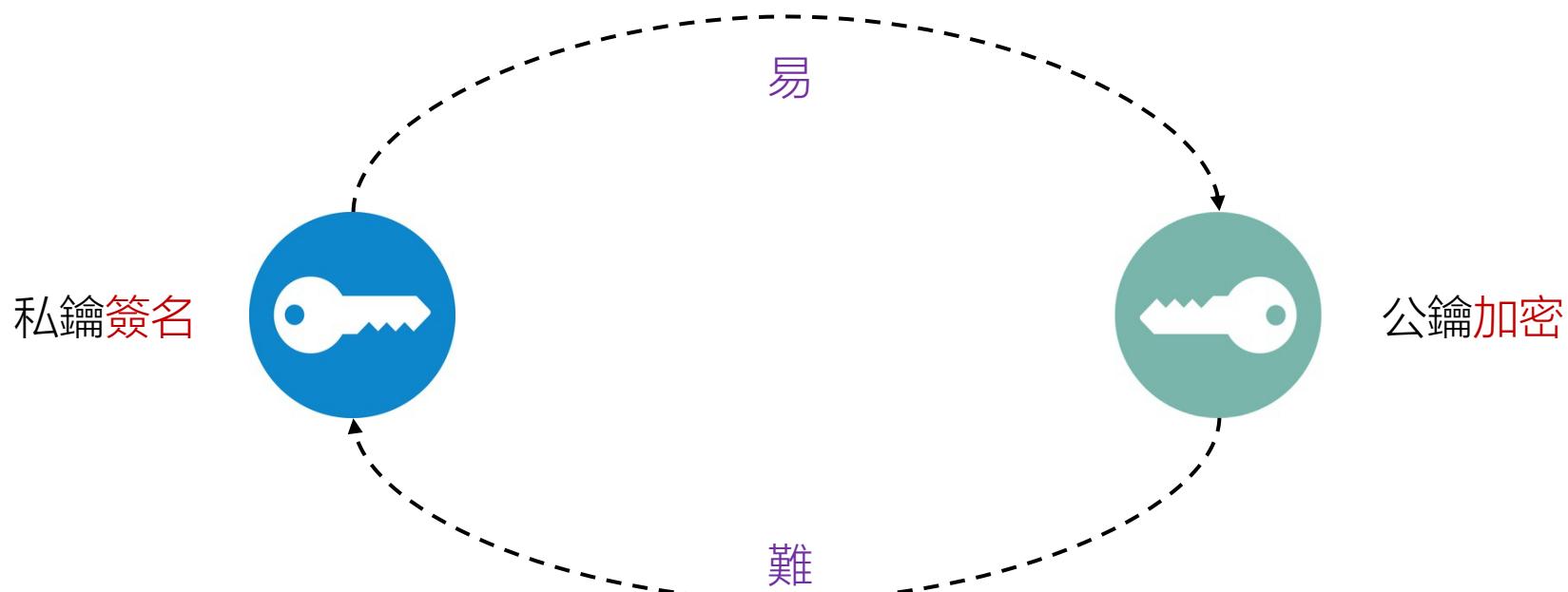
# 加密演算法 – 對稱密鑰



# 加密演算法 – 非對稱公私鑰



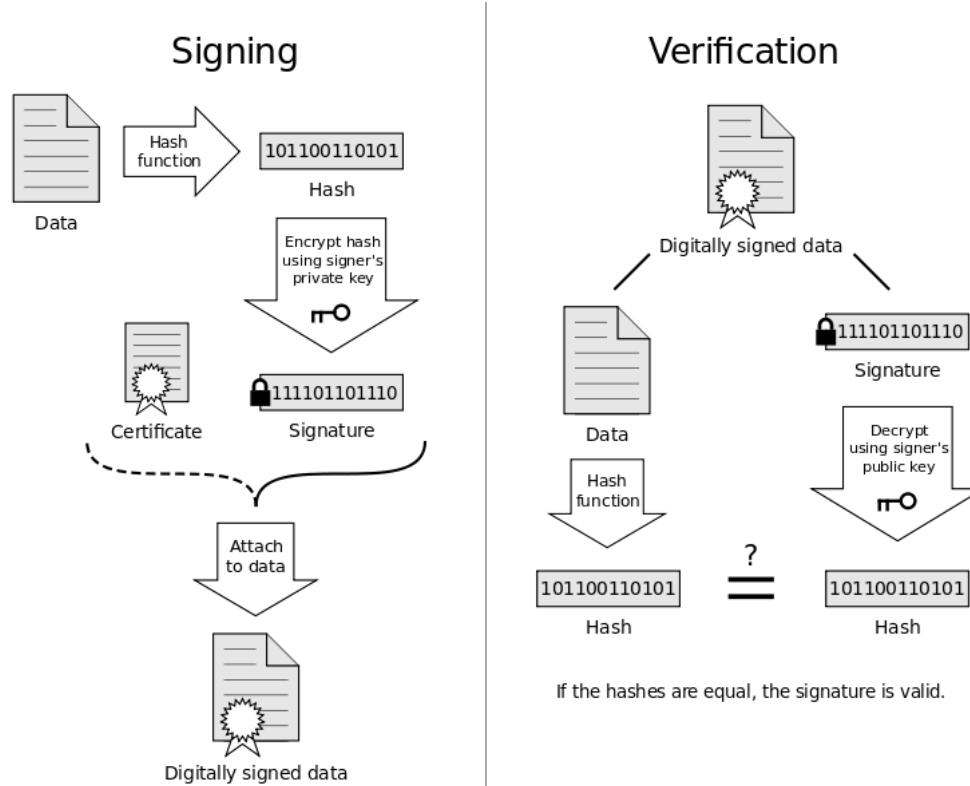
# Public-Key Cryptography



# Public-Key Cryptography

1. RSA
2. Elliptic Curve Cryptography (ECC)
  - ECDH
  - ECIES
  - **ECDSA**
    - The Elliptic Curve Digital Signature Algorithm (ECDSA) is based on the Digital Signature Algorithm
  - EdDSA
  - ECMQV
  - ECQV

# Digital signature



# ECDSA

(Elliptic Curve Digital Signature Algorithm)

- 可辨識的簽名，無法被偽造
- 保護資料不被竄改，但資料**沒有加密**
- 數位簽章不像 AES

好處: Key 長度短 , 可省空間 (disk space) 與頻寬 (bandwidth)  
且安全性與 RSA 相當

# Why ECC?

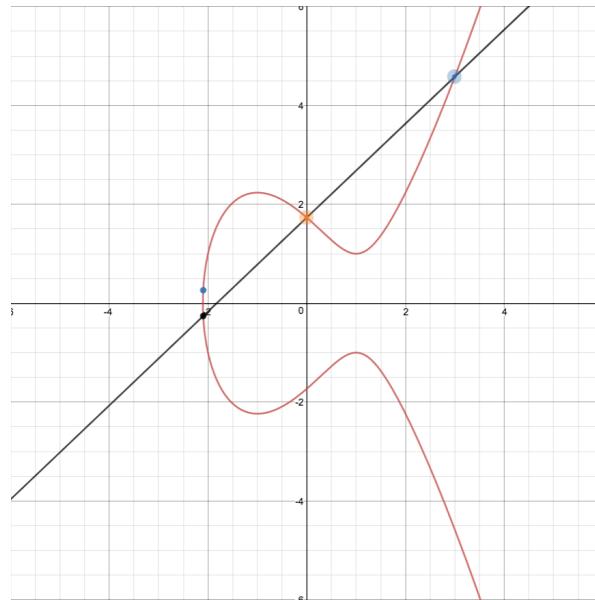
- A 256 bit key in ECC offers about the same security as 3072 bit key using RSA.

