2. The RSA System

前面提過, 對於 symmetric 的密碼系統, key 的保存及傳送問題是困難解決的. 而 public key 的概念是解決這問題的一個方法. RSA 系統是 public key 中最常被提及也最容易被理解的系統. 在這一節中, 我們介紹 RSA 系統.

Public key 的想法是接收者公布他的 public key, 任何要傳輸資料給他的人, 只要用他的 public key 加密資料給他. 而接收者便用他的 private key 解密. 既然 encryption key 公開, 表示他人很難從 encryption key 得到 decryption key. 所以使用 public key 的密碼系統當然是 asymmetric system. 我們先說明 RAS 系統的執行步驟, 在說明它確為密碼系統, 最後說明它的效率性以及安全性.

2.1. **執行步驟.** 首先先挑選兩個 "很大的" 相異質數 p,q, 然後計算 $n=pq$ 以及 $\phi=\phi(n)=(p-1)(q-1)$. 接著選出 e 滿足 $\gcd(e,\phi)=1$ (即和 ϕ 互質) 並計算求出 d 滿足 $d\cdot e\equiv 1 \pmod{\phi}$. 這裡要公布的 public key 是 n,e, 而保持機密的 private key 是 d. 任何人知道 public key n,e 就可以將要加密的 plaintext m (需滿足 $0\le m<n$), 使用 encryption function

$$E_{n,e}(m)=c\equiv m^e \pmod{n},\quad 0\le c<n$$

加密. 收到 c 後就可以用 private key, 以及 decryption function

$$D_d(c)=E_{n,d}(c)\equiv c^d \pmod{n}$$

解密. 這裡要注意, 只有 public key n,e 可以公開. 雖然我們稱 d 為 private key, 但除了 d 外, p,q,ϕ 也不能公開.

我們也談論一下在 RSA 系統中如何將 plaintext 數字化. 一般來說所選的 n 應該相當的大, 用十進位表示應該大於 300 位數. 假設 plaintext 想用 r 進位表示, 先取 s 為小於等於 $\log_r[n]$ 的最大正整數. 接著就將 plaintext 每 s 個一組用 r 進位寫出. 也就是說 $m_1m_2\ldots m_{s-1}m_s$ 表示數值為 $m=m_1r^{s-1}+m_2r^{s-2}+\cdots+m_{s-1}r+m_s$. 這樣我們就會有

$$m\le(r-1)r^{s-1}+(r-1)r^{s-2}+\cdots+(r-1)r+(r-1)=r^s-1<n.$$

例如 plaintext 為英文, 可將英文字母以及空白鑑共 27 個符號用 2 進位編碼. 這樣每個符號要五位數表示. 比方, 空白用 00000, A 用 00001,..., Z 用 11010. 如果選的 n 大約 10^{31}, 則用二進位表示 $\log_2 n$ 約為 100. 此時我們可以將 plaintext 的英文字母 (包含空白) 每 24 個一組. 這樣每一組就會是用二進位表示的 100 位數的數字.

最後我們強調一下, 用上述方法將 plaintext 編碼會是一個用 r 進位表示有 s 位數的 block m. 不過加密後 $E_{n,e}(m)=c$ 因為僅知會小於 n, 所以 c 得到後用 r 進位來表示會是 $c=c_1r^s+c_2r^{s-1}+\cdots+c_sr+c_{s+1}$, 其位數會是 $s+1$.

2.2. **正確性.** 接下來我們說明 RSA 確為一個密碼系統. 我們需要處理以下兩件事

(1) 假設 p,q 為兩相異質數且令 $n=pq$, $\phi=(p-1)(q-1)$. 任取 $e\in\mathbb{N}$ 滿足 $\gcd(e,\phi)=1$, 則必存在 $d\in\mathbb{N}$ 滿足 $d\cdot e\equiv 1 \pmod{\phi}$.

(2) 任取 $m\in\mathscr{M}$, n,e 為 public key, d 為 private key, 則 $D_d(E_{n,e}(m))=m$, 即 $(m^e)^d\equiv m \pmod{n}$.

(1) 成立的原因是因為 e,ϕ 互質, 故存在整數 a,b 滿足 $a\cdot e+b\cdot\phi=1$. 因此若令 $d=a$, 則 $d\cdot e=1-b\cdot\phi\equiv 1 \pmod{\phi}$.

至於 (2), 首先我們敘述兩個基本數論的定理, 這兩個定理有更一般的形式, 不過這裡我們簡化僅敘述目前需要的情形, 也不提供證明. 有需要進一步了解的同學, 請自行參閱 elementary number theory 相關書籍.

