Crittografia simmetrica (a chiave condivisa)

Crittografia simmetrica (a chiave condivisa)

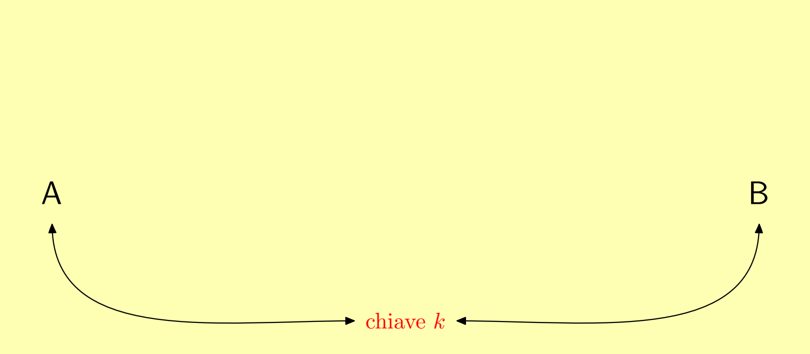
Schema di trasmissione con chiave condivisa:

Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:

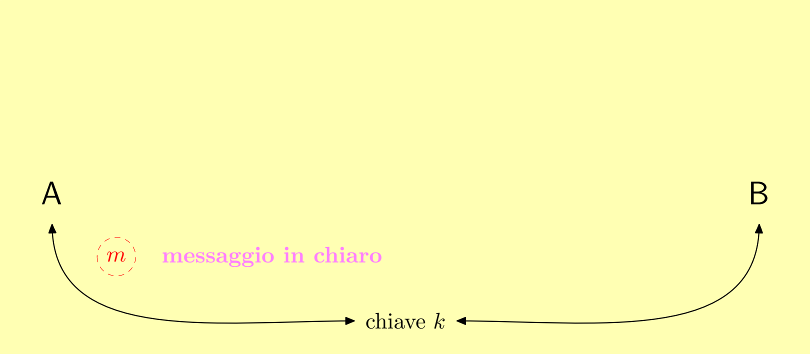
Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:



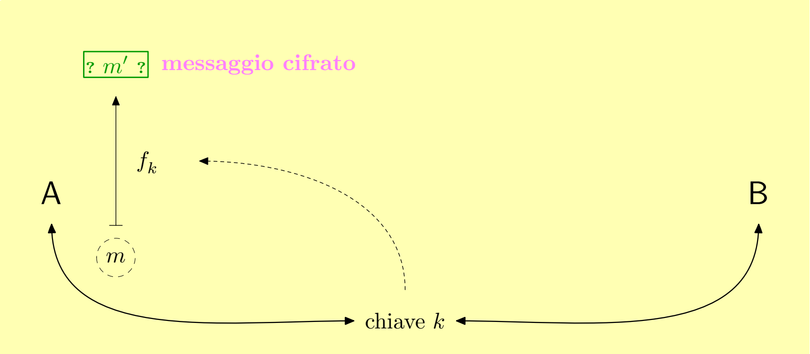
Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:



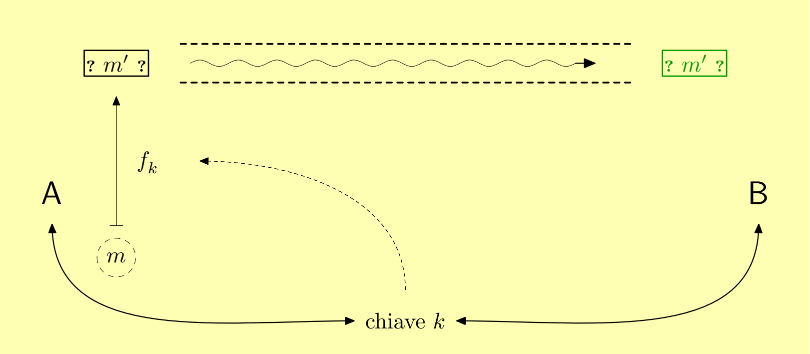
Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:



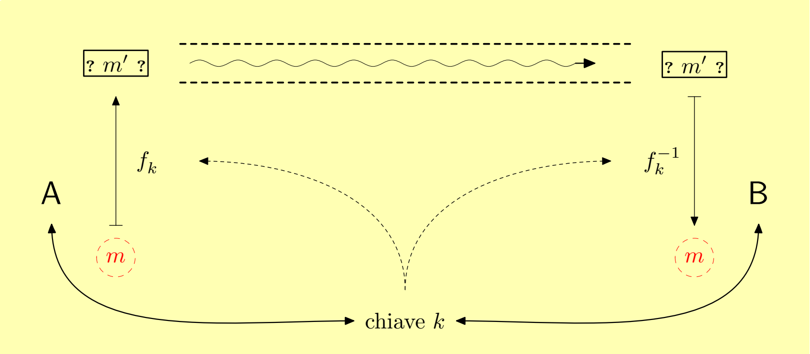
Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:



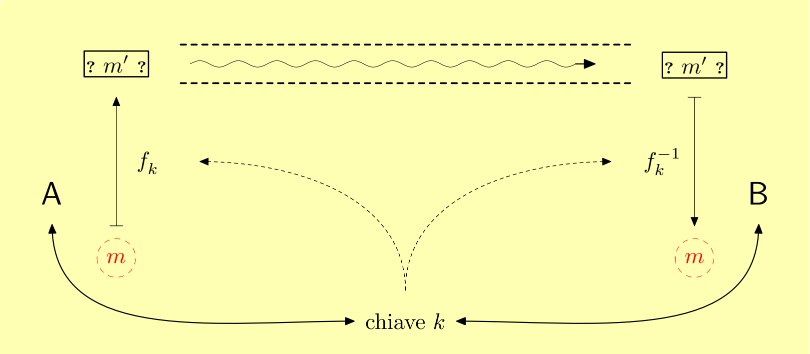
Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:



