Azzolini Riccardo 2021-04-15

RSA

1 RSA

RSA è lo schema di cifratura asimmetrica più utilizzato. Esso prende il nome dalle iniziali dei tre crittografi dell'MIT che l'hanno definito nel 1977: Rivest, Shamir e Adleman.

RSA si basa sull'operazione di elevamento a potenza di interi modulo numeri primi, e utilizza interi di grandi dimensioni (le dimensioni di questi interi si misurano in base al numero di bit necessario per codificarli, e attualmente si consigliano almeno 2048 bit). La sicurezza è garantita dal costo di fattorizzazione di numeri grandi.

2 Cifratura e decifratura

Siccome RSA opera sui numeri interi (per la precisione sui numeri naturali), il primo passo per cifrare un messaggio è interpretare la sequenza di bit che lo codifica come un numero $M \in \mathbb{N}$. Da qui in poi, tutte le operazioni verranno eseguite su questo intero M (e non su gruppi di bit del messaggio, come invece avveniva negli algoritmi simmetrici).

Per usare RSA si fissa un numero $n \in \mathbb{N}$ grande (ad esempio codificato in 2048 bit), ed è necessario che sia verificata la condizione

$$0 < M < n$$

Ciò significa che RSA è un algoritmo di cifratura a blocchi, con la lunghezza del blocco limitata dal valore di n. Di conseguenza, anche per RSA esiste un analogo delle modalità di funzionamento degli algoritmi simmetrici (ad esempio OAEP, Optimal Asymmetric Encryption Padding).

Oltre a n, la cifratura e la decifratura richiedono altri due parametri, tipicamente chiamati e (da "encryption", cifratura) e d (da "decryption", decifratura). Se si vuole garantire la segretezza/confidenzialità, cioè se A vuole mandare a B un messaggio M (qui si considera già l'interpretazione del messaggio come numero intero) in modo che solo B lo possa leggere, allora:

1. per cifrare il messaggio, A computa

   $$C = M^e \bmod n$$

   ottenendo così il testo cifrato C, che è anch'esso un numero intero;

2. per decifrare il messaggio, B computa

$$M = C^d \bmod n$$

Per la segretezza si cifra con la chiave pubblica del ricevente B e si decifra con la chiave privata di B, dunque:

- i parametri che il mittente A deve conoscere per la cifratura, ovvero e e n, costituiscono la chiave pubblica di B:

$$KP_B = \{e, n\}$$

- i parametri che servono al ricevente B per la decifratura, cioè d e n, costituiscono la chiave privata di B:

$$KR_B = \{d, n\}$$

Si osservi che il parametro n è presente in entrambe le chiavi, dunque l'unica informazione che deve assolutamente rimanere segreta è d. Invece, la lunghezza delle chiavi è data dal numero di bit necessari per codificare n, che come si vedrà più avanti è ciò che determina il costo degli attacchi, e quindi il livello di sicurezza fornito.

È anche possibile cifrare con la chiave privata e decifrare con la chiave pubblica,

$$C = M^d \bmod n \qquad M = C^e \bmod n$$

al fine di realizzare l'autenticazione. Detto da un altro punto di vista, le operazioni di cifratura (con e) e decifratura (con d) sono intercambiabili: se si usano delle lettere che astraggono i significati di plaintext e ciphertext, le formule per la segretezza e per l'autenticazione sono identiche,

$$\begin{array}{lll} Z = W^e \bmod n & W = Z^d \bmod n & \\ C = M^e \bmod n & M = C^d \bmod n & \text{(segretezza: } W = M \text{ e } Z = C) \\ M = C^e \bmod n & C = M^d \bmod n & \text{(autenticazione: } W = C \text{ e } Z = M) \end{array}$$

cambia solo quale dei due messaggi è considerato in chiaro e quale è considerato cifrato. Questa dualità è una caratteristica particolare di RSA, non comune a tutti gli algoritmi di cifratura asimmetrici.

Le formule che descrivono la cifratura e la decifratura sono semplici, contengono solo due operazioni, l'elevamento a potenza e il modulo, ma tali operazioni sono costose da computare su numeri grandi, dunque RSA è decisamente meno efficiente rispetto agli algoritmi simmetrici.

3 Funzionamento

Affinché la cifratura e decifratura RSA funzionino, deve esistere una relazione matematica ben precisa tra i parametri e, d e n. Adesso, tale relazione verrà presentata ragionando sulla cifratura con la chiave pubblica $\{e, n\}$ e la decifratura con la chiave privata $\{d, n\}$, cioè nel caso della segretezza, ma per la dualità osservata in precedenza lo stesso identico ragionamento vale nel caso dell'autenticazione.