Lemma 2.1 (Euler)**.** 假設 p 為質數, $a\in\mathbb{Z}$ 滿足 $p\nmid a$, 則 $a^{p-1}\equiv 1 \pmod{p}$.

Lemma 2.2 (Chinese Remainder Theorem)**.** 假設 p,q 為兩相異質數. 任取 $a,b\in\mathbb{Z}$ 皆存在唯一的 $m\in\mathbb{Z}$ 滿足 $0\le m<pq$ 且 $m\equiv a \pmod{p}$, $m\equiv b \pmod{q}$.

有了 Lemma 2.1 和 Lemma 2.2, 我們就可以證得以下之結果, 並得到 (2) 成立.

Theorem 2.3. 假設 p,q 為兩相異質數且令 $n=pq$, $\phi=(p-1)(q-1)$. 假設 $e\in\mathbb{N}$ 滿足 $\gcd(e,\phi)=1$ 且 $d\in\mathbb{N}$ 滿足 $d\cdot e\equiv 1 \pmod{\phi}$. 則對任意 $m\in\mathbb{N}$, $0\le m<n$ 皆有 $(m^e)^d\equiv m \pmod{n}$.

Proof. 由 $d\cdot e\equiv 1 \pmod{\phi}$ 我們假設 $de=1+\lambda\phi=1+\lambda(p-1)(q-1)$, 其中 $\lambda\in\mathbb{Z}$. 現考慮 $m^{ed}=m^{1+\lambda(p-1)(q-1)}$, 在 modulo p 的情形. 因為

$$m^{ed}=m\cdot(m^{p-1})^{\lambda(q-1)}. \tag{2.1}$$

若 $p\mid m$ (即 $m\equiv 0 \pmod{p}$), 我們自然有 $m^{ed}\equiv 0\equiv m \pmod{p}$. 而若 $p\nmid m$, 由 Lemma 2.1 知 $m^{p-1}\equiv 1 \pmod{p}$, 故由式子 (2.1) 得 $m^{ed}\equiv m \pmod{p}$. 同理, 考慮 modulo q, 我們有 $m^{ed}\equiv m \pmod{q}$. 現考慮 m^{ed} 除以 n 的餘數 r, 我們知 $0\le r<n$, 而且 $m^{ed}\equiv r \pmod{n}$. 又因 $n=pq$, 故 $m^{ed}\equiv r \pmod{p}$ 且 $m^{ed}\equiv r \pmod{q}$. 換言之 r 符合 $r\equiv m \pmod{p}$, $r\equiv m \pmod{q}$ 以及 $0\le r<pq$. 然而 m 也符合同樣的式子, 故由 Lemma 2.2 的唯一性知 $r=m$, 故得證本定理. □

由以上, 我們確認 RSA system 確為密碼系統.

2.3. **效率性.** 可以預知的, 一個 asymmetric 編碼系統因為 encryption function 複雜, 所以才會造成 encryption function 很難找到. 所以我們也要探討 RSA 系統執行的效率性.

首先遇到的問題是給定 e 滿足 $\gcd(e,\phi)=1$, 如何找出 d 使得 $d\cdot e\equiv 1 \pmod{\phi}$. 前面提過, 找出 d 的方法是找到方程式 $ex+\phi y=1$ 的一組整數解 x,y. 怎樣找到這樣的一組整數解呢? 最簡單的方法就是使用輾轉相除法. 回顧一下輾轉相除法原理是說, 要求兩整數 a,b (其中 $a>b$) 的最大公因數, 我們可以將 a 除以 b 得餘數 r, 則 $\gcd(a,b)=\gcd(b,r)$. 由於餘數 r 滿足 $0\le r<b$, 所以輾轉相除法可以將求兩個較大的數公因數轉換成較小的數的公因數. 也就是說要求 a,b 的最大公因數, 我們可以將 a 除以 b 得 $a=bh+r$, 其中 $0\le r<b$. 若 $r=0$ 表示 b 整除 a, 故 $\gcd(a,b)=b$ (也可用輾轉相除法原理得 $\gcd(a,b)=\gcd(b,0)=b$). 若 $r\neq 0$, 則可再將 b 除以 r 得餘數 r', 此時我們有 $\gcd(a,b)=\gcd(b,r)=\gcd(r,r')$. 由於這樣操作每次所得餘數會比前一次小, 所以只要操作有限多次後就可得到餘數為 0, 此時前一次的餘數就是最大公因數了. 使用輾轉相除法不只可以幫助我們快速的找到兩個整數 a,b 的最大公因數 d, 也能依此將 d 寫成 a,b 的整係數線性組合 (即 $d=ma+nb$, $m,n\in\mathbb{Z}$). 這

是因為由 a 除以 b 的式子, $a = bh + r$, 我們可以將餘數 r 寫成 a,b 的整係數線性組合. 換言之, 餘數可以寫成除數與被除數的整係數線性組合, 所以我們只要將輾轉相除法求最大公因數的式子逆推回去, 就可將之寫成 a,b 的整係數線性組合.