Crittografia simmetrica (a chiave condivisa)

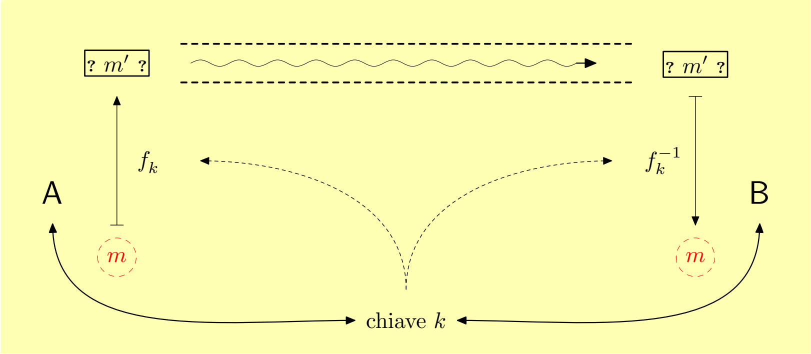
Schema di trasmissione con chiave condivisa:



Una grossa difficoltà nell'uso di questo sistema risiede nella gestione delle chiavi

Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:

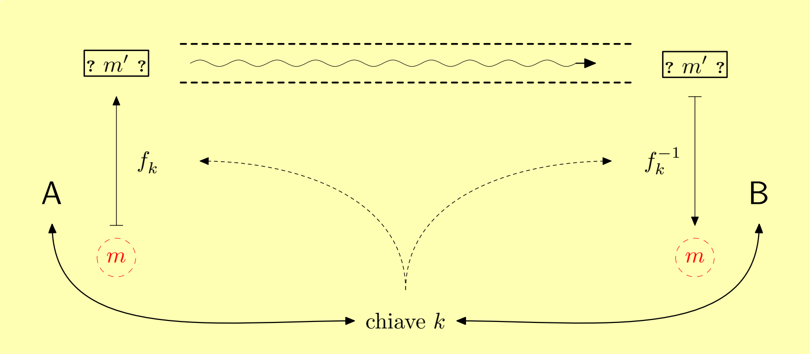


Problema:
gestione delle chiavi

- servono chiavi lunghe per garantire la sicurezza;

Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:

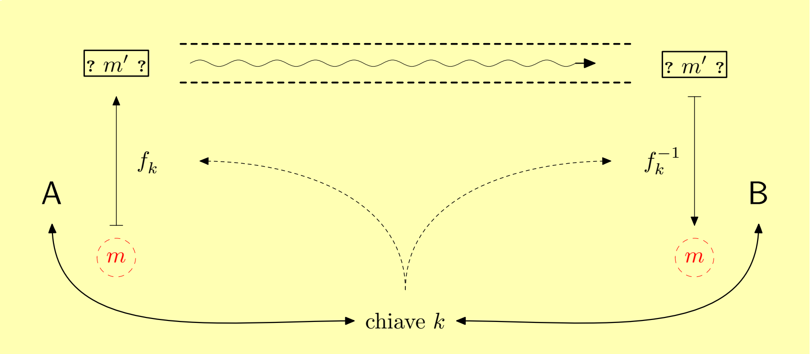


Problema:
gestione delle chiavi

- servono chiavi lunghe per garantire la sicurezza;
- lo scambio è problematico;

Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:

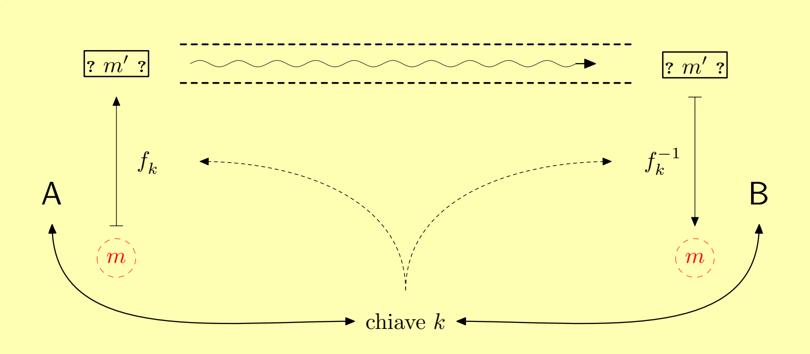


Problema:
gestione delle chiavi

- servono chiavi lunghe per garantire la sicurezza;
- lo scambio è problematico;
- ne servono tante:

Crittografia simmetrica (a chiave condivisa)

Schema di trasmissione con chiave condivisa:



Problema:
gestione delle chiavi

- servono chiavi lunghe per garantire la sicurezza;
- lo scambio è problematico;
- ne servono tante: $n(n-1)/2$ per n interlocutori.

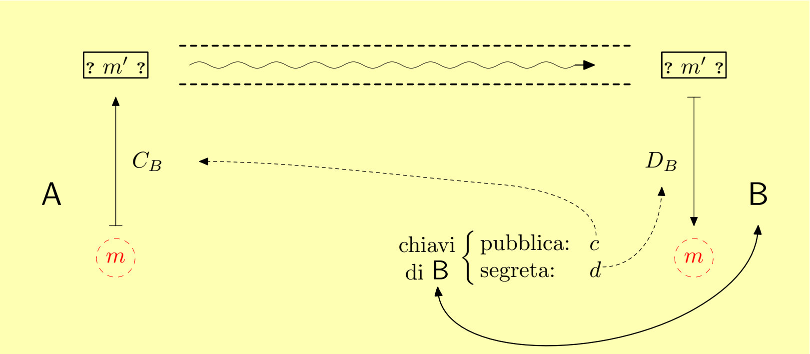
Crittografia asimmetrica (a chiave pubblica)