Symmetric Key Size (bits)	RSA and Diffie-Hellman Key Size (bits)	Elliptic Curve Key Size (bits)
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512
NIST recommend key sizes		

1. <https://csrc.nist.gov/publications/detail/sp/800-131a/rev-1/final>
2. <https://www.keylength.com/en/compare/>

# ECDSA (簽名)

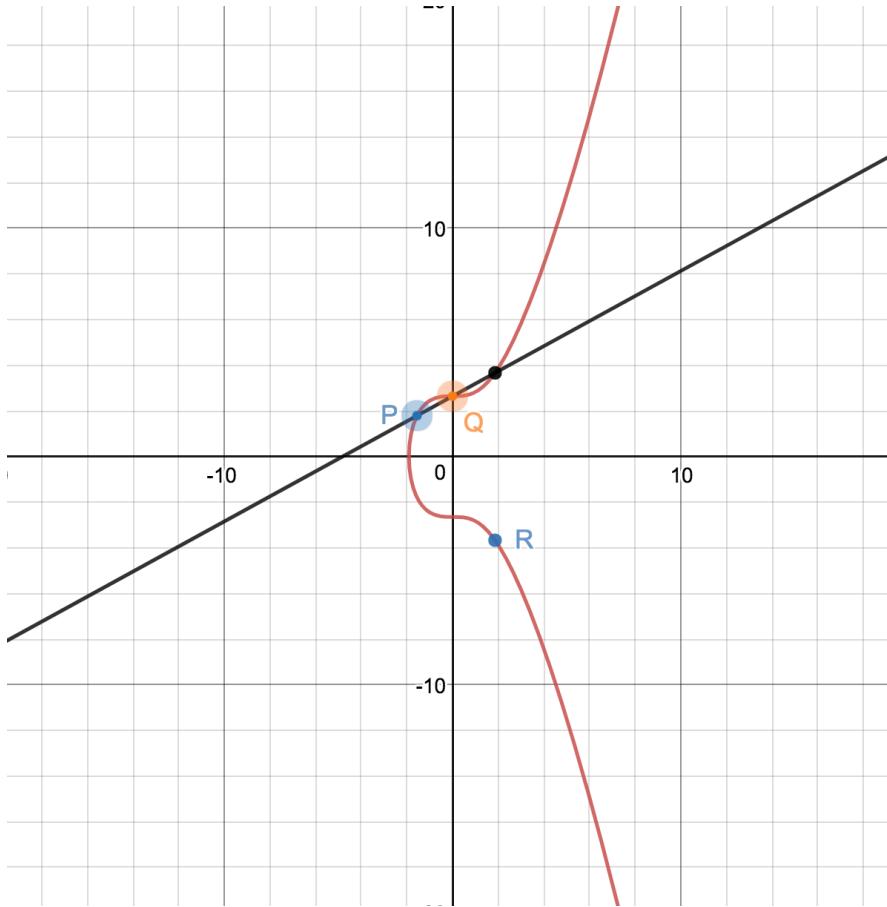
- $d_A$ : Private key (隨機數)
- $Q_A$ : Public key (曲線上的點)
- $Q_A = d_A \times G$  ( $G$  是曲線上的某個點)
- 假設要對檔案進行簽名，會用到
  1. 計算檔案的 Hash 演算法 (ex: SHA<sup>2</sup>56, SHA3)
  2. 用私鑰簽名得到的簽章
  3. 數位簽章包含兩個部分:  $r$  與  $s$



$$y^2 = x^3 - 3x + 3$$

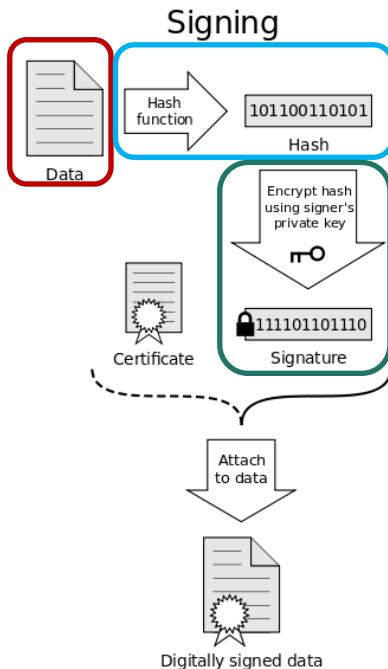
# secp256k1

$$y^2 = x^3 + 7$$



# ECDSA on Ethereum

EIP#155



rlp + hash

0xdaf5a779ae972f972197303d7b574746c7ef83eadac0f2791ad23db92e4c8e53

sign with privateKey  
and modify v

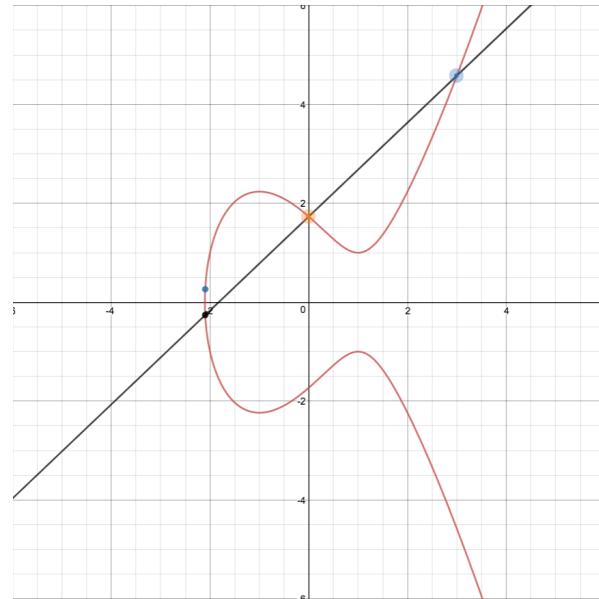
```
v: '0x25'  
r: '0x28ef61340bd939bc2195fe537567866003e1a15d3c71ff63e1590620aa636276'  
s: '0x67cbe9d8997f761aecb703304b3800ccf555c9f3dc64214b297fb1966a3b6d83'  
                                signature
```

