以密碼學為基礎實作建構出區塊鏈應用常見的錢包地址

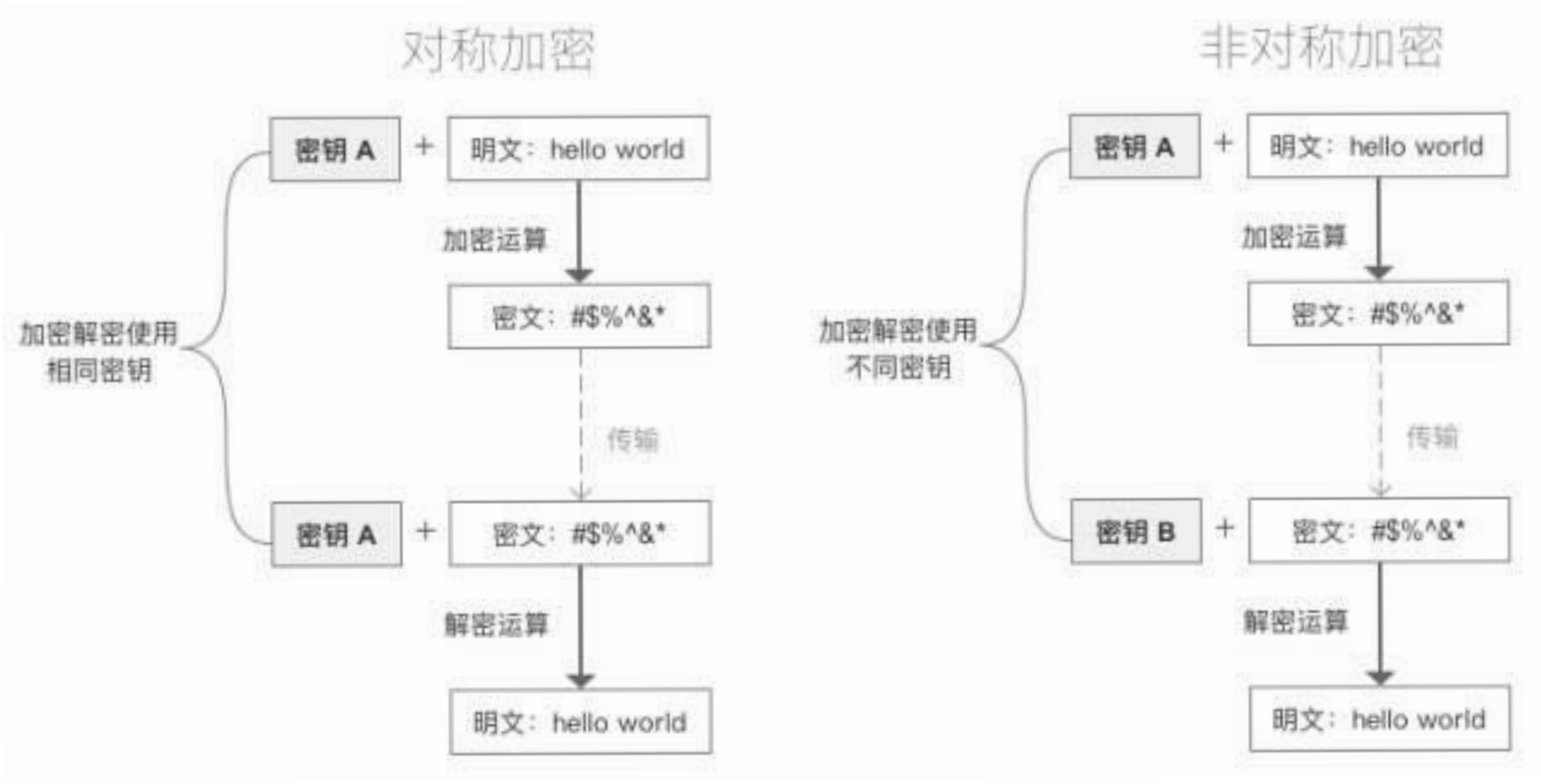
盧瑞山 教授

先從比特幣的
安全性說起

Alice 轉帳給Bob 0.5枚比特幣

- Alice有足夠的比特幣嗎？
- 如何確認是Alice 發出的交易申請？
- 怎麼轉帳給Bob？
- 交易的整個過程是什麼？
- 誰來確保交易的安全？
- Bob何時收到款項？
- Bob何時可以花費收到的款項？
- Bob的帳戶裡的比特幣餘額安全嗎？