在 RSA 系統中, 我們找到 e 滿足 $\gcd(e,\phi) = 1$, 此時就可以用輾轉相除法將 1 寫成 e 和 ϕ 的整係數線性組合 $1 = r \times e + s \times \phi$, 其中 $r,s \in \mathbb{Z}$. 此時令 $d = r$, 則 d 就會滿足 $d \cdot e \equiv 1 \pmod{\phi}$. 我們用以下的例子說明.

Example 2.4. 考慮 $\phi = 24$, $e = 17$, 利用輾轉相除法, 我們有以下式子: $24 = 17 \times 1 + 7$, $17 = 7 \times 2 + 3$, $7 = 3 \times 2 + 1$. 故知 $\gcd(24,17) = 1$. 最後一個式子可將 1 寫成 3,7 的線性組合, 即 $1 = 1 \times 7 - 2 \times 3$. 接下來將 3 寫成 17,7 的線性組合, 即 $3 = 1 \times 17 - 2 \times 7$ 再代入前一式得 $1 = 1 \times 7 - 2 \times (1 \times 17 - 2 \times 7) = 5 \times 7 - 2 \times 17$, 也就是說我們將 1 寫成 17,7 的線性組合. 最後將 7 寫成 24,17 的線性組合, 即 $7 = 1 \times 24 - 1 \times 17$ 再代入上一式, 得 $1 = 5 \times (1 \times 24 - 1 \times 17) - 2 \times 17 = 5 \times 24 - 7 \times 17$. 我們成功地將 1 寫成 24,17 的整係數線性組合. 最後找最小的正整數滿足 $d \equiv -7 \pmod{24}$, 即 $d = 17$

在 RSA 系統中另一個要處理的問題就是要計算 $m^e \bmod n$ (即除以 n 的餘數). 這裡如果直接計算 m^e, 要處理 $e-1$ 次計算. 而若用 repeated-squaring (多次平方) 的方法, 可以減少乘法使用的次數. 在此我們約略說明處理的方法. 首先計算 m^2, m^4, m^8,..., m^{2^r}, 其中 r 為小於等於 $\log_2 e$ 的最大正整數. 在這個步驟我們僅需做 r 次的乘法, 而且每次數值若超過 n 可以用除以 n 的餘數取代以減少計算. 接著將 e 用二進位表示, 也就是說將 e 寫成 $e = e_1 2^r + e_2 2^{r-1} + \cdots + e_r 2 + e_{r+1}$, 其中 $r_i \in \{0,1\}$. 此時我們有

$$m^e = m^{e_1 2^r} \cdot m^{e_2 2^{r-1}} \cdots m^{e_r 2} \cdot m^{e_{r+1}}.$$

因為這裡 e_i 非 0 即 1 所以最多還要做 r 次乘法. 也就是說這個方法最多需要 $2\log_2 e$ 次的乘法, 明顯少於 $e-1$. 我們用以下的例子說明.

Example 2.5. 假設 $n = 35$, $m = 10$, $e = 17$. 因小於 $\log_2 17$ 的最大整數為 4. 我們分別計算 $10^2, 10^4, 10^8, 10^{16}$ 在 modulo 35 之下為 $30, 25, 30, 25$. 將 17 寫成 $17 = 2^4 + 1$, 最後得

$$10^{13} = 10^{16} \cdot 10 \equiv 25 \cdot 10 \equiv 5 \pmod{35}$$

RSA 最後的步驟就是將接到的 c 用 private key d 計算 $c^d \bmod n$. 當然了, 我們可以用上述方法計算, 不過由於接收者知道 $n = pq$, 可以分別計算 $c^d \pmod{p}$ 以及 $c^d \pmod{q}$. 為了方便起見我們假設 c^d 除以 p,q 的餘數分別為 a,b. 此時由於 p,q 互質, 故存在整數 r,s 使得 $rp + sq = 1$. 考慮 $u = brp + asq$, 此時我們有 $u \equiv a \pmod{p}$ (因 $sq \equiv 1 \pmod{p}$), 以及 $u \equiv b \pmod{q}$. 最後求出 u 除以 $n = pq$ 的餘數, 由中國剩餘定理 (Lemma 2.2) 的唯一性, 就得到 m. 我們用以下的例子簡單說明一下.