signature

(r, s): each 32 bytes

# ECDSA (驗證)

- $d_A$ : Private key (隨機數)
- $Q_A$ : Public key (曲線上的點)
- $Q_A = d_A \times G$  ( $G$  是曲線上的某個點)
- 假設要對數位簽章進行驗證
  1. 從簽章中取出  $s$
  2. 代回橢圓曲線計算，得到  $r$
  3. 若  $r$  有效，則簽章合法



$$y^2 = x^3 - 3x + 3$$

ps. 目前沒有方法能從公鑰快速推得私鑰

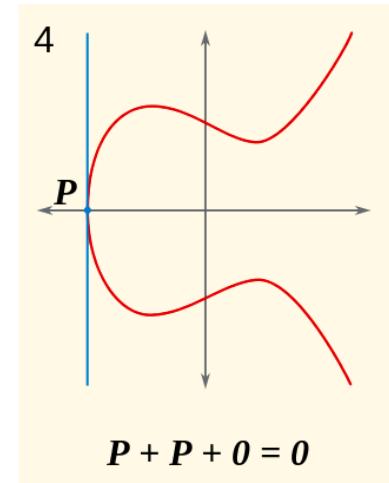
# Trap-door function

- 暗門函數
  - 單向驗證計算簡單，但逆推困難
- RSA
  - 紿定一個大數，試找出兩個質數的乘積等於這個大數？
- ECC
  - 紿定曲線上的起點與終點，試問經過多少個 hop？
  - $P + P + P + P + \dots + P = k * P$

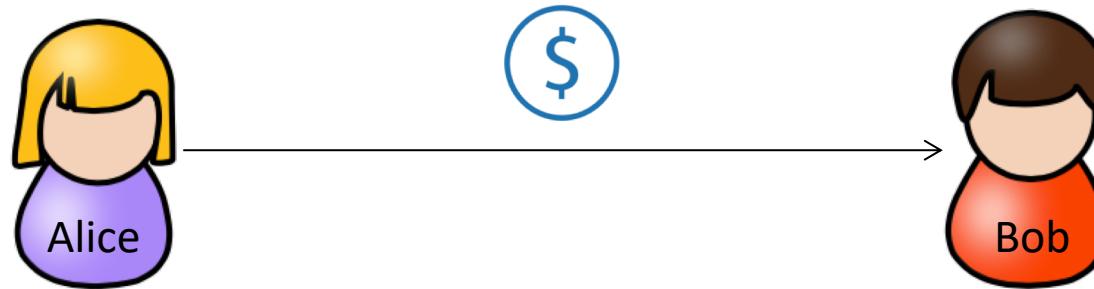
知道  $k$  要驗算相對容易，但要找到  $k$  很困難

\* Discrete Logarithm Problem (DLP)