基礎密碼學



對稱式加解密法(編碼解碼使用同一個key)

1. 編碼　key+原始資料－＞編碼資料
2. 解碼　key+編碼資料－＞原始資料

非對稱式加解密法(編碼解碼使用不同的key)

1. 編碼　key1+原始資料－＞編碼資料
2. 解碼　key2+編碼資料－＞原始資料

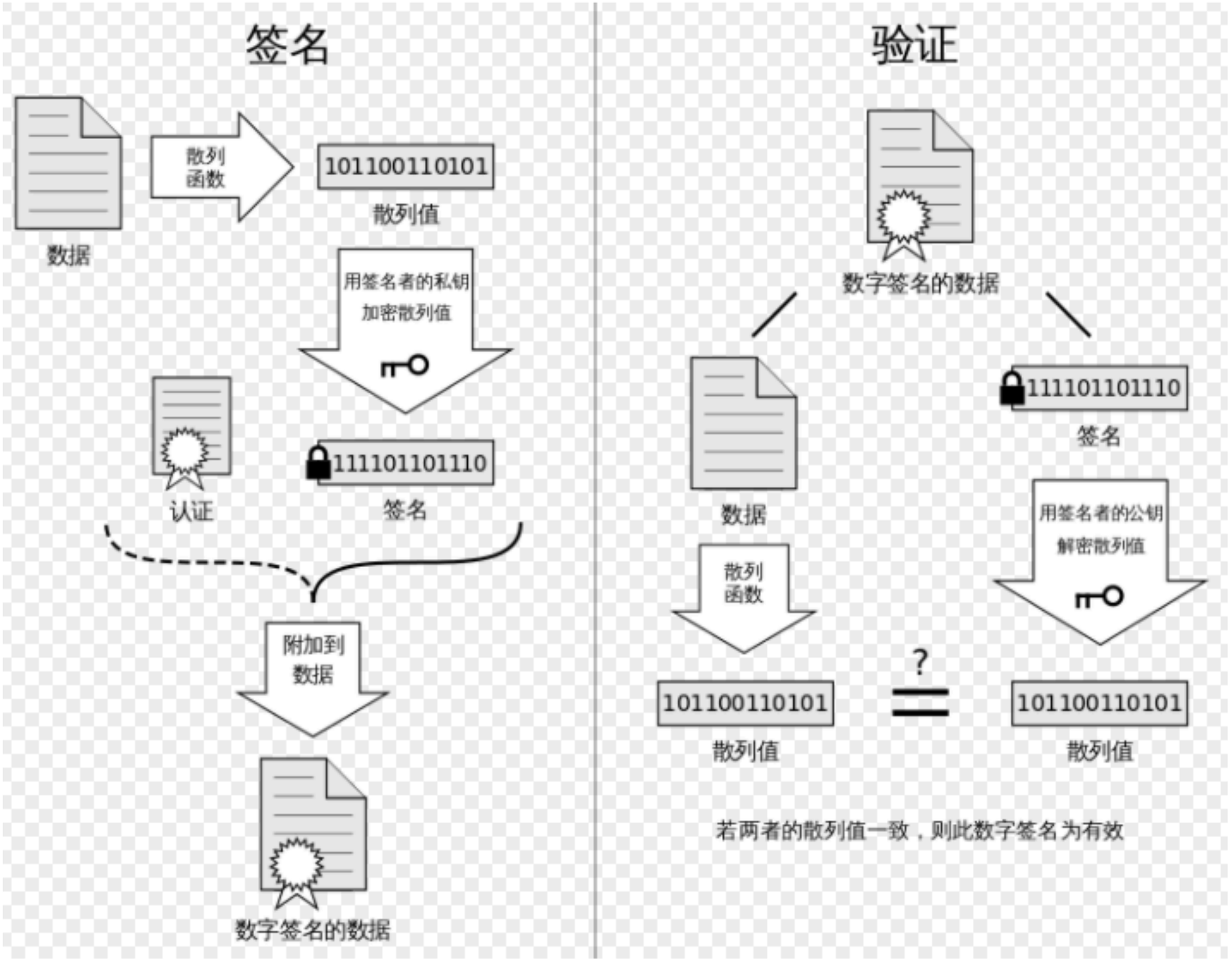
觀念的澄清與補充

注意，RSA 非對稱式加解密演算法因為先天的限制，無法加密過大的檔案，若遇到此問題時，OpenSSL 會輸出如下的錯誤訊息：