Crittografia asimmetrica (a chiave pubblica)

Schema di trasmissione con chiave pubblica:

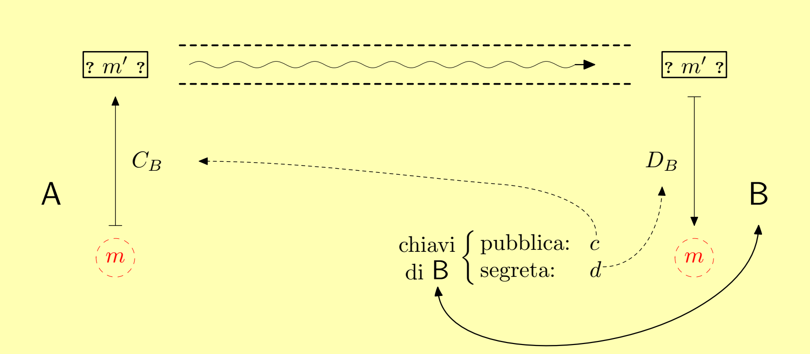
Crittografia asimmetrica (a chiave pubblica)

Schema di trasmissione con chiave pubblica:



Crittografia asimmetrica (a chiave pubblica)

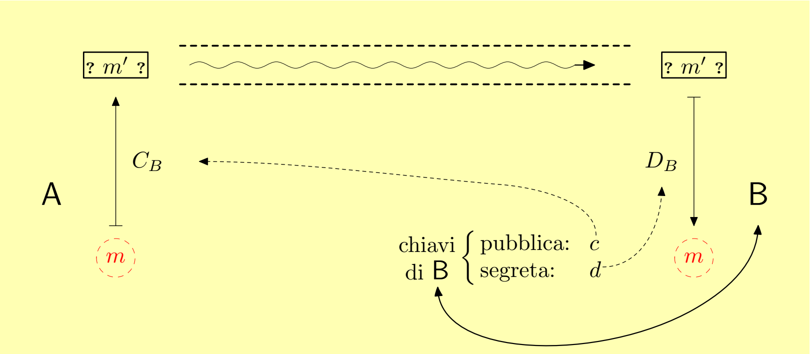
Schema di trasmissione con chiave pubblica:



Requisiti:

Crittografia asimmetrica (a chiave pubblica)

Schema di trasmissione con chiave pubblica:

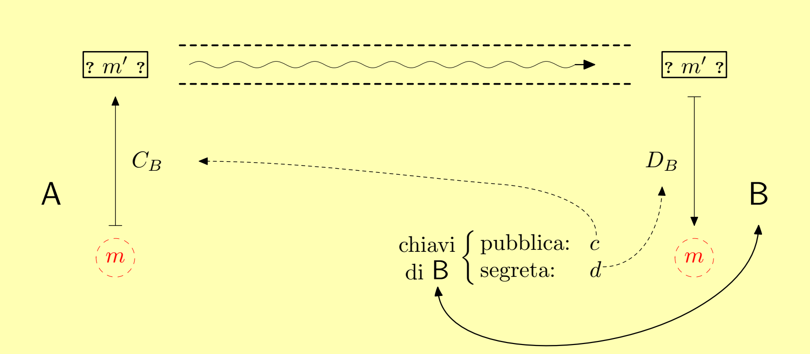


- D_B è l'inversa di C_B
  (non strettamente necessario, se non per l'autenticazione);

Requisiti:

Crittografia asimmetrica (a chiave pubblica)

Schema di trasmissione con chiave pubblica:



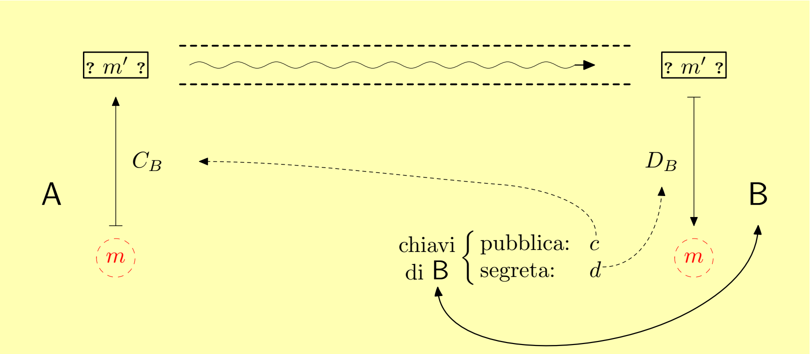
- D_B è l'inversa di C_B (non strettamente necessario, se non per l'autenticazione);

Requisiti:

- dato m, è computazionalmente facile calcolare $C_B(m)$;

Crittografia asimmetrica (a chiave pubblica)

Schema di trasmissione con chiave pubblica:



Requisiti:

- D_B è l'inversa di C_B (non strettamente necessario, se non per l'autenticazione);
- dato m, è computazionalmente facile calcolare $C_B(m)$;
- dato m', è praticamente impossibile calcolare $D_B(m')$, a meno di non conoscere d.

Aritmetica modulare

Dati gli interi a e b e l'intero positivo n, si pone $a \equiv_n b$ o anche $a \equiv b \pmod{n}$ se e solo se n divide $a - b$. Ciò equivale dire che a e b hanno lo stesso resto nella divisione per n.

Aritmetica modulare

Dati gli interi a e b e l'intero positivo n, si pone $a \equiv_n b$ o anche $a \equiv b \pmod{n}$ se e solo se n divide $a - b$. Ciò equivale dire che a e b hanno lo stesso resto nella divisione per n.

Gli interi sono così ripartiti in *n classi di resto modulo n*:

$[0]_n$, l'insieme dei multipli di n;
$[1]_n$, l'insieme degli interi che, divisi per n, hanno resto 1;
$\vdots$
$[n-1]_n$, l'insieme degli interi che, divisi per n, hanno resto $n-1$.