為了求  $k$ ，目前的算法最快是  $O(\sqrt{n}) \approx O(2^{128})$



# 我, Alice, 要給 Bob 一個貨幣

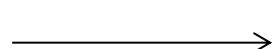


電子貨幣可能會被複製  
如何證明付款的是 Alice

# 我, Alice, 要給 Bob 一個貨幣

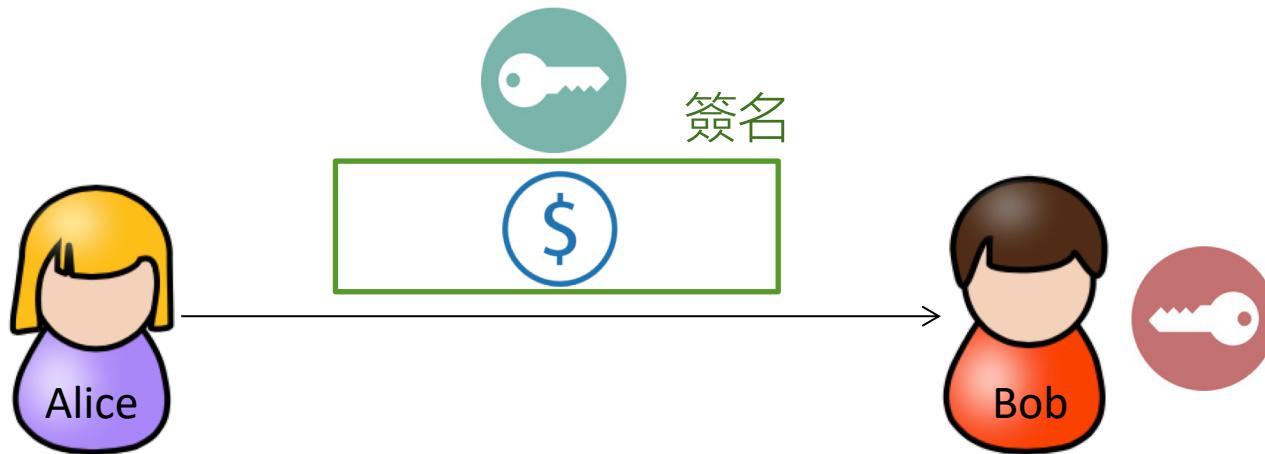


電子貨幣可能會被複製  
如何證明付款的是 Alice



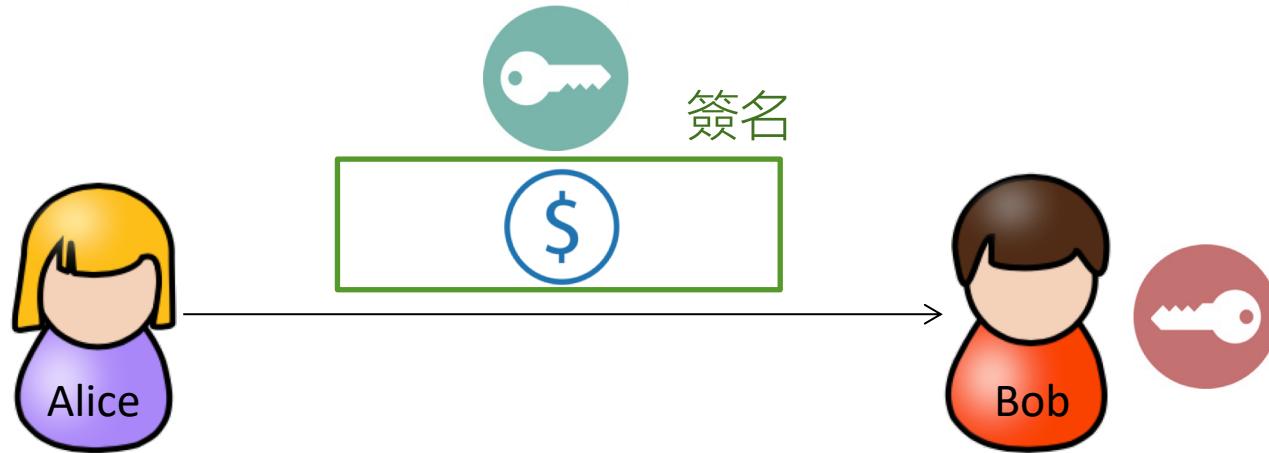
Alice 以私鑰對電子貨幣進行簽名  
Bob 以公鑰驗證簽名

# 我, Alice, 要給 Bob 一個貨幣



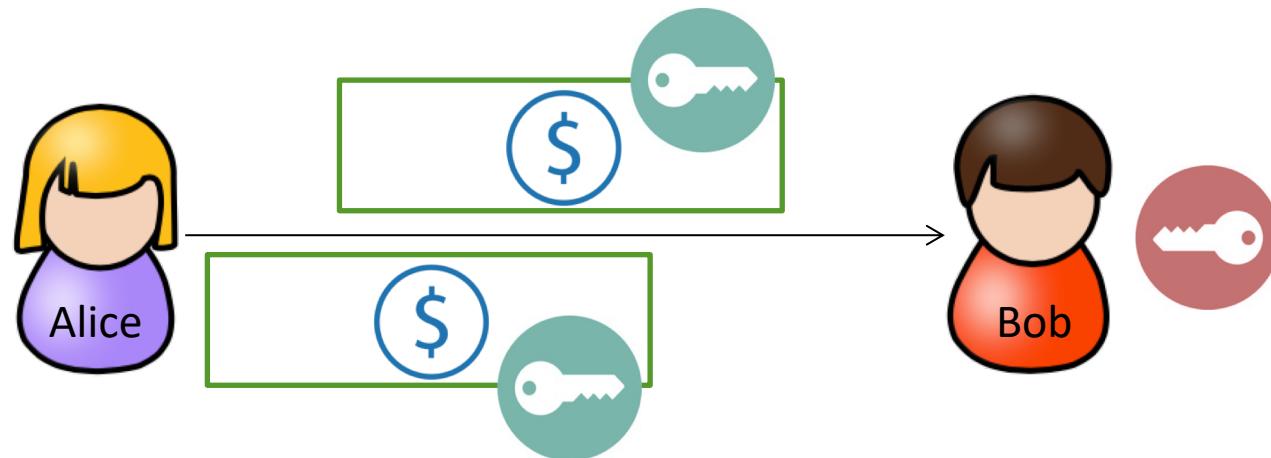
電子貨幣的來源可以驗證  
電子貨幣無法偽造

# 我, Alice, 要給 Bob 一個貨幣



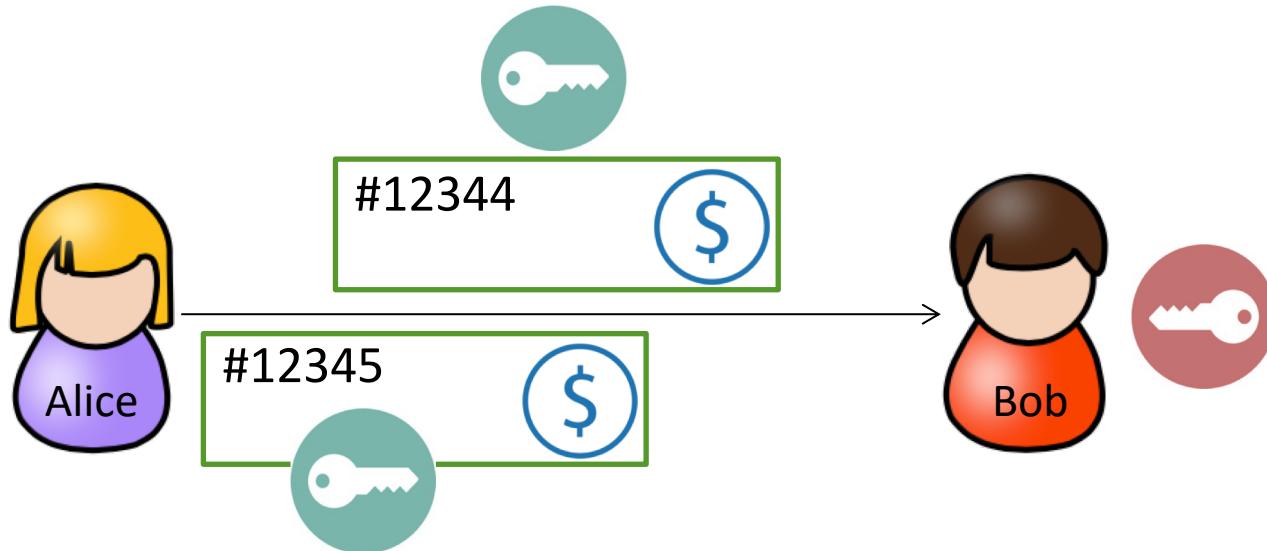
Alice 無法否認要給 Bob 一個貨幣的事實  
電子貨幣還是會被複製

# 貨幣無法辨識



Bob 無法辨識拿到的貨幣有何不同  
如何確認這個貨幣有沒有被二次消費

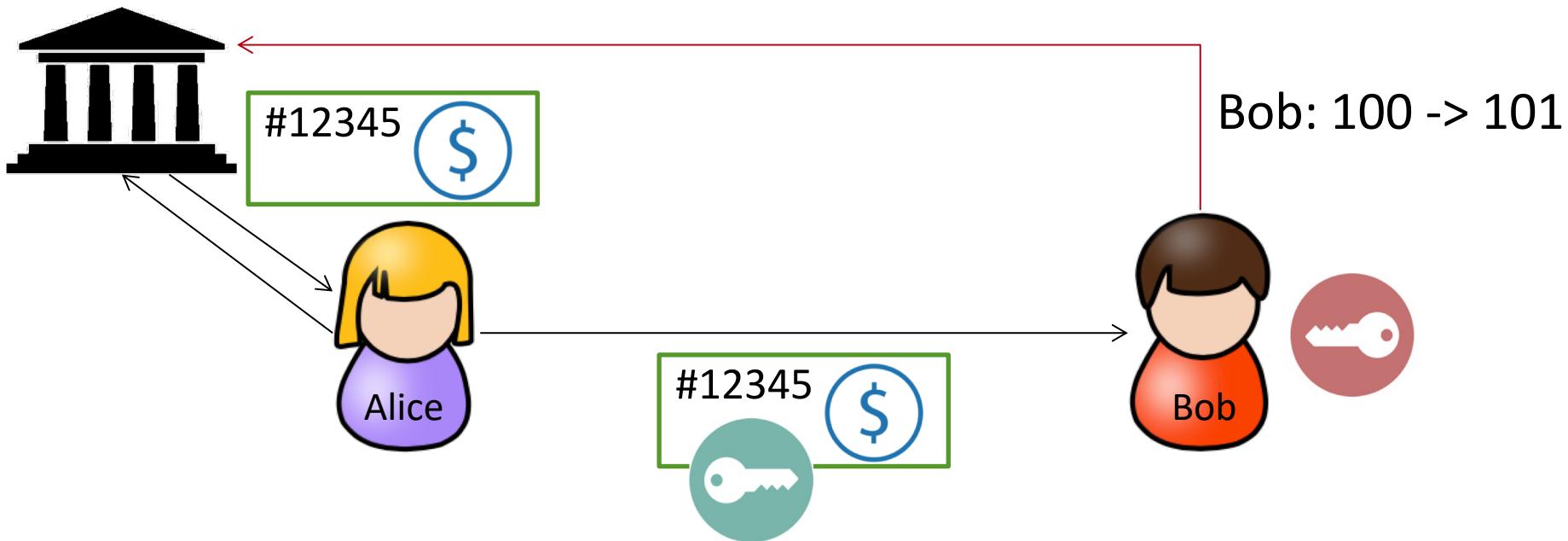
# 需要辨識貨幣 (流水號)