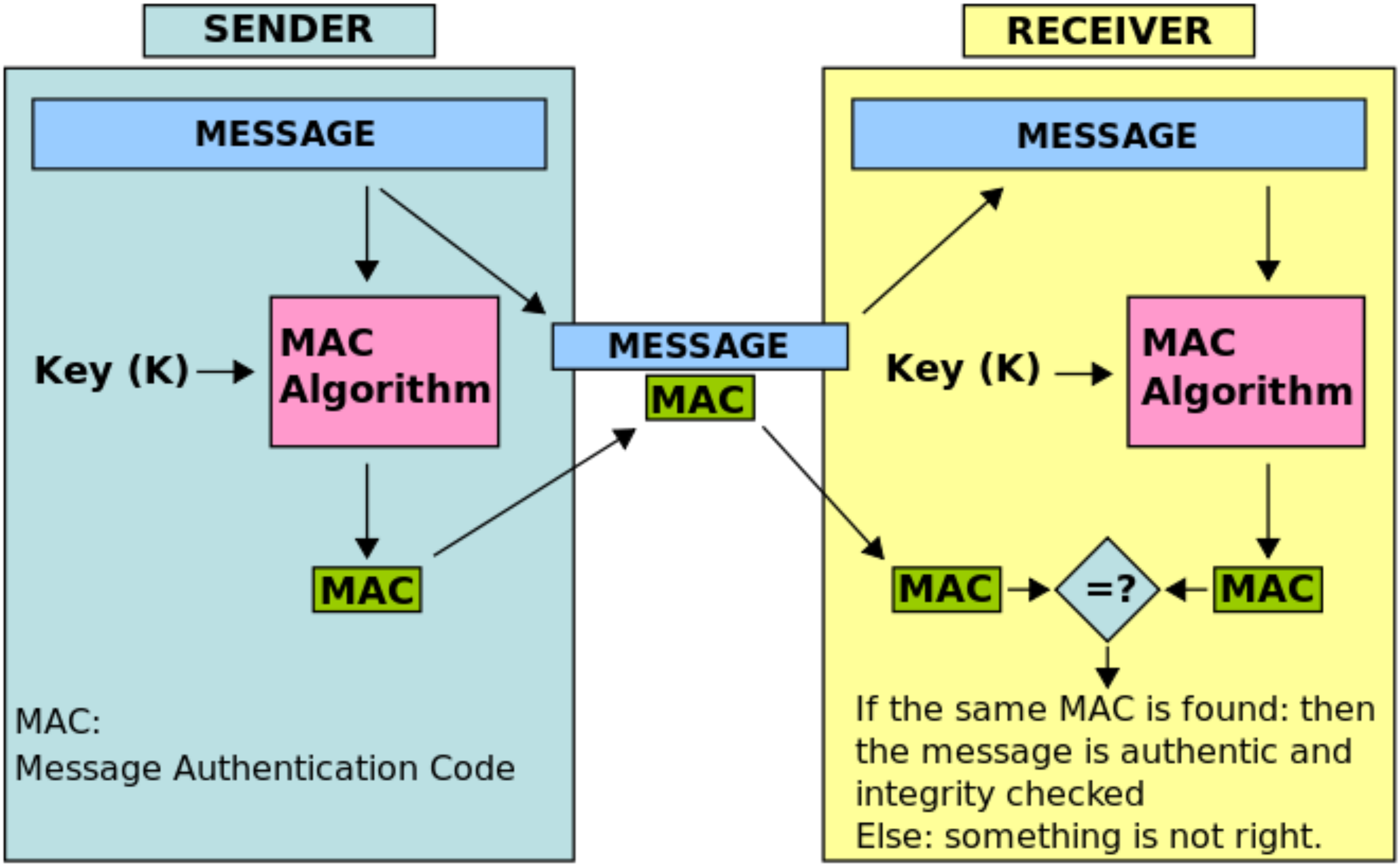
```
RSA operation error
13931:error:0406D06E:rsa routines:RSA_padding_add_PKCS1_type_2:data too large for key size:rsa_pk1.c:
151:
```

實務應用上，可採對稱加密演算法與非對稱加密演算法混用。
例如：visa,master等組織所推的SET協議是專門用在網上交易的電子支付交易協議，其就是將交易內容先用DES加密，再用RSA加密一次