Aritmetica modulare

Dati gli interi a e b e l'intero positivo n, si pone $a \equiv_n b$ o anche $a \equiv b \pmod{n}$ se e solo se n divide $a - b$. Ciò equivale dire che a e b hanno lo stesso resto nella divisione per n.

Gli interi sono così ripartiti in *n classi di resto modulo n*:

$[0]_n$, l'insieme dei multipli di n;
$[1]_n$, l'insieme degli interi che, divisi per n, hanno resto 1;
$\vdots$
$[n-1]_n$, l'insieme degli interi che, divisi per n, hanno resto $n-1$.

Si ha $[i]_n = \{kn + i \mid k \in \mathbb{Z}\}$, per ogni $i \in \{0, 1, 2, \ldots, n-1\}$.

Aritmetica modulare

Dati gli interi a e b e l'intero positivo n, si pone $a \equiv_n b$ o anche $a \equiv b \pmod{n}$ se e solo se n divide $a - b$. Ciò equivale dire che a e b hanno lo stesso resto nella divisione per n.

Gli interi sono così ripartiti in *n classi di resto modulo n*:

$[0]_n$, l'insieme dei multipli di n;
$[1]_n$, l'insieme degli interi che, divisi per n, hanno resto 1;
$\vdots$
$[n-1]_n$, l'insieme degli interi che, divisi per n, hanno resto $n-1$.

Si ha $[i]_n = \{kn + i \mid k \in \mathbb{Z}\}$, per ogni $i \in \{0, 1, 2, \dots, n-1\}$.

Esempio: Se $n = 2$ le classi di resto sono: $[0]_2$, l'insieme dei numeri pari, e $[1]_2$, l'insieme dei numeri dispari.

Aritmetica modulare

Dati gli interi a e b e l'intero positivo n, si pone $a \equiv_n b$ o anche $a \equiv b \pmod{n}$ se e solo se n divide $a - b$. Ciò equivale dire che a e b hanno lo stesso resto nella divisione per n.

Gli interi sono così ripartiti in *n classi di resto modulo n*:

$[0]_n$, l'insieme dei multipli di n;
$[1]_n$, l'insieme degli interi che, divisi per n, hanno resto 1;
$\vdots$
$[n-1]_n$, l'insieme degli interi che, divisi per n, hanno resto $n-1$.

Si ha $[i]_n = \{kn + i \mid k \in \mathbb{Z}\}$, per ogni $i \in \{0, 1, 2, \ldots, n-1\}$.

Compatibilità: fissato n, per ogni $a, b, c, d \in \mathbb{Z}$ vale:

$$\begin{pmatrix} a \equiv_n b \\ c \equiv_n d \end{pmatrix} \implies \begin{pmatrix} a + c \equiv_n b + d \\ ac \equiv_n bd \end{pmatrix}$$

Ciò permette di definire le operazioni di addizione e moltiplicazione tra le classi di resto, ponendo per ogni a e b: $[a]_n + [b]_n = [a+b]_n$ e $[a]_n[b]_n = [ab]_n$. Abbiamo così una "aritmetica modulo n".

Alcuni teoremi

Alcuni teoremi

Invertibili in $\mathbb{Z}_n$

Se (e solo se) l'intero a è coprimo con n la classe $[a]_n$ è invertibile, cioè esiste un intero a' (inverso di a modulo n) tale che $aa' \equiv_n 1$.

Alcuni teoremi

Invertibili in $\mathbb{Z}_n$

Se (e solo se) l'intero a è coprimo con n la classe $[a]_n$ è invertibile, cioè esiste un intero a' (inverso di a modulo n) tale che $aa' \equiv_n 1$. Un tale a' è facile da calcolare grazie ad un algoritmo molto efficiente (l'algoritmo euclideo).

Alcuni teoremi

Invertibili in $\mathbb{Z}_n$

Se (e solo se) l'intero a è coprimo con n la classe $[a]_n$ è invertibile, cioè esiste un intero a' (inverso di a modulo n) tale che $aa' \equiv_n 1$.
Un tale a' è facile da calcolare grazie ad un algoritmo molto efficiente (l'algoritmo euclideo).

Funzione di Eulero — Teorema di Fermat-Eulero

Sia $\varphi(n)$ il numero degli interi positivi minori o uguali ad n e coprimi con n. Allora:

- $\varphi(n)$ è facile da calcolare se si conosce la fattorizzazione di n in prodotto di primi;

Alcuni teoremi

Invertibili in $\mathbb{Z}_n$

Se (e solo se) l'intero a è coprimo con n la classe $[a]_n$ è invertibile, cioè esiste un intero a' (inverso di a modulo n) tale che $aa' \equiv_n 1$.
Un tale a' è facile da calcolare grazie ad un algoritmo molto efficiente (l'algoritmo euclideo).

Funzione di Eulero — Teorema di Fermat-Eulero

Sia $\varphi(n)$ il numero degli interi positivi minori o uguali ad n e coprimi con n. Allora:

- $\varphi(n)$ è facile da calcolare se si conosce la fattorizzazione di n in prodotto di primi;
- **(TFE)** per ogni intero a coprimo con n si ha $a^{\varphi(n)} \equiv_n 1$;

Alcuni teoremi

Invertibili in $\mathbb{Z}_n$

Se (e solo se) l'intero a è coprimo con n la classe $[a]_n$ è invertibile, cioè esiste un intero a' (inverso di a modulo n) tale che $aa' \equiv_n 1$. Un tale a' è facile da calcolare grazie ad un algoritmo molto efficiente (l'algoritmo euclideo).