Innanzitutto, si osserva che C è il risultato di un'operazione di modulo, $C = M^e \bmod n$, dunque $0 \leq C < n$ e di conseguenza $C \bmod n = C$. Allora, la formula della cifratura $C = M^e \bmod n$ può essere riscritta come

$$C \bmod n = M^e \bmod n$$

Quando due moduli danno lo stesso risultato, come in questo caso, si ha una **congruenza**: l'uguaglianza tra i moduli può essere espressa equivalentemente con la notazione

$$C \equiv M^e \pmod n \qquad \text{oppure} \qquad M^e \equiv C \pmod n$$

e si dice che C *è congruo a* M^e (e viceversa) modulo n.

Una proprietà delle congruenze è la seguente:

$$a \equiv b \pmod n \qquad \text{implica} \qquad a^k \equiv b^k \pmod n$$

Se si applica tale proprietà alla precedente congruenza scegliendo $k = d$, si ottiene

$$C^d \equiv (M^e)^d \pmod n$$

ovvero per le proprietà delle potenze

$$C^d \equiv M^{ed} \pmod n$$

e riscrivendo come uguaglianza

$$C^d \bmod n = M^{ed} \bmod n$$

Ora, applicando la formula della decifratura $M = C^d \bmod n$, quest'ultima uguaglianza diventa:

$$M = M^{ed} \bmod n$$

Poi, come già detto in precedenza, RSA impone la condizione $0 < M < n$, quindi si ha $M \bmod n = M$, e ciò permette di trasformare l'uguaglianza in una congruenza:

$$M \bmod n = M^{ed} \bmod n$$
$$M \equiv M^{ed} \pmod n$$

Si è così ricavata la condizione necessaria per il funzionamento di RSA: i parametri e, d e n devono essere tali che

$$\boxed{M \equiv M^{ed} \pmod n}$$

per ogni M nell'intervallo $0 < M < n$. Esiste un teorema che fornisce un modo per soddisfare questa congruenza, ma prima di presentarlo è necessario introdurre alcuni concetti.

4 Numeri primi, coprimi e toziente di Eulero

Un numero intero (naturale) $p \in \mathbb{N}$ è un numero **primo** se e solo se i suoi unici divisori sono 1 e p. Ogni intero $a \in \mathbb{N}$ può essere fattorizzato in numeri primi,

$$a = p_1^{a_1} \cdot p_2^{a_2} \cdots p_t^{a_t}$$

dove $p_1 < p_2 < \cdots < p_t$ sono numeri primi, che prendono il nome di *fattori primi* di a, e gli a_i sono interi positivi. Individuare i fattori primi di un numero grande non è facile, cioè tale computazione ha un costo elevato.

Due numeri $a, b \in \mathbb{N}$ sono detti **coprimi** (o *primi relativi*) se non hanno divisori comuni a parte 1, cioè se $\mathrm{MCD}(a, b) = 1$. Determinare il massimo comune divisore di due interi (e quindi verificare se essi sono coprimi) è facile se si conoscono i loro fattori primi; ad esempio:

$$\begin{aligned} 300 &= 2^2 \cdot 3^1 \cdot 5^2 \\ 18 &= 2^1 \cdot 3^2 \\ \mathrm{MCD}(300, 18) &= 2^1 \cdot 3^1 \cdot 5^0 = 6 \end{aligned}$$

Dato un numero $n \in \mathbb{N}$ positivo, la funzione **toziente di Eulero** $\phi(n)$ conta il numero di interi positivi minori di n che sono coprimi con n:

$$\phi(n) = \mathrm{card}\{x \in \{1, 2, \ldots, n-1\} \mid \mathrm{MCD}(x, n) = 1\}$$

Ad esempio:

$$\begin{aligned} \phi(3) &= \mathrm{card}\{1, 2\} = 2 \\ \phi(5) &= \mathrm{card}\{1, 2, 3, 4\} = 4 \\ \phi(6) &= \mathrm{card}\{1, 5\} = 2 \end{aligned}$$

In generale, il valore di $\phi(n)$ può essere calcolato considerando uno a uno gli interi positivi minori di n e calcolando l'MCD di ciascuno di essi per verificare quali sono coprimi con n e contarli, il che ha una complessità paragonabile alla fattorizzazione di n.[1] Ci sono però due casi particolari in cui il calcolo è facile:

- Dato un numero p primo si ha

  $$\phi(p) = p - 1$$

  perché p è coprimo con tutti i numeri compresi tra 1 e $p - 1$, in quanto non ha divisori diversi da 1 e p, quindi può avere solo il divisore 1 in comune con qualunque numero minore di p. Ad esempio:

  $$\phi(37) = \mathrm{card}\{1, 2, \ldots, 36\} = 36$$

  (si noti che anche 3 e 5 negli esempi precedenti erano primi).