誰來管理貨幣流水號？需要一個可信任的機構組織

Alice: 100 -> 99

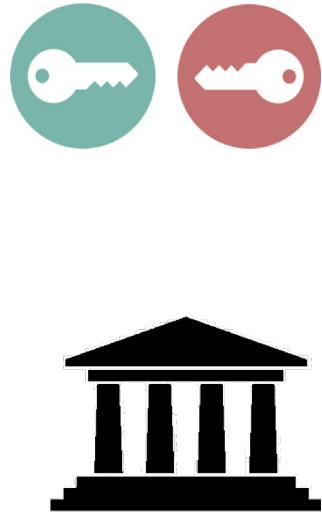
# 驗證交易是否合法



1. 這筆錢是不是 Alice 的？
2. 確認 Alice 沒花過這筆錢

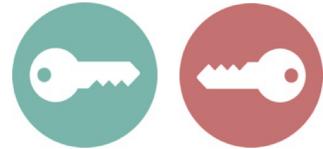
# 概括整理

- 非對稱式公私鑰
  - 貨幣無法偽造 (需要簽名)
  - 交易的不可否認性 (身分性)
- 可信任第三方
  - 管理資產
  - 追蹤貨幣流向
  - 避免雙花 (Double spending)



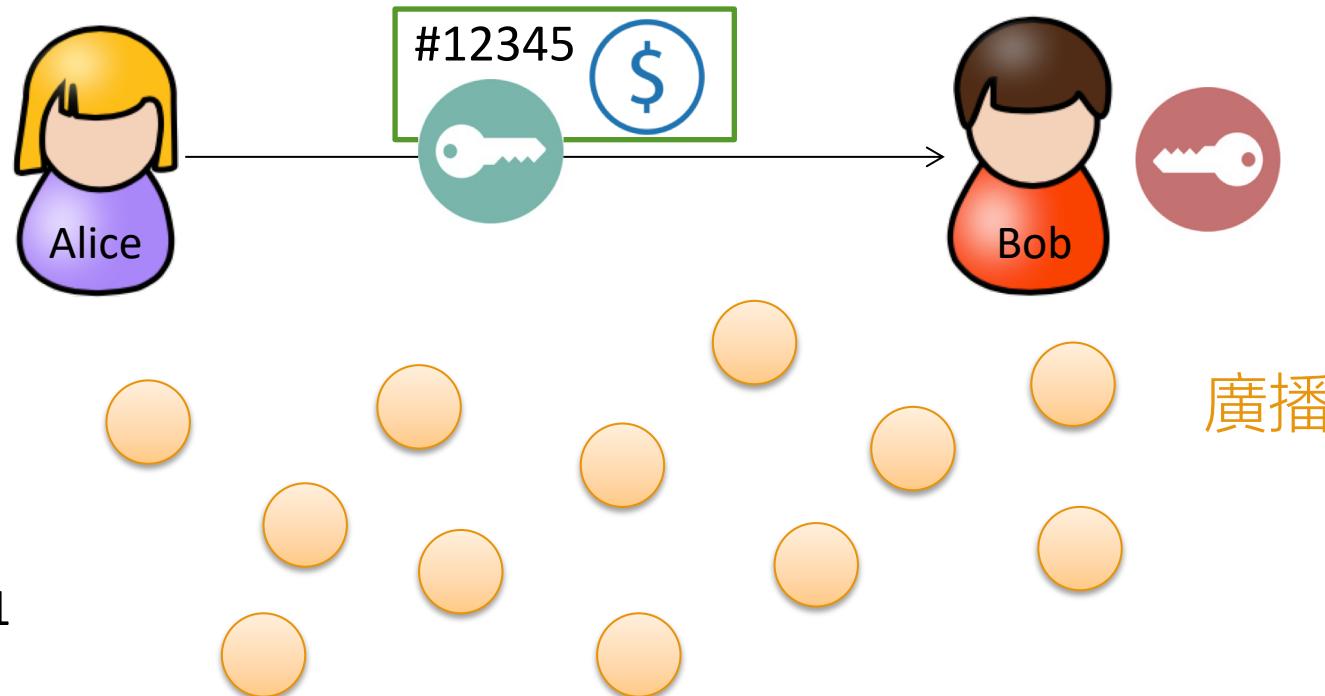
# 去中心化

- 非對稱式公私鑰
  - 貨幣無法偽造 (需要簽名)
  - 交易的不可否認性 (身分性)
- ~~可信任第三方~~
  - 管理資產
  - 追蹤貨幣流向
  - 避免雙花 (Double spending)