Example 2.6. 假設 $n = 35$, $c = 5$, $d = 17$. 因 $n = 5 \times 7$, 所以我們分別討論 modulo 5 和 7 的情況. 由於 $c \equiv 0 \pmod{5}$, 所以我們有 $c^{17} \equiv 0 \pmod{5}$. 又由 Lemma 2.1 知 $c^6 \equiv 1 \pmod{7}$, 因此由 $17 = 6 \times 2 + 5$, 得 $c^{17} \equiv c^5 \pmod{7}$. 計算得 $c^2 \equiv 4 \pmod{7}$ 以及 $c^4 \equiv 2 \pmod{7}$. 故得 $c^5 \equiv 3 \pmod{7}$. 要找到 m, 即 c^{17} 除以 $n = 35$ 之餘數, 我們利用中國剩餘定

理, 先找到 $5x+7y=1$ 的整數解. 例如 $x=3, y=-2$, 再考慮 $3\times(5\times 3)+0\times(7\times(-2))=45$ 除以 35 的餘數為 10, 即得 $m=10$.

2.4. **安全性.** 一般來說 asymmetric 的密碼系統其安全性在於其 decryption function 的難解. 而這個難解的問題都會是某個已知的數學難題. RSA system 的安全性就維繫在一般來說, 還沒有有效的方法將一個很大的整數分解. 底下我們將說明能由 public key n,e 解出 private key d 的困難度是和分解整數 $n=pq$ 困難度相同的. 也就是說若能將一般 RSA system 的 private key 解出, 就等同於能夠解決整數分解問題. 由於到目前為止還沒有有效的方法將一個很大的整數分解, 從這個角度來說 RSA system 是相對安全的. 不過要注意的是, 這裡是證明了由 public key 求得 private key 和整數分解是等價的, 但不是指破解 RSA 系統和整數分解是等價的. 到目前為止還無法證明沒有 private key 就無法破解 RSA 系統, 所以到目前為止還無法證明破解 RSA system 和整數分解是等價的. 從這個角度來說, RSA system 並不是絕對安全的.

我們很容易看出, 若知道 $n=pq$, 則由 public key n,e 利用前一小節的方法, 利用輾轉相除法很容易算出 d 滿足 $d\cdot e\equiv 1 \pmod{(p-1)(q-1)}$, 也因此求出 private key d. 我們真正要說明的是如何由已知 n,e 以及 d 來分解 n 得 $n=pq$. 這裡需要用到探討一個和 n 互質的數其 *order* 的問題. 由於以後談 discrete logarithm 也會談到這樣的概念, 這裡我們就先大略介紹一下.

考慮在 modulo n 的情形, 和 n 互質的整數在 modulo n 之下僅有有限多個 (事實上依定義有 $\phi(n)$ 個). 由於和 n 互質的數相乘仍和 n 互質, 所以我們知道 $\{a,a^2,a^3,\ldots,a^i,\ldots\}$ 中的元素在 modulo n 之下一定不會完全相異 (因為和 n 互質的整數在 modulo n 之下只有有限多個). 也就是說, 存在 $i>j$ 滿足 $a^i\equiv a^j \pmod n$. 又因 a^j 和 n 互質, 所以 $(a^j)^{-1}$ 在 modulo n 之下是存在著的, 因此得 $a^{i-j}\equiv 1 \pmod n$. 這說明了對任何與 n 互質的數, 必存在正整數 r 滿足 $a^r\equiv 1 \pmod n$. 最小的正整數 r 使得 $a^r\equiv 1 \pmod n$ 就稱為 a 在 modulo n 之下的 *order*, 我們用 $\mathrm{ord}_n(a)$ 來表示. 我們列出兩個和 order 有關的性質.

Lemma 2.7. 假設 $n\in\mathbb{N}$ 且 $a\in\mathbb{N}$ 滿足 $\gcd(a,n)=1$, 則我們有以下之性質

(1) $a^r\equiv 1 \pmod n$ 若且唯若 $\mathrm{ord}_n(a)\mid r$.
(2) $\mathrm{ord}_n(a^i)=\mathrm{ord}_n(a)/\gcd(i,\mathrm{ord}_n(a))$, $\forall i\in\mathbb{N}$.