[1] Esistono altri metodi di calcolo di $\phi(n)$ per un arbitrario n, ma anch'essi hanno una complessità paragonabile alla fattorizzazione di n.

- Se $n = pq$ è il prodotto di due numeri primi p e q diversi tra loro, allora

  $$\phi(n) = \phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$$

  Ad esempio:

  $$\phi(21) = \phi(3 \cdot 7) = \phi(3)\phi(7) = (3-1)(7-1) = 2 \cdot 6 = 12$$

  (e negli esempi precedenti $6 = 2 \cdot 3$ era il prodotto di due primi, dunque $\phi(6) = (2-1)(3-1) = 1 \cdot 2 = 2$).

5 Teorema di Eulero

Il **teorema di Eulero** (una generalizzazione del teorema di Fermat) afferma che, se m e n sono due interi positivi coprimi ($\text{MCD}(m, n) = 1$), allora

$$m^{\phi(n)} \bmod n = 1$$

ovvero, scritto come congruenza,

$$m^{\phi(n)} \equiv 1 \pmod n$$

Ad esempio:

- per $m = 3$ e $n = 10$, siccome $\phi(10) = \phi(2 \cdot 5) = 1 \cdot 4 = 4$ si ha che

  $$3^{\phi(10)} = 3^4 = 81 \equiv 1 \pmod{10}$$

- per $m = 2$ e $n = 11$, siccome $\phi(11) = 11 - 1 = 10$ si ha che

  $$2^{\phi(11)} = 2^{10} = 1024 \equiv 1 \pmod{11}$$

  (perché 1023 è un multiplo di 11, dunque la divisione di 1024 per 11 dà appunto resto 1).

In RSA si applica il seguente *corollario* del teorema di Eulero: dato $n = pq$, dove p e q sono due numeri primi diversi tra loro, e dato m tale che $0 < m < n$, la congruenza

$$m^{k\phi(n)+1} \equiv m \pmod n$$

vale per ogni $k \in \mathbb{N}$.

5.1 Applicazione in RSA

Come anticipato, il corollario del teorema di Fermat è lo strumento che permette di soddisfare la congruenza

$$M \equiv M^{ed} \mod n$$

necessaria per il funzionamento di RSA. Infatti, se si sceglie un n che è il prodotto di due numeri primi p e q, ovvero $n = pq$, avendo già imposto la condizione $0 < M < n$ si può applicare tale corollario, ottenendo la congruenza

$$M^{k\phi(n)+1} \equiv M \mod n \qquad \text{per ogni } k$$

da cui si deduce che la congruenza precedente è soddisfatta se si pone $ed = k\phi(n) + 1$ per un qualche k:

$$M^{ed} = M^{k\phi(n)+1} \equiv M \mod n$$

Ora si osserva che l'uguaglianza

$$ed = k\phi(n) + 1$$

significa che la divisione di ed per $\phi(n)$ dà quoziente k e resto 1, quindi, considerando in particolare il resto, si ha che

$$ed \bmod \phi(n) = 1$$

cioè, scritto come congruenza,

$$ed \equiv 1 \mod \phi(n)$$

Si conclude così che i parametri e e d devono essere **inversi moltiplicativi modulo $\boldsymbol{\phi(n)}$**, il che può essere indicato anche scrivendo:

$$d = e^{-1} \bmod \phi(n)$$

Due numeri possono essere inversi moltiplicativi modulo $\phi(n)$ solo se sono coprimi con $\phi(n)$, dunque e e d devono soddisfare le condizioni[2]

$$\text{MCD}(e, \phi(n)) = 1 \quad \text{e} \quad \text{MCD}(d, \phi(n)) = 1$$

6 Generazione delle chiavi

La generazione della coppia di chiavi KP_U e KR_U per un utente U avviene nel seguente modo:

1. Si selezionano casualmente due numeri primi grandi p e q.

[2] In realtà è sufficiente che uno dei due numeri sia coprimo con $\phi(n)$: allora, il suo inverso moltiplicativo esiste ed è garantito che sia anch'esso coprimo con $\phi(n)$.

2. Si calcolano $n = pq$ e $\phi(n) = (p-1)(q-1)$.
3. Si seleziona un numero e tale che $1 < e < \phi(n)$[3] e $\text{MCD}(e, \phi(n)) = 1$.
4. Si calcola l'inverso moltiplicativo $d = e^{-1} \bmod \phi(n)$.
5. La chiave pubblica è $KP_U = \{e, n\}$ e la chiave privata è $KR_U = \{d, n\}$.