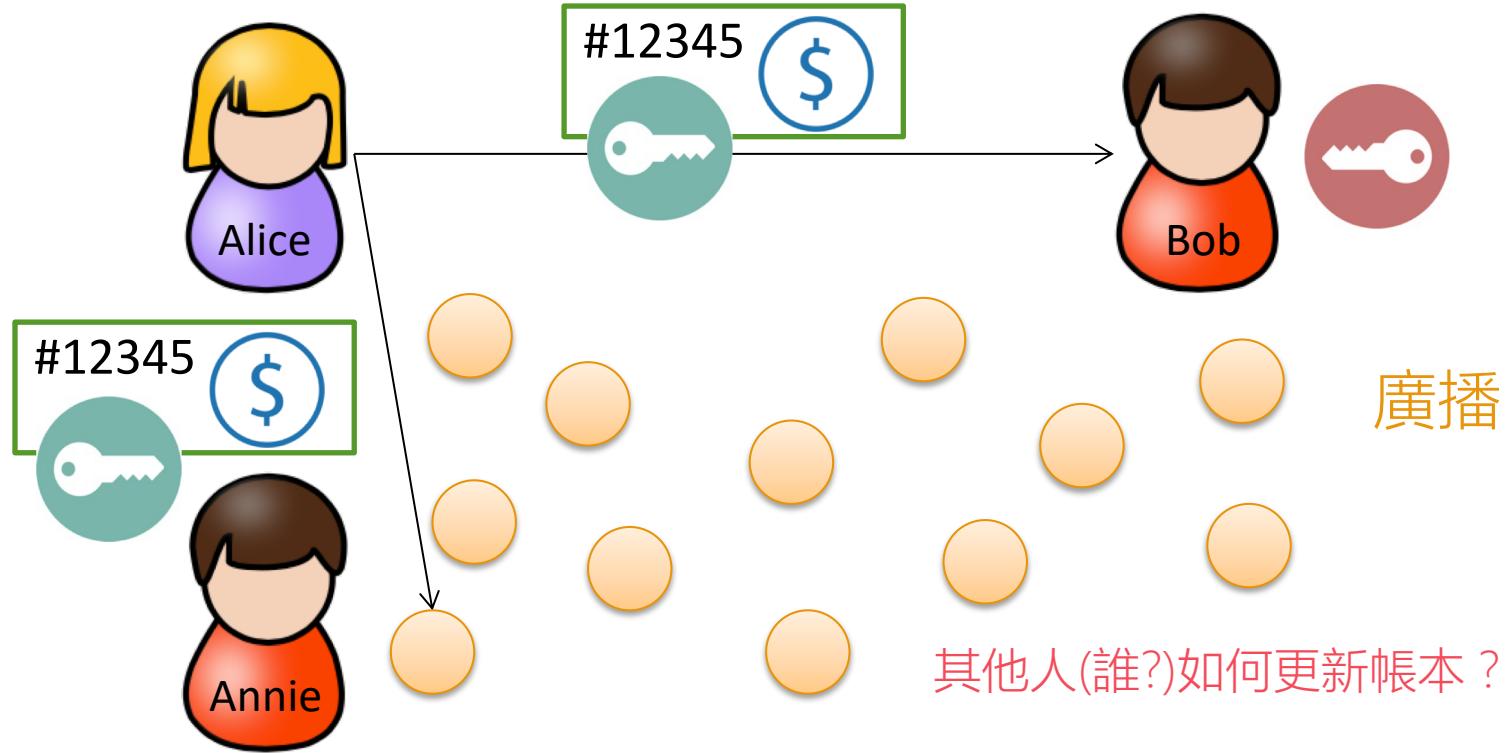
銀行帳本 → 公開透明且共享的帳本  
→ 區塊鏈 (Blockchain)