事實上 Euler 的定理 Lemma 2.1 可推廣到一般的狀況 n, 它是說若 $\gcd(a,n)=1$, 則 $a^{\phi(n)}\equiv 1 \pmod n$, 因此由 Lemma 2.7 (1) 我們知 $\mathrm{ord}_n(a)\mid\phi(n)$.

假設我們知 n,e,d, 怎麼找到 p,q 滿足 $n=pq$ 呢? 這裡基本的想法是先選出 a 滿足 $\gcd(a,n)=1$. 若 $n=pq$, 則當然 $\gcd(a,p)=\gcd(a,q)=1$. 又如果恰發生 $\mathrm{ord}_p(a)\neq\mathrm{ord}_q(a)$ 的情形 (其實這情形發生的機率不小), 比方說 $\mathrm{ord}_p(a)=r<\mathrm{ord}_q(a)$. 此時我們有 $a^r\equiv 1 \pmod p$ 但 $a^r\not\equiv 1 \pmod q$, 也就是說 $p\mid a^r-1$ 但 $q\nmid a^r-1$. 因此, 此時我們有 $\gcd(a^r-1,n)=p$, 因而將 n 分解成 $n=pq$. 現在的問題是要怎樣找到這個 r 呢? 我們只好一個一個找了, 也就是說我們直接考慮 $\gcd(a^i-1,n)$, 其中 $i=1,2,3,\ldots$ 直到得到最大公因數不是 1 為止.

Example 2.8. 考慮 $a=2$, $n=35$, 此時 $\gcd(a^i-1,n)$ 當 $i=1,2,3$ 時依次為 $\gcd(1,35)=1$, $\gcd(3,35)=1$, $\gcd(7,35)=7$. 所以我們找到了 $7\mid 35$, 得 $35=7\times 5$. 事實上, 我們有 $\mathrm{ord}_7(2)=3\neq \mathrm{ord}_5(2)=4$.

上述做法要一個一個找 $\gcd(a^i-1,n)$, $i=1,2,\ldots$ 當 n 很大時計算是費時的, 幾乎無法達到. 不過若我們知道 d, 因為 $d\cdot e\equiv 1 \pmod{(p-1)(q-1)}$, 也就是 $(p-1)(q-1)\mid d\cdot e-1$. 對任意和 n 互質的 a, 當然也和 p,q 互質, 所以由 Lemma 2.1, 我們得 $a^{(p-1)(q-1)}\equiv 1 \pmod p$ 且 $a^{(p-1)(q-1)}\equiv 1 \pmod q$, 因此得 $a^{d\cdot e-1}\equiv 1 \pmod n$ (因為 $n=pq$ 且 p,q 互質). 要注意到目前為止雖然我們不知道 p,q 不過由於上面的式子中 p,q 原本都有, 所以由知道 d 之假設, $a^{d\cdot e-1}\equiv 1 \pmod n$ 當然會成立. 接下來我們將 $d\cdot e-1$ 寫成 $d\cdot e-1=r\cdot 2^s$, 其中 r 為奇數且 $s>0$ (因 $(p-1)(q-1)\mid d\cdot e-1$, 故 $d\cdot e-1$ 必為偶數).

現考慮 $\mathrm{ord}_n(a^r)$. 由於 $(a^r)^{2^s}=a^{d\cdot e-1}\equiv 1 \pmod n$, 故由 Lemma 2.7 (1) 知 $\mathrm{ord}_n(a^r)\mid 2^s$. 又 $p\mid n$ 且 $q\mid n$, 所以也由 $(a^r)^{2^s}\equiv 1 \pmod p$ 以及 $(a^r)^{2^s}\equiv 1 \pmod q$ 知 $\mathrm{ord}_p(a^r)\mid 2^s$ 以及 $\mathrm{ord}_q(a^r)\mid 2^s$. 因此我們可以考慮 $\gcd((a^r)^{2^i}-1,n)$, 其中 $i=1,2,3,\ldots s$. 依前面的論述, 只要 $\mathrm{ord}_p(a^r)\neq \mathrm{ord}_q(a^r)$, 就會有一個 i 在 $1,\ldots,s$ 之間, 使得 $\gcd((a^r)^{2^i}-1,n)=p$ 或 q, 也因此得到 n 的分解方式.