Si osservi che in questo processo si calcola $\phi(n)$ per ricavare il parametro privato d a partire dal parametro pubblico e. Ciò è facile durante la generazione delle chiavi perché si conoscono i numeri primi p e q di cui n è il prodotto, dunque si può applicare la formula $\phi(n) = (p-1)(q-1)$. Invece, per un attaccante che ha solo la chiave pubblica, ovvero non conosce p e q (i quali sono tenuti segreti, privati), lo stesso calcolo è difficile, dato che fattorizzare n per risalire a p e q oppure computare direttamente $\phi(n)$ senza usare p e q è molto costoso.

Il valore del parametro e viene spesso scelto in modo non casuale, al fine di ottimizzare i calcoli (come si vedrà più avanti). Di conseguenza, le operazioni di cifratura e di verifica di una firma con la chiave pubblica risultano più efficiente delle operazioni di decifratura e di generazione di una firma con la chiave privata. In alternativa, per la dualità tra e e d è possibile selezionare prima d e poi computare $e = d^{-1} \bmod \phi(n)$, il che rende più efficienti le operazioni con la chiave privata rispetto a quelle con la chiave pubblica. Allora:

- quando si genera una chiave usata prevalentemente per la segretezza si seleziona e e si computa d;
- quando si genera una chiave usata prevalentemente per l'autenticazione si seleziona d e si computa e.

A uno stesso utente potrebbero poi essere associate due coppie di chiavi, una per la segretezza e una per l'autenticazione, in modo da massimizzare l'efficienza di tutte le operazioni.

6.1 Esempio

Per meglio illustrare la generazione delle chiavi è utile fare un esempio con numeri piccoli.

1. Si selezionano i numeri primi $p = 17$ e $q = 11$.
2. Si calcolano $n = pq = 17 \cdot 11 = 187$ e $\phi(n) = (p-1)(q-1) = 16 \cdot 10 = 160$.

[3] L'algoritmo RSA funziona anche se si sceglie $e > \phi(n)$, ma segue dalle proprietà dell'aritmetica modulare che per ogni valore $e > \phi(n)$ esiste un qualche valore minore di $\phi(n)$ (in particolare, il valore $e \bmod \phi(n)$) che è equivalente (ha lo stesso effetto nella cifratura e lo stesso inverso moltiplicativo d), dunque limitando la selezione agli $e < \phi(n)$ non si "perde" nulla, e anzi si rendono più efficienti i calcoli perché si evita di operare su numeri più grandi del necessario.

3. Si seleziona $e = 7$ e si verifica che $\text{MCD}(e, \phi(n)) = \text{MCD}(7, 160) = 1$ (7 è sicuramente coprimo con 160 perché è primo e non è un fattore di $160 = 2^5 \cdot 5$).
4. Si calcola $d = e^{-1} \bmod \phi(n) = 7^{-1} \bmod 160$.

   In una vera implementazione di RSA si userebbe l'*algoritmo di Euclide esteso*, con cui il calcolo di d viene eseguito insieme alla verifica che e sia coprimo con $\phi(n)$: dati e e $\phi(n)$, tale algoritmo computa il loro MCD e, se questo è 1, restituisce direttamente anche l'inverso moltiplicativo di e modulo $\phi(n)$, cioè appunto il valore di d.

   Invece, in un esempio semplice come questo si può trovare d sfruttando l'uguaglianza

   $$ed = k\phi(n) + 1$$

   che, come osservato prima, corrisponde alla congruenza $ed \equiv 1 \mod \phi(n)$. A tale scopo, si riscrive l'uguaglianza nella forma

   $$d = \frac{k\phi(n) + 1}{e} = \frac{160k + 1}{7}$$

   e si prova a inserire diversi valori di k, finché non si ottiene come risultato un valore d intero positivo.[4] In questo caso, con $k = 1$ si ottiene:

   $$d = \frac{160 \cdot 1 + 1}{7} = \frac{161}{7} = 23$$

5. La chiave pubblica è $KP = \{e, n\} = \{7, 187\}$, mentre la chiave privata è $KR = \{d, n\} = \{23, 187\}$.

Usando le chiavi così generate si possono provare anche la cifratura e la decifratura, considerando ad esempio il messaggio $M = 88$, che va bene in quanto soddisfa la condizione $0 < M < e$: $0 < 88 < 187$.

- La cifratura con la chiave pubblica avviene in questo modo:

  $$C = M^e \bmod n = 88^7 \bmod 187 = 11$$

- La decifratura con la chiave privata avviene in questo modo:

  $$M = C^d \bmod n = 11^{23} \bmod 187 = 88$$

[4] Con diversi valori di k si ottengono diversi d interi positivi, ma tutti questi d sono equivalenti modulo $\phi(n)$, quindi è sufficiente scegliere il d più piccolo (che è dato dal più piccolo k per cui si ha un risultato intero).