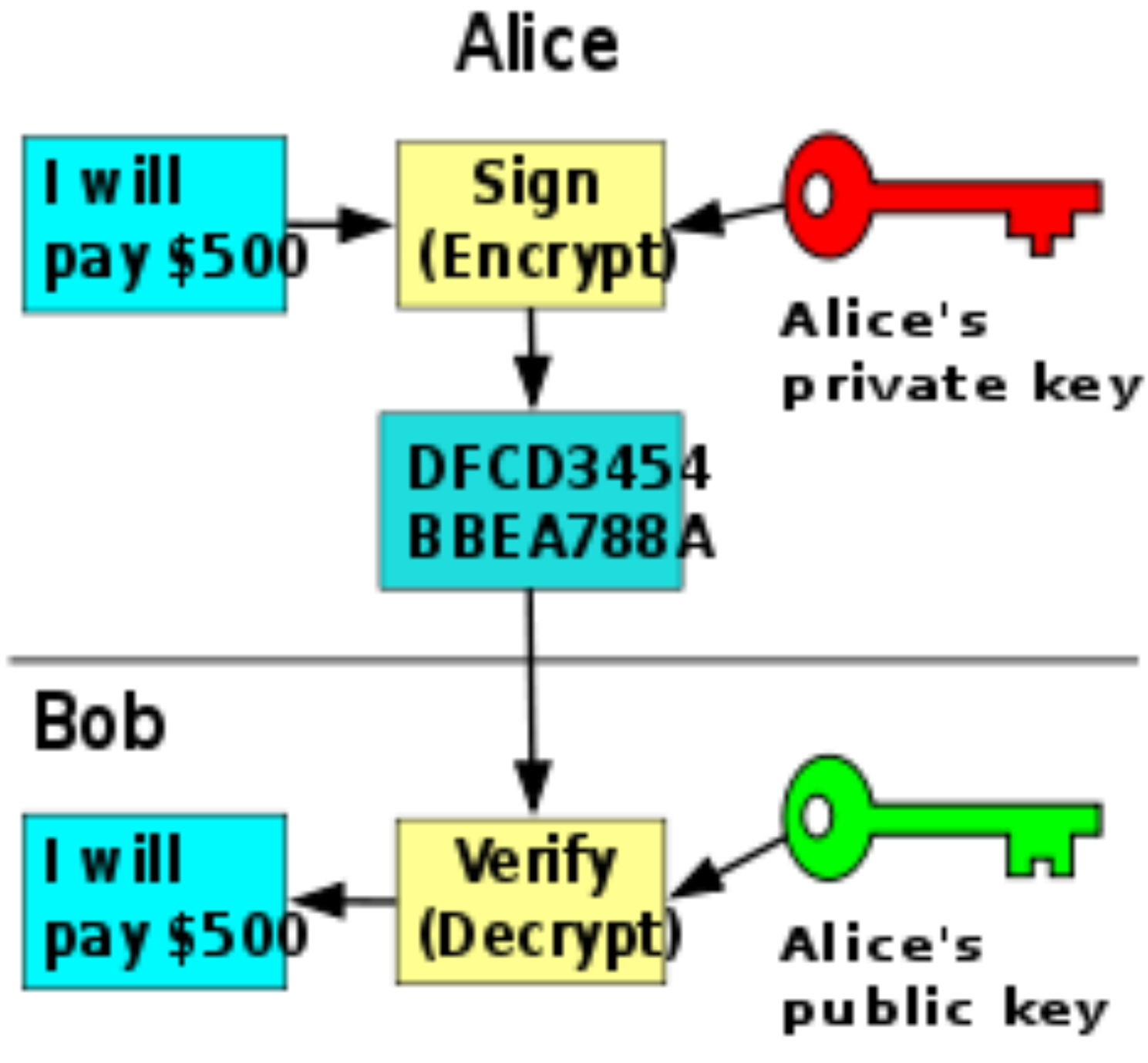
數位簽章原理


签名
数据
散列
函数
101100110101
散列值
用签名者的私钥
加密散列值
认证
111101101110
签名
附加到
数据
数字签名的数据
验证
数字签名的数据
111101101110
签名
数据
用签名者的公钥
解密散列值
散列
函数
?
101100110101
散列值
101100110101
散列值
若两者的散列值一致，则此数字签名为有效

H-MAC (**H**ash-based **M**essage **A**uthentication **C**ode)