Funzione di Eulero — Teorema di Fermat-Eulero

Sia $\varphi(n)$ il numero degli interi positivi minori o uguali ad n e coprimi con n. Allora:

- $\varphi(n)$ è facile da calcolare se si conosce la fattorizzazione di n in prodotto di primi;
- **(TFE)** per ogni intero a coprimo con n si ha $a^{\varphi(n)} \equiv_n 1$;
- se n non è divisibile per alcun quadrato maggiore di 1 e se t è un intero tale che $t \equiv_{\varphi(n)} 1$, si ha $a^t \equiv_n a$ per ogni intero a.

RSA: scelta delle chiavi

Ogni utente sceglie le sue chiavi (pubblica e privata) in questo modo:

RSA: scelta delle chiavi

Ogni utente sceglie le sue chiavi (pubblica e privata) in questo modo:

- sceglie due primi distinti (molto grandi) p e q e ne calcola il prodotto n;

RSA: scelta delle chiavi

Ogni utente sceglie le sue chiavi (pubblica e privata) in questo modo:

- sceglie due primi distinti (molto grandi) p e q e ne calcola il prodotto n;
- sceglie un intero positivo c che sia minore di n e coprimo con $f := (p-1)(q-1)$;

RSA: scelta delle chiavi

Ogni utente sceglie le sue chiavi (pubblica e privata) in questo modo:

- sceglie due primi distinti (molto grandi) p e q e ne calcola il prodotto n;
- sceglie un intero positivo c che sia minore di n e coprimo con $f := (p-1)(q-1)$;
- calcola l'inverso d di c modulo f (dunque $cd \equiv_f 1$).

RSA: scelta delle chiavi

Ogni utente sceglie le sue chiavi (pubblica e privata) in questo modo:

- sceglie due primi distinti (molto grandi) p e q e ne calcola il prodotto n;
- sceglie un intero positivo c che sia minore di n e coprimo con $f := (p-1)(q-1)$;
- calcola l'inverso d di c modulo f (dunque $cd \equiv_f 1$).
- chiavi: $\begin{cases} (n, c) & \text{pubblica} \\ d & \text{privata} \end{cases}$

RSA: scelta delle chiavi

Ogni utente sceglie le sue chiavi (pubblica e privata) in questo modo:

- sceglie due primi distinti (molto grandi) p e q e ne calcola il prodotto n;
- sceglie un intero positivo c che sia minore di n e coprimo con $f := (p-1)(q-1)$;
- calcola l'inverso d di c modulo f (dunque $cd \equiv_f 1$).
- chiavi: $\begin{cases} (n, c) & \text{pubblica} \\ d & \text{privata} \end{cases}$

N.B. Si ha: $f = \varphi(n)$.

RSA: come funziona

$$n = pq, \qquad f = \varphi(n) = (p-1)(q-1), \qquad cd \equiv_f 1.$$

Chiavi: (n, c) pubblica; d privata.

RSA: come funziona

$$n = pq, \qquad f = \varphi(n) = (p-1)(q-1), \qquad cd \equiv_f 1.$$

Chiavi: (n, c) pubblica; d privata.

Cifratura: **A** intende mandare un messaggio a **B**. Allora **A** prende atto della chiave pubblica (n, c) di **B**, codifica in un qualsiasi modo il messaggio con un numero intero m tale che $0 < m < n$ (se ciò non è possibile perché il messaggio è troppo lungo, **A** suddivide preliminarmente quest'ultimo in blocchi) e calcola il resto di m^c modulo n (esistono metodi molto rapidi per farlo). Il messaggio cifrato sarà appunto questo resto, m'.

RSA: come funziona

$n = pq, \qquad f = \varphi(n) = (p-1)(q-1), \qquad cd \equiv_f 1.$
Chiavi: (n, c) pubblica; d privata.

Cifratura: **A** intende mandare un messaggio a **B**. Allora **A** prende atto della chiave pubblica (n, c) di **B**, codifica in un qualsiasi modo il messaggio con un numero intero m tale che $0 < m < n$ (se ciò non è possibile perché il messaggio è troppo lungo, **A** suddivide preliminarmente quest'ultimo in blocchi) e calcola il resto di m^c modulo n (esistono metodi molto rapidi per farlo). Il messaggio cifrato sarà appunto questo resto, m'.

Decifrazione: **B** riceve m' e, usando la sua chiave privata d, calcola il resto di $(m')^d$ modulo n. Ottiene così m, cioè il messaggio originario.

RSA: perché funziona

$$n = pq, \qquad f = \varphi(n) = (p-1)(q-1), \qquad cd \equiv_f 1.$$

Chiavi: (n, c) pubblica; d privata.

Cifratura: $m \mapsto m' = m^c \bmod n$

Decifrazione: $m' \mapsto m = (m')^d \bmod n$.

RSA: perché funziona

$n = pq, \qquad f = \varphi(n) = (p-1)(q-1), \qquad cd \equiv_f 1.$
Chiavi: (n, c) pubblica; d privata.
Cifratura: $m \mapsto m' = m^c \bmod n$
Decifrazione: $m' \mapsto m = (m')^d \bmod n$.

Sia m_1 il resto di $(m')^d$ modulo n. Abbiamo $m' \equiv_n m^c$ e quindi $m_1 \equiv_n (m')^d \equiv_n m^{cd}$. Inoltre $f = \varphi(n)$ e $cd \equiv_f 1$, dunque $m^{cd} \equiv_n m$. Allora si ha $0 \leq m_1, m < n$ e $m_1 \equiv_n m$; da ciò segue $m_1 = m$.

RSA: perché funziona

$$n = pq, \qquad f = \varphi(n) = (p-1)(q-1), \qquad cd \equiv_f 1.$$

Chiavi: (n, c) pubblica; d privata.
Cifratura: $m \mapsto m' = m^c \bmod n$
Decifrazione: $m' \mapsto m = (m')^d \bmod n$.