從以上說明, 我們總結當知道 private key d 後如何分解 n 的方法. 首先將 $d\cdot e-1$ 分解成 $r2^s$, 其中 r 為奇數. 接著選取一個小於 n 的正整數 a. 如果 $\gcd(a,n)\neq 1$, 則 Bingo, 因 n 是兩個奇質數的乘積, 我們有 $\gcd(a,n)=p$ 為質數, 也因此得到 n 的分解方式 (不過這機率很低). 而若 $\gcd(a,n)=1$, 此時我們考慮 $\gcd((a^r)^{2^i}-1,n)$, $1\le i\le s$. 如果 $\mathrm{ord}_p(a^r)\neq \mathrm{ord}_q(a^r)$, 則必可在之中找到 i 使得 $\gcd((a^r)^{2^i}-1,n)\neq 1$, 也因此找到了 n 的分解方式. 這裡當我們求出 a^r 後, 只要用 repeated-squaring 處理 $(a^r)^{2^i}$, 也因此所用的乘法相對的少很多. 現在遇到的問題是, 如果選到的 a 滿足 $\mathrm{ord}_p(a^r)=\mathrm{ord}_q(a^r)$ 怎麼辦? 此時對於 $1\le i\le s$ 會有 $\gcd((a^r)^{2^i}-1,n)=1$ 或 $\gcd((a^r)^{2^i}-1,n)=n$, 我們是無法得知 n 的分解方式, 所以我們必須再找另一個比 n 小且與 n 互質的數然後再照剛才程序走一次. 接下來我們要說明事實上在比 n 小且與 n 互質的數中, a 會滿足 $\mathrm{ord}_p(a^r)\neq \mathrm{ord}_q(a^r)$ 的機率是大於等於 $1/2$ 的, 所以連續找到不符合的 a 的機率最後會趨近於 0.

我們要說明當 $n=pq$ 在小於 n 且與 n 互質的 $(p-1)(q-1)$ 個數中有多於 $(p-1)(q-1)/2$ 個數 a 會滿足 $\mathrm{ord}_p(a^r)\neq \mathrm{ord}_q(a^r)$. 我們先回顧一下 *primitive root* 的定義. 當 p 是質數時, 所有小於 p 的 $p-1$ 個正整數中, 存在一個數 g 滿足 $\mathrm{ord}_p(g)=p-1$, 這等同於說 $g,g^2,\ldots,g^{p-1}$ 這 $p-1$ 個數在 modulo p 之下是相異的, 我們稱之為 modulo p 的一個 *primitive root*, 此時對任意和 n 互質的數 a, 我們都可找到 i 使得 $a\equiv g^i \pmod p$. 現假設 g_p,g_q 分別是 modulo p 和 modulo q 之下的 primitive root, 則由 Lemma 2.2, 我們可找到 g 滿足 $g\equiv g_p \pmod p$ 且 $g\equiv g_q \pmod q$. 也就是說 g 在 modulo p 和 modulo q 之下都是 primitive root. 因此所有和 n 互質的數 a, 由於 a 也和 p,q 互質, 所以存在 $x\in\{1,\ldots,p-1\}$, $y\in\{1,\ldots,q-1\}$ 滿足 $a\equiv g^x \pmod p$ 以及 $a\equiv g^y \pmod q$. 我們現在用 Lemma 2.7 (2) 來計算 a^r 在 modulo p 和 modulo q 之下的 order. 我們有

$$\mathrm{ord}_p(a^r)=\mathrm{ord}_p(g^{rx})=\mathrm{ord}_p((g^r)^x)=\frac{\mathrm{ord}_p(g^r)}{\gcd(x,\mathrm{ord}_p(g^r))}, \tag{2.2}$$

$$\mathrm{ord}_q(a^r) = \mathrm{ord}_q(g^{ry}) = \mathrm{ord}_q((g^r)^y) = \frac{\mathrm{ord}_q(g^r)}{\gcd(y, \mathrm{ord}_q(g^r))}. \tag{2.3}$$