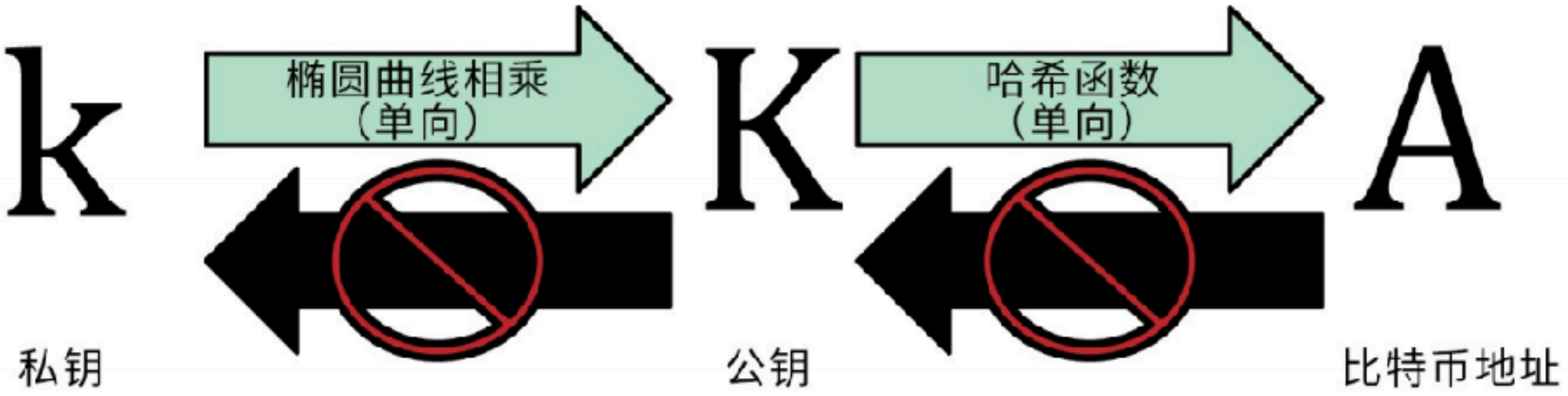
Alice pay 500 to Bob



用Bitcoin Core實作

從公鑰到私鑰的原理

比特幣地址的生成原理



參照課本第四章

如何產生私鑰？

比特幣的私鑰是一個256位元長度的亂數。

從編程的角度看, 從一個符合密碼學安全的隨機長字串, 然後對其運算SHA256, 這樣就可以產生**256**位元的數字, 然後檢驗是否小於n-1 其中n=1.158 * 10^77

比特币私钥空间的大小是2^{256}，这是一个非常大的数字。用十进制表示的话，大约是10^{77}，而可见宇宙被估计只含有10^{80}个原子。

openSSL Demo

產生 ECC 公私鑰(跟產生 RSA 私鑰是類似)

產生私鑰指令：

openssl ecparam -out ecckey -name prime256v1 -genkey

openssl ecparam -out ecckey -name secp256k1 -genkey

產生公鑰指令

openssl ec -in ecckey -pubout -out ecpubkey.pem **以檔案輸出**

openssl ec -in ecckey -noout -text (以Hex明文在螢幕上輸出)