Sia m_1 il resto di $(m')^d$ modulo n. Abbiamo $m' \equiv_n m^c$ e quindi $m_1 \equiv_n (m')^d \equiv_n m^{cd}$. Inoltre $f = \varphi(n)$ e $cd \equiv_f 1$, dunque $m^{cd} \equiv_n m$. Allora si ha $0 \leq m_1, m < n$ e $m_1 \equiv_n m$; da ciò segue $m_1 = m$.

E lo spione?

RSA: perché funziona

$n = pq, \qquad f = \varphi(n) = (p-1)(q-1), \qquad cd \equiv_f 1.$
Chiavi: (n, c) pubblica; d privata.
Cifratura: $m \mapsto m' = m^c \bmod n$
Decifrazione: $m' \mapsto m = (m')^d \bmod n$.

Se una persona diversa da **B** sapesse fattorizzare n, allora saprebbe anche calcolare $f = (p-1)(q-1)$ e quindi ricavare d da (n, c) e decifrare il messaggio. Il punto è che, se i primi p e q sono scelti bene, fattorizzare n è (meglio: si ritiene che sia) estremamente complesso, a meno di non utilizzare *computer quantistici*. Si può anche (viceversa) dimostrare che il problema di ricavare d da (n, c) ha lo stesso grado di complessità di quello di fattorizzare n, quindi altissimo. Ciò non esclude che esista qualche metodo per scardinare RSA che prescinda dalla fattorizzazione di n, anzi è stato dimostrato che se esiste un metodo "molto efficiente" ne esiste anche uno di questo tipo. Al momento (forse, speriamo) non sono noti tali metodi.

RSA: perché funziona

$n = pq, \qquad f = \varphi(n) = (p-1)(q-1), \qquad cd \equiv_f 1.$
Chiavi: (n, c) pubblica; d privata.
Cifratura: $m \mapsto m' = m^c \bmod n$
Decifrazione: $m' \mapsto m = (m')^d \bmod n.$

RSA fornisce anche un metodo sicuro di autenticazione.

Un esempio di cifratura RSA

TESTO: “Matematica per la crittografia”.

Un esempio di cifratura RSA

TESTO: “Matematica per la crittografia”.

Scelta delle chiavi: (p e q sono primi; $cd \equiv_f 1$)

$p = 25893247$	$c = 41794313$
$q = 34747121$	$d = 43054457$
$n = pq = 899715786591887$	chiave pubblica: (n, c)
$f = (p-1)(q-1) = 899715725951520$	chiave privata: d

Un esempio di cifratura RSA

TESTO: “Matematica per la crittografia”.

Scelta delle chiavi: (p e q sono primi; $cd \equiv_f 1$)

$p = 25893247$	$c = 41794313$
$q = 34747121$	$d = 43054457$
$n = pq = 899715786591887$	chiave pubblica: (n, c)
$f = (p-1)(q-1) = 899715725951520$	chiave privata: d

N.B. i numeri qui utilizzati sono troppo piccoli perché la cifratura sia sicura. Inoltre, il metodo usato per codificare le stringhe di caratteri in interi, anche se comodo, è molto poco efficiente.

Un esempio di cifratura RSA

TESTO: “Matematica per la crittografia”.

Scelta delle chiavi: (p e q sono primi; $cd \equiv_f 1$)

$p = 25893247$	$c = 41794313$
$q = 34747121$	$d = 43054457$
$n = pq = 899715786591887$	chiave pubblica: (n, c)
$f = (p-1)(q-1) = 899715725951520$	chiave privata: d

Codifica del testo: ogni lettera minuscola viene trasformata in maiuscola, ed ogni carattere viene sostituito dal suo codice ASCII (due cifre decimali); ad una stringa di caratteri corrisponde il numero ottenuto concatenando da questi codici:

…		…	,	-	.	…	A	B	C	…	Z
…	32	…	44	45	46	…	65	66	67	…	90

quindi, ad esempio, “a B. Zac” si codifica come 6532664632906567.

CHIAVI: $(n, c) = (899715786591887, 41794313)$ $d = 43054457$

CHIAVI: $(n, c) = (899715786591887, 41794313)$ $d = 43054457$

Il protocollo RSA permette di cifrare in un unico passaggio le stringhe codificate da un un intero minore di n, quindi, col sistema ora definito, certamente quelle di al più sette caratteri. Dunque, dividiamo la stringa "Matematica per la crittografia" in blocchi di lunghezza al più 7, e codifichiamo questi cominciando dal primo:

M	A	T	E	M	A	T
77	65	84	69	77	65	84

e similmente per gli altri:

"MATEMAT"	"ICA PER"	" LA CRI"	"TTOGRAF"	"IA"
77658469776584	73676532806982	32766532678273	84847971826570	7365

Il protocollo RSA permette di cifrare in un unico passaggio le stringhe codificate da un un intero minore di n, quindi, col sistema ora definito, certamente quelle di al più sette caratteri. Dunque, dividiamo la stringa "Matematica per la crittografia" in blocchi di lunghezza al più 7, e codifichiamo questi cominciando dal primo:

M	A	T	E	M	A	T
77	65	84	69	77	65	84

e similmente per gli altri:

"MATEMAT"	"ICA PER"	" LA CRI"	"TTOGRAF"	"IA"
77658469776584	73676532806982	32766532678273	84847971826570	7365

Ora:

$$77658469776584^c \equiv_n 348775283430137; \quad 73676532806982^c \equiv_n 406106154987544$$
$$32766532678273^c \equiv_n 727567161267633; \quad 84847971826570^c \equiv_n 704911937371759$$
$$7365^c \equiv_n 285112597092454$$