# 我, Alice, 要給 Bob 一個貨幣



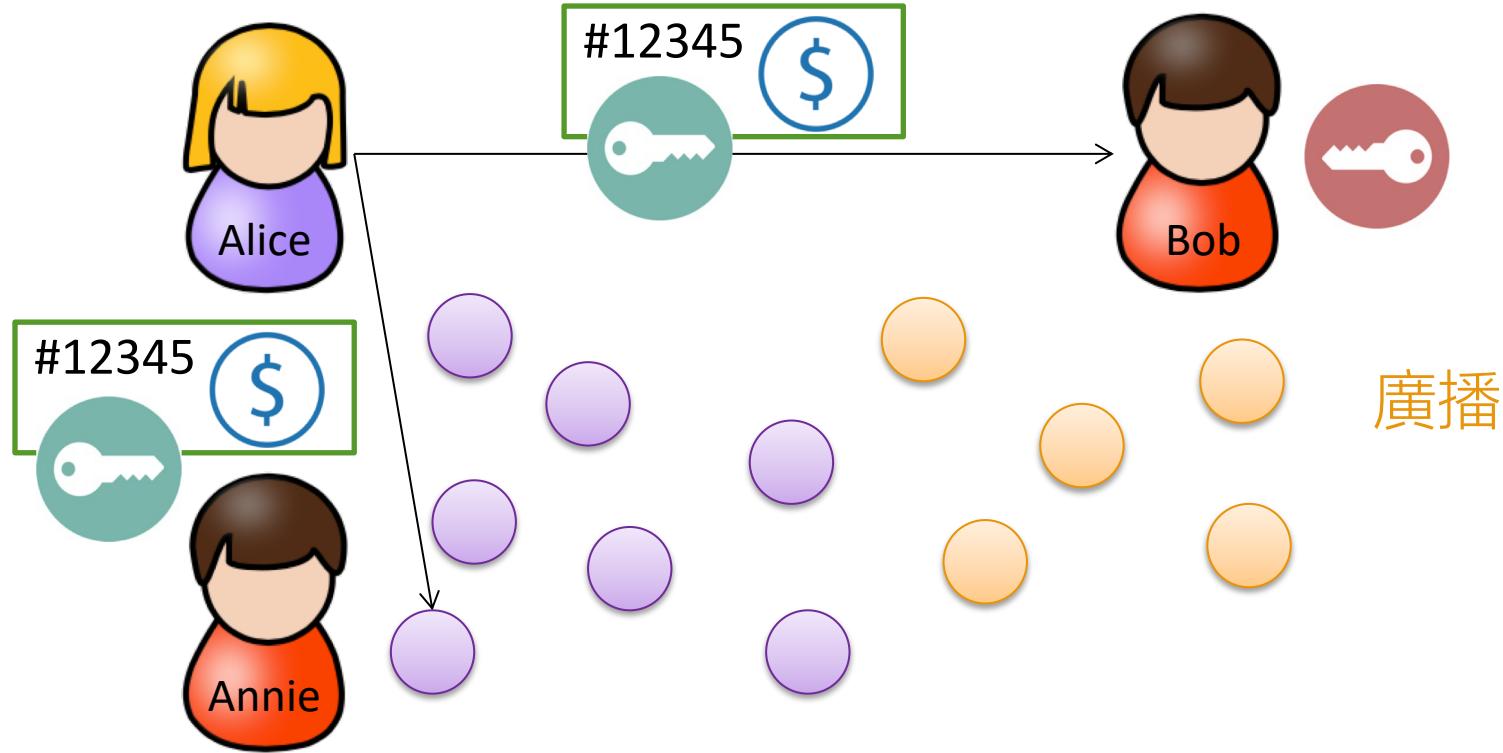
# 雙花攻擊

Alice 左手換右手  
把錢放到他的第二個帳戶

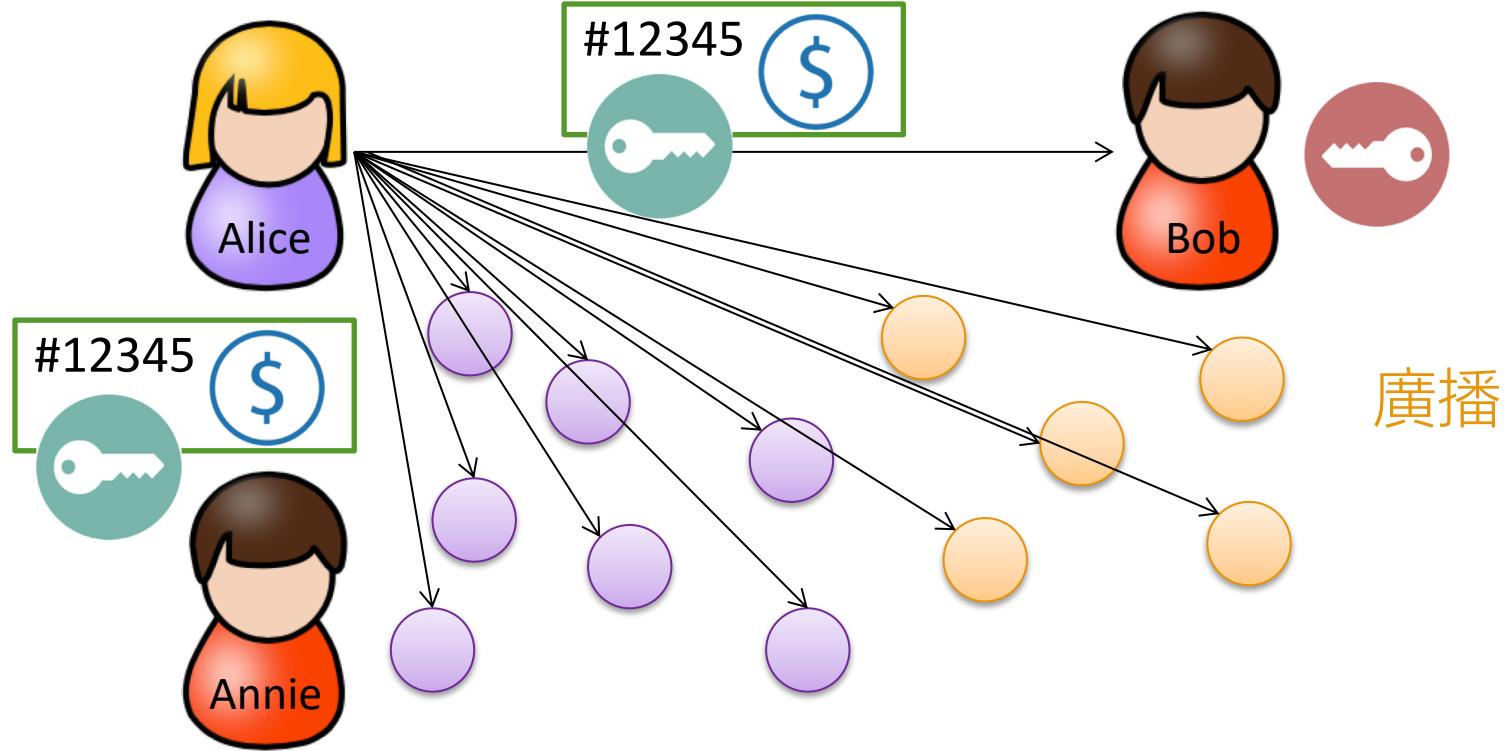


# 時間差

決定能否發動雙花攻擊



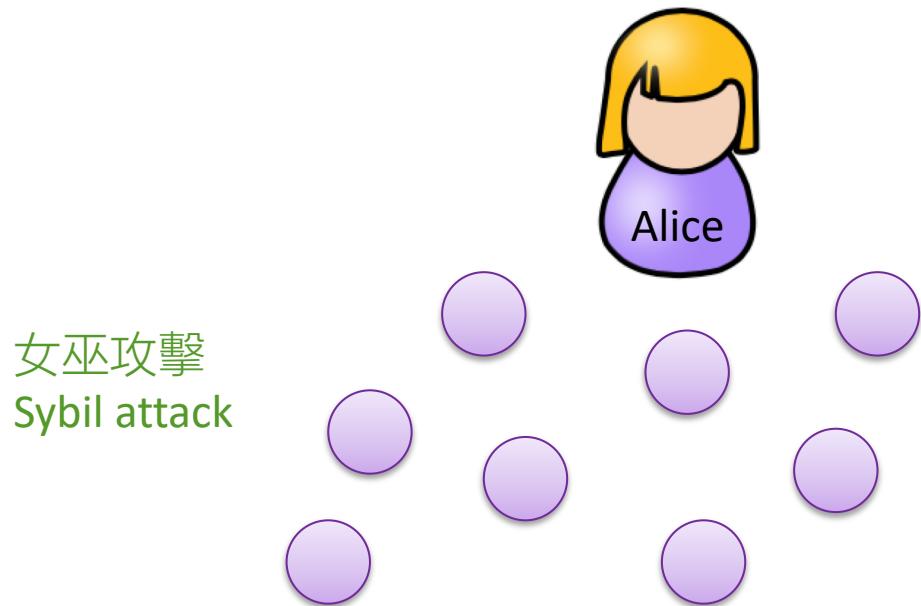
所有人都收到交易並作驗證，如果 Alice 送出同樣的貨幣給 Bob 和 Annie，問題就會產生



# 概括整理

- 如何避免雙花攻擊？
- 帳本多久更新？
- 需過多少人驗證才會更新帳本？

Alice 控制網路



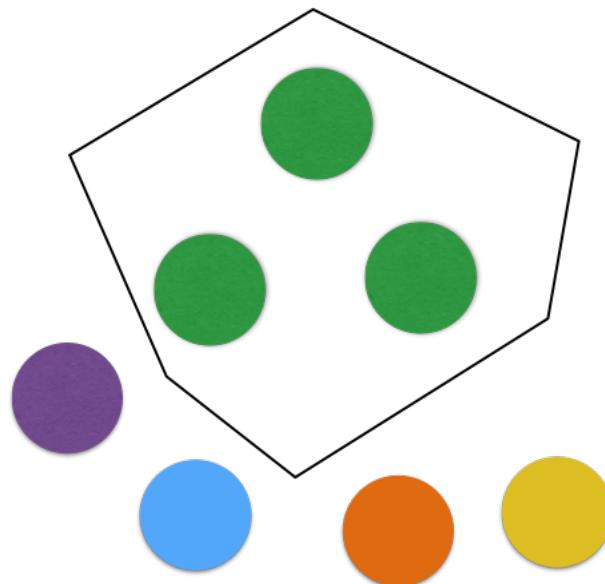
# 需要共識演算法

- 共識算法**是**為讓大家取得共識
- 但比特幣的共識算法**不是**讓大家藉由相互溝通形成共識
- 共識算法會選出一個代表人來描述現狀
  - 寫帳
  - 選出代表人的方式要公平 (隨機)
- 為鼓勵代表人來維護共識
  - 以**比特幣**作為獎勵打賞他 (鑄幣來源)