回顧一下, 我們有 $d \cdot e - 1 = r2^s$, 其中 r 為奇數, 又 g 與 p,q 皆互質, 故由前知 $\mathrm{ord}_p(g^r) \mid 2^s$ 且 $\mathrm{ord}_q(g^r) \mid 2^s$. 又 r 是奇數, 所以知 $\gcd(r, \mathrm{ord}_p(g))$, $\gcd(r, \mathrm{ord}_q(g))$ 皆為奇數. 然而 $\mathrm{ord}_p(g) = p-1$, $\mathrm{ord}_q(g) = q-1$ 皆為偶數, 故知 $\mathrm{ord}_p(g^r) = \mathrm{ord}_p(g)/\gcd(r, \mathrm{ord}_p(g))$ 和 $\mathrm{ord}_q(g^r) = \mathrm{ord}_q(g)/\gcd(r, \mathrm{ord}_q(g))$ 皆為偶數. 因此我們可以假設 $\mathrm{ord}_p(g^r) = 2^u$, $\mathrm{ord}_q(g^r) = 2^v$, 其中 u,v 皆大於 0. 現若 $u > v$, 則當 x 為奇數時, $\gcd(x, \mathrm{ord}_p(g^r)) = \gcd(x, 2^u) = 1$, 故式子 (2.2) 告訴我們 $\mathrm{ord}_p(a^r) = 2^u$. 此時當 y 是任意整數, 式子 (2.3) 告訴我們 $\mathrm{ord}_q(a^r) \le 2^v$. 也就是說, 當我們選 x 為 $\{1, \ldots, p-1\}$ 中的奇數時, 任選 $y \in \{1, \ldots, q-1\}$, 這樣的 a 都會滿足 $\mathrm{ord}_p(a^r) > \mathrm{ord}_q(a^r)$. 因此在這種情況之下我們至少可找到 $(p-1)(q-1)/2$ 個 a 滿足 $\mathrm{ord}_p(a^r) > \mathrm{ord}_q(a^r)$. 同理當 $v > u$ 時, 我們只要選擇 y 為 $\{1, \ldots, q-1\}$ 中的奇數, 也會有 $(p-1)(q-1)/2$ 個 a 滿足 $\mathrm{ord}_q(a^r) > \mathrm{ord}_p(a^r)$.

最後我們僅要處理 $u = v$ 的情況了. 此時若選 x 為 $\{1, \ldots, p-1\}$ 中的奇數, 而 y 為 $\{1, \ldots, p-1\}$ 中的偶數, 則 $\gcd(x, \mathrm{ord}_p(g^r)) = 1$, $\gcd(y, \mathrm{ord}_p(g^r)) \ge 2$, 故得 $\mathrm{ord}_p(a^r) > \mathrm{ord}_q(a^r)$. 反之, 若選 x 為偶數, y 為奇數, 則有 $\mathrm{ord}_q(a^r) > \mathrm{ord}_p(a^r)$. 也就是說當我們分別在 $\{1, \ldots, p-1\}$, $\{1, \ldots, q-1\}$ 中選出一奇一偶的 x,y, 這樣所得的 $(p-1)(q-1)/2$ 個 a 也都會滿足 $\mathrm{ord}_p(a^r) \ne \mathrm{ord}_q(a^r)$.

綜合以上, 我們說明了: 可以求得 private key d 就等同於可以分解 n.

2.5. **注意事項.** 前一小節, 提到 RSA 系統目前來說是相對安全的. 不過這是指一般的狀況, 對於兩相異質數 p, q, 以及 public key e 的選取, 在某些特殊情況是容易被破解的. 在這一節中, 我們提出一些情況探討.

首先兩個質數 p,q 的大小不能相差太大, 因為即使 $n = pq$ 夠大, 但是若其中一個 prime 太小 n 是容易被分解的. 不過另一方面 p,q 也不能差距太小. 由於 p,q 皆為奇數, 我們假設 $p - q = 2r$, 此時令 p,q 的中間值為 a, 我們有 $a = p - r = q + r$. 因此得 $n = pq = (a+r)(a-r) = a^2 - r^2$, 即 $n + r^2 = a^2$. 所以, 如果我們考慮將 $i = 1, 2, \ldots$ 一個一個代到 $n + i^2$. 當代到 $i = r$ 時會使得 $n + r^2$ 是一個整數 a 的平方, 此時 n 就可以分解成 $(a+r)(a-r)$ 了. 注意, 雖然整數分解不易, 不過對於開方根, 卻有很有效的演算法, 因此 r 不能太小以避免用此方法破解. 另一個要避免是一些特定性質的質數, 例如若 $p-1, q-1$ 的質因數都不大, 就可能容易被破解. 又隨著近年來資訊的發達, 有的駭客會蒐集許多的 public key, 現若有兩個相異 n_1, n_2 恰有一個相同的質因數 p, 則 $\gcd(n_1, n_2) = p$. 若此情況發生, 這兩個系統就同時被破解了. 所以一般來說 p,q 要隨機在一定的範圍內選取才可以避免這些情況發生.