Quindi "Matematica per la crittografia" risulta cifrato come

CHIAVI: $(n, c) = (899715786591887, 41794313)$ $d = 43054457$

Il protocollo RSA permette di cifrare in un unico passaggio le stringhe codificate da un un intero minore di n, quindi, col sistema ora definito, certamente quelle di al più sette caratteri. Dunque, dividiamo la stringa "Matematica per la crittografia" in blocchi di lunghezza al più 7, e codifichiamo questi cominciando dal primo:

M	A	T	E	M	A	T
77	65	84	69	77	65	84

e similmente per gli altri:

"MATEMAT"	"ICA PER"	" LA CRI"	"TTOGRAF"	"IA"
77658469776584	73676532806982	32766532678273	84847971826570	7365

Ora:

$$77658469776584^c \equiv_n 348775283430137; \quad 73676532806982^c \equiv_n 406106154987544$$
$$32766532678273^c \equiv_n 727567161267633; \quad 84847971826570^c \equiv_n 704911937371759$$
$$7365^c \equiv_n 285112597092454$$

Quindi "Matematica per la crittografia" risulta cifrato come

348775283430137 406106154987544 727567161267633 704911937371759 285112597092454.

Logaritmo discreto

Problema (PLD)

Dati gli interi a, b, n (con $n > 0$), nell'ipotesi che $b \equiv_n a^l$ per un opportuno intero l, si trovi un tale l.

Logaritmo discreto

Problema (PLD)

Dati gli interi a, b, n (con $n > 0$), nell'ipotesi che $b \equiv_n a^l$ per un opportuno intero l, si trovi un tale l.

Il PLD si può porre in ambienti diversi (gruppi, in genere) ed è spesso di grande complessità computazionale. Si possono quindi costruire funzioni "a senso unico" del tipo $t \mapsto a^t$.
Queste funzioni esponenziali vengono usate in diversi protocolli crittografici. È utile il fatto che, per ogni fissata base a, esse commutano tra loro.

Logaritmo discreto

Problema (PLD)

Dati gli interi a, b, n (con $n > 0$), nell'ipotesi che $b \equiv_n a^l$ per un opportuno intero l, si trovi un tale l.

Il PLD si può porre in ambienti diversi (gruppi, in genere) ed è spesso di grande complessità computazionale. Si possono quindi costruire funzioni "a senso unico" del tipo $t \mapsto a^t$.

Queste funzioni esponenziali vengono usate in diversi protocolli crittografici. È utile il fatto che, per ogni fissata base a, esse commutano tra loro.

Esempio: tavola dei logaritmi in base 10 modulo 4567:

1	0	6	3731	11	196	16	2532	21	1107
2	2916	7	292	12	2081	17	213	22	3112
3	815	8	4182	13	552	18	4546	23	2659
4	1266	9	1630	14	3208	19	2143	24	431
5	1651	10	1	15	2466	20	2917	25	3302