# Proof of Work (工作量證明)

- 網路用戶驗證交易計算成本高
  - 交易驗證無法被網路單元數目控制 (Sybil attack)
  - 提供獎勵(誘因)讓他們願意幫忙驗證交易
- 
- **解題時間長** (計算成本高 → 避免被女巫攻擊 → 系統慢)
  - **解題時間短** (高效 → 同時挖到礦機率高, 有分叉風險)

區塊產生時間  $T$  ( $\sim 10$  分鐘)  
→ ←



# PoW 設計哲學

- 每個人 (Decentralization) 都能參與解題 (Puzzle)
- 一般主機 (CPU as a Vote) 都能參與
- 不會有人掌控記帳權 (Randomize)
- 概率相同

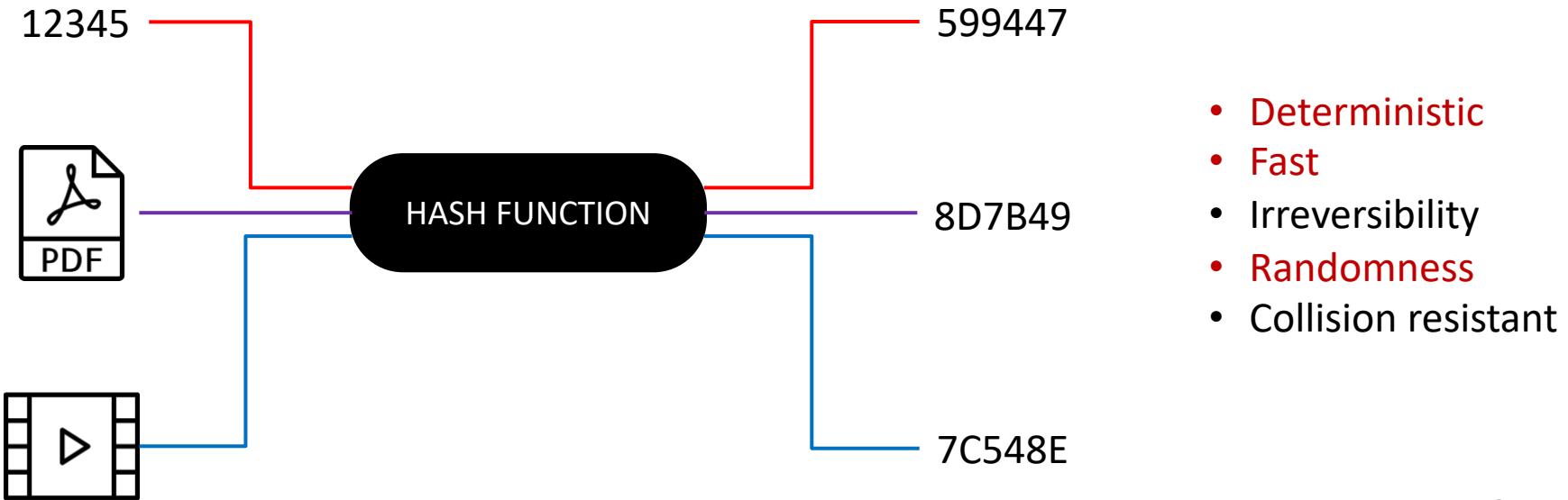


擲到數字小於 3 的人可以寫帳本

# Hash function

2. changwu@changwu-mbp: ~ (zsh)

```
(*) |ws:ws) sha256 12345  
5994471ABB01112AFCC18159F6CC74B4F511B99806DA59B3CAF5A9C173CACFC5  
(*) |ws:ws) sha256 /Users/changwu/Desktop/Screen\ Shot\ 2018-09-03\ at\ 11.26.20\ AM.png  
8D7B49815818A27ED1F9B5F71B53AF9AB9104CF1EF51A340FA6D6763036FF6E4  
(*) |ws:ws) sha256 /Users/changwu/An_Overview_Of_Governance_In_Blockchains.webm  
7C548EBA8675AE0970D688C394C286A06A67D8D79ACAC33378208076BC2C0EC1  
(*) |ws:ws) _
```



# Hash function

- 資料完整性
  - Checksums
  - Password hashing
- Proof of Work
  - Mathematical puzzle
- 資料身份性
  - Hash table
  - P2P networks
  - TX ID

# PoW 設計哲學

0x1011 = 11  
0x0101 = 05  
0x0011 = 03

- 解題困難，驗證容易，可調整難度 (Hash function)

SHA-256<sub>2</sub>("abc" + "1") =

158c09e82d88955d8a051934d12f74a53ea205743778165d1140a8903686e1ac

SHA-256<sub>2</sub>("abc" + "2") =

c72b0720d3302d76cd7b6b3f3dc554d05f14fee8567cdda3ee8b7ff51e02015

.

.

.

SHA-256<sub>2</sub>("abc" + "19") =



Hash(TXs+nonce) < number

00000000000000005b95bf7c8628e212d15a7d08c42235a381e06caa55856a0c

# PoW 設計哲學

- 解題困難，驗證容易，可調整難度 (Hash function)

SHA-256<sub>2</sub>("abc" + "1") =

158c09e82d88955d8a051934d12f74a53ea205743778165d1140a8903686e1ac

SHA-256<sub>2</sub>("abc" + "2") =

c72b0720d3302d76cd7b6b3f3dc554d05f14fee8567cdda3ee8b7ff51e02015

.

算力越強的人越有機會擁有記帳權

.

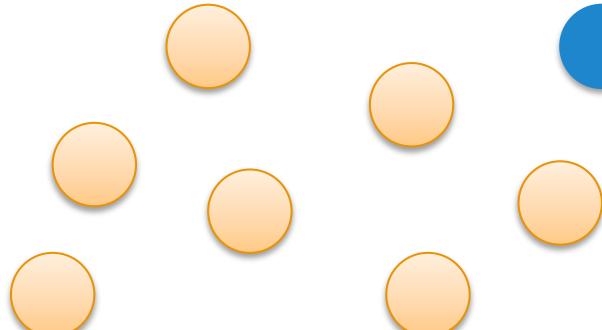
但不代表掌握 70% 算力別人就沒機會，只是概率低

.

SHA-256<sub>2</sub>("abc" + "19") =

005efb2ff3e871185b95bf7c8628e212d15a7d08c42235a381e06caa55856a0c

# 解題成功舉手廣播



獲得獎勵

為什麼要給獎勵？  
人少？人多？

# 概括整理

- 如何避免雙花攻擊？
  - 以 PoW 解決共識問題
  - 讓交易驗證分散到各個節點
- ~~帳本多久更新？~~
- ~~需要多少人驗證才會更新帳本？~~

Alice 控制網路