選擇了 p,q 後, 下一步就是要選 e 滿足 $\gcd(e, (p-1)(q-1)) = 1$. 當然了 e 越小, 當對方要將 plaintext m 加密給你時所需的計算 m^e 就越快速. 不過 e 絕不能選 2, 因為 $(p-1)(q-1)$ 會是偶數. 如果許可的話 $e = 3$ 是許多人的選擇. 不過 e 越小越有機會被人用所謂的 low-exponent attack 破解 m. 這是發生於若有一個人同時發出同一個 plaintext m 給 e 個人, 這些人的 public key 分別為 $(n_1, e), (n_2, e), \ldots, (n_e, e)$ (就是說他們都用同樣的 e). 當然了這些 n_i 若不是兩兩互質, 前面談過, 就可以將某些 n_i 分解, 以至於讓攻擊者知道 m

為何了. 而若這些 n_i 皆兩兩互質, 則攻擊者便可用中國剩餘定理解出 m 來. 這裡所用的中國剩餘定理比 Lemma 2.2 可應用於更一般的狀況, 其敘述如下.

Lemma 2.9 (Chinese Remainder Theorem)**.** 假設 $n_1,\ldots,n_e \in \mathbb{N}$ 兩兩互質. 任意給定 $a_1,\ldots,a_e \in \mathbb{Z}$, 則存在唯一的 a 滿足 $0 \le a < n_1\cdots n_e$ 且 $a \equiv a_i \pmod{n_i}$, $\forall i \in \{1,\ldots,e\}$.

我們簡單說明如何得到這個 a. 首先令 $N = n_1\cdots n_e$, 考慮 $n_i' = N/n_i$, 當 $i \ne j$ 時, 我們有 $n_i' \equiv 0 \pmod{n_j}$. 另一方面 $\gcd(n_i, n_i') = 1$, 故存在 z_i 滿足 $z_i n_i' \equiv 1 \pmod{n_i}$. 此時令 $A = \sum_{i=1}^{e} a_i z_i n_i'$ 且令 a 為 A 除以 N 的餘數即為所求. 現假設由同一個 plaintext m 分別利用 public keys $(n_1,e),\ldots,(n_e,e)$ 加密得到的 ciphertext 為 $c_1,\ldots,c_e$. 亦即 $c_1 \equiv m^e \pmod{n_1},\ldots,c_e \equiv m^e \pmod{n_e}$. 我們可以利用中國剩餘定理 Lemma 2.9 找到一個 a 滿足 $0 \le a < n_1\cdots n_e$ 且 $a \equiv c_i \pmod{n_i}$, $\forall i \in \{1,\ldots,e\}$. 注意依 RSA 系統操作方式, 這裡 m 須滿足 $0 \le m < n_i$, $\forall i \in \{1,\ldots,e\}$. 因此我們知 $0 \le m^e < n_1\cdots n_e$. 然而 m^e 也滿足 $m^e \equiv c_i \pmod{n_i}$, $\forall i \in \{1,\ldots,e\}$, 因此再由 Lemma 2.9 的唯一性, 可知 $m^e = a$. 注意這裡是真正的相等, 也就是說我們可以將所求得的 a 開 e 次方根, 所得的整數就是 m. 因為開 e 次方根可以用有效的演算法得到, 所以這個方法也有效的破解出 m. 由此可知, 在選取 e 時還是不要太小為宜.

最後要提醒的是 RSA 系統具有可乘性 (multiplicativity), 也就是說當兩個 plaintext m_1, m_2 用 public key n,e 加密後, 我們有 $c_1 \equiv m_1^e \pmod{n}$ 以及 $c_2 \equiv m_2^e \pmod{n}$. 也因此 $c_1 \times c_2 \equiv (m_1 \times m_2)^e \pmod{n}$. 也因此任何人若攔截到 c_1, c_2, 他可將 c_1, c_2 相乘, 傳輸 $c \equiv c_1 \times c_2 \pmod{n}$ 給 public key n,e 的使用者. 該使用者解密得 $m \equiv m_1 \times m_2 \pmod{n}$, 而無法得知此攔截者其實是完全不知 m 是甚麼. 這就是所謂的 *existential forgery*. 為了避免這種偽造的情形發生, 我們可以要求傳輸資料者傳輸特殊形式的 plaintext 以茲辨認. 例如在原來的 plaintext 的首尾加入相同的數字, 這樣當收受者解密得到的 plaintext 的首尾不相同時, 就知道這是偽造的. 不過要注意, plaintext 加入這樣的限制, 很可能就有人因為這樣多加的限制而有了破解的方法, 因此這樣的安全性就又要多加注意了.