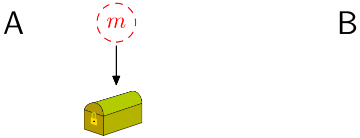
…

Crittografia senza chiavi condivise

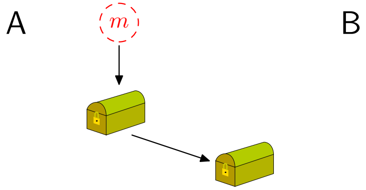
Crittografia senza chiavi condivise

A m B

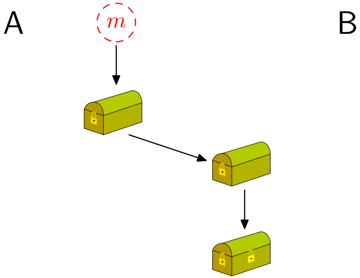
Crittografia senza chiavi condivise



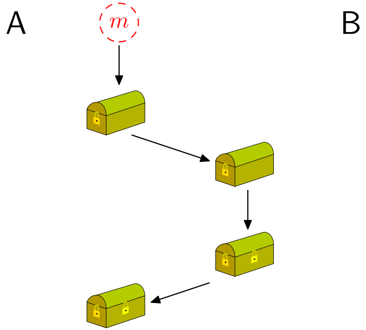
Crittografia senza chiavi condivise



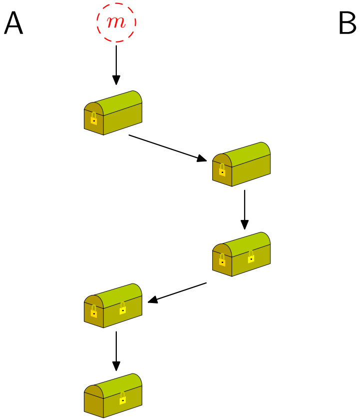
Crittografia senza chiavi condivise



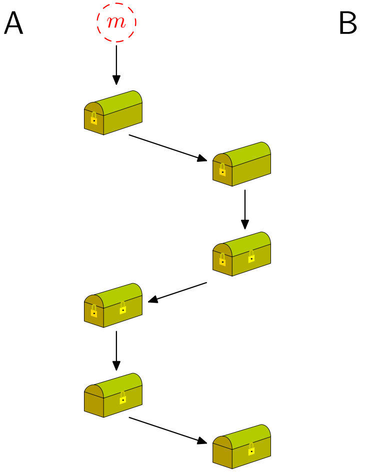
Crittografia senza chiavi condivise



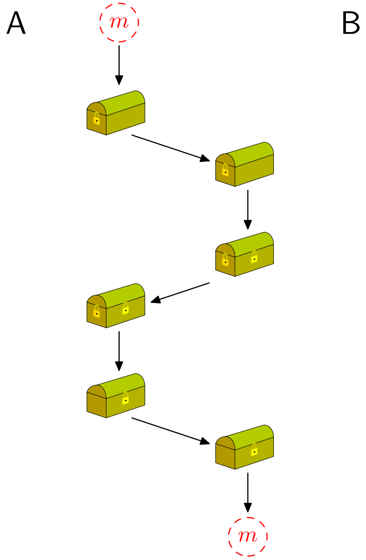
Crittografia senza chiavi condivise



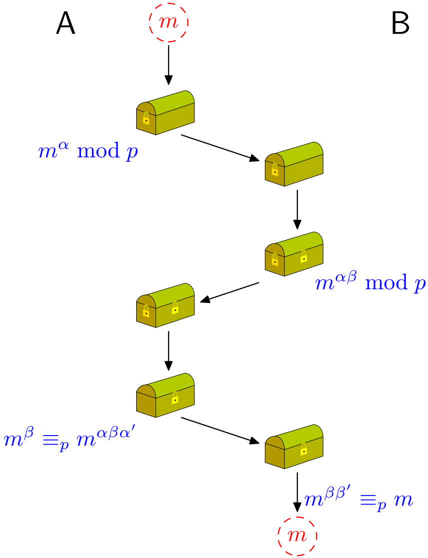
Crittografia senza chiavi condivise



Crittografia senza chiavi condivise



Crittografia senza chiavi condivise



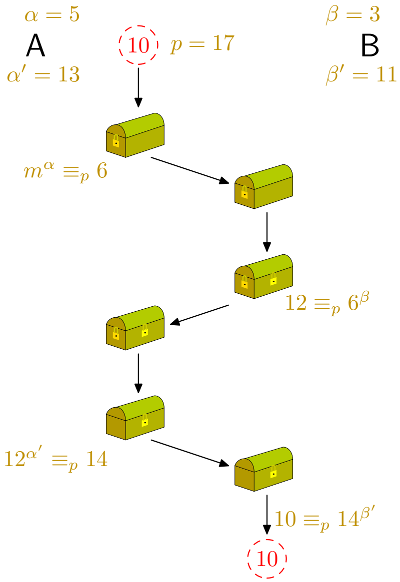
A e **B** fissano un primo p e scelgono un intero segreto ciascuno: α e β, entrambi coprimi con $p-1$.
A può calcolare α' tale che

$$\alpha\alpha' \equiv_{p-1} 1.$$

B può calcolare β' tale che

$$\beta\beta' \equiv_{p-1} 1.$$

Crittografia senza chiavi condivise



A e **B** fissano un primo p e scelgono un intero segreto ciascuno: α e β, entrambi coprimi con $p - 1$.
A può calcolare α' tale che

$$\alpha\alpha' \equiv_{p-1} 1.$$

B può calcolare β' tale che

$$\beta\beta' \equiv_{p-1} 1.$$

Protocollo Diffie-Hellman(-Merkle)

come **A** e **B** si possono accordare su una chiave senza trasmetterla:

Protocollo Diffie-Hellman(-Merkle)

come **A** e **B** si possono accordare su una chiave senza trasmetterla:

- **A** fissa un primo p, ed un opportuno intero a tale che $0 < a < p$, ed un intero (segreto) α, e trasmette a **B**: p, a e $a_1 := a^\alpha \bmod p$;
  Esempio: **A**: $[\alpha = 5]$ $\boxed{p = 13,\ a = 7,\ a_1 = 11}$ $\rightarrow$ **B**

Protocollo Diffie-Hellman(-Merkle)

come **A** e **B** si possono accordare su una chiave senza trasmetterla:

- **A** fissa un primo p, ed un opportuno intero a tale che $0 < a < p$, ed un intero (segreto) α, e trasmette a **B**: p, a e $a_1 := a^\alpha \bmod p$;
  Esempio: **A**: $[\alpha = 5]$ $\boxed{p = 13, a = 7, a_1 = 11}$ $\rightarrow$ **B**
- **B** fissa un intero segreto β e trasmette ad **A** $a_2 := a^\beta \bmod p$;
  B: $[\beta = 8]$ $\boxed{a_2 = 3}$ $\rightarrow$ **A**

Protocollo Diffie-Hellman(-Merkle)

come **A** e **B** si possono accordare su una chiave senza trasmetterla:

- **A** fissa un primo p, ed un opportuno intero a tale che $0 < a < p$, ed un intero (segreto) α, e trasmette a **B**: p, a e $a_1 := a^\alpha \bmod p$;
  Esempio: **A**: $[\alpha = 5]$ $\boxed{p = 13,\ a = 7,\ a_1 = 11}$ $\rightarrow$ **B**
- **B** fissa un intero segreto β e trasmette ad **A** $a_2 := a^\beta \bmod p$;
  B: $[\beta = 8]$ $\boxed{a_2 = 3}$ $\rightarrow$ **A**
- a questo punto **A** e **B** possono calcolare la loro chiave comune k: il resto di $a_2^\alpha \equiv_p a_1^\beta \equiv_p a^{\alpha\beta}$ modulo p.
  A: $3^5 \equiv_{13} 9$; **B**: $11^8 \equiv_{13} 9$

Protocollo Diffie-Hellman(-Merkle)

come **A** e **B** si possono accordare su una chiave senza trasmetterla:

- **A** fissa un primo p, ed un opportuno intero a tale che $0 < a < p$, ed un intero (segreto) α, e trasmette a **B**: p, a e $a_1 := a^\alpha \bmod p$;
  Esempio: **A**: $[\alpha = 5]$ $\boxed{p = 13, a = 7, a_1 = 11}$ $\rightarrow$ **B**
- **B** fissa un intero segreto β e trasmette ad **A** $a_2 := a^\beta \bmod p$;
  B: $[\beta = 8]$ $\boxed{a_2 = 3}$ $\rightarrow$ **A**
- a questo punto **A** e **B** possono calcolare la loro chiave comune k: il resto di $a_2^\alpha \equiv_p a_1^\beta \equiv_p a^{\alpha\beta}$ modulo p.
  A: $3^5 \equiv_{13} 9$