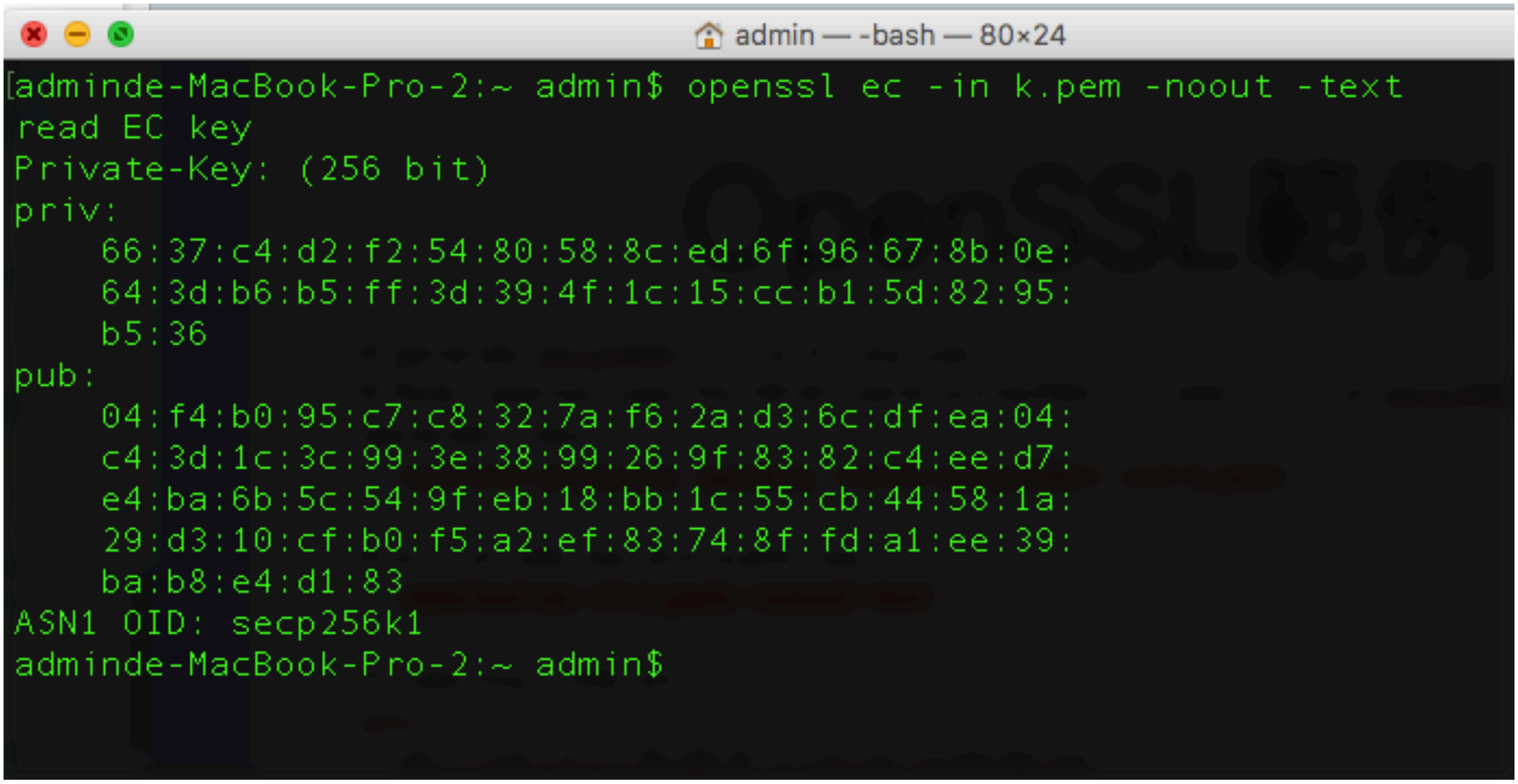
OpenSSL範例

generate secp256r1 curve EC key pair
Note: openssl uses the X9.62 name prime256v1 to refer to curve secp256r1, so this will generate output
% **openssl ecparam -genkey -name secp256r1 -out k.pem**

print private key and public key
% **openssl ec -in k.pem -noout -text**

```
Private-Key: (256 bit)
priv:
    00:11:b5:73:7c:f9:d9:3f:17:c0:cb:1a:84:65:5d:
    39:95:a0:28:24:09:7e:ff:a5:ed:d8:ee:26:38:1e:
    b5:d6:c3
pub:
    04:a0:15:32:a3:c0:90:00:53:de:60:fb:ef:ef:cc:
    a5:87:93:30:15:98:d3:08:b4:1e:6f:4e:36:4e:38:
    8c:27:11:be:f4:32:c5:99:14:8c:94:14:3d:4f:f4:
    6c:2c:b7:3e:3e:6a:41:d7:ee:f2:3c:04:7e:a1:1e:
    60:66:7d:e4:25
ASN1 OID: prime256v1
```

openssl ec -in k.pem -noout -text 的輸出結果



利用Pybitcointools從字串變成亂數,再由亂數產生公私鑰再產生錢包地址

pybitcointools

```
$ git clone https://github.com/vbuterin/pybitcointools.git
$ cd pybitcointools
$ python3
> from bitcoin import *
> priv = sha256('some big long brainwallet password')
> priv
'57c617d9b4e1f7af6ec97ca2ff57e94a28279a7eedd4d12a99fa11170e94f5a4'
> pub = privtopub(priv)
> pub
'0420f34c2786b4bae593e22596631b025f3ff46e200fc1d4b52ef49bbdc2ed00b26c5
84b7e32523fb01be2294a1f8a5eb0cf71a203cc034ced46ea92a8df16c6e9'
> addr = pubtoaddr(pub)
> addr
'1CQLd3bhw4EzaURHbKCwM5YZbUQfA4ReY6'
> h = history(addr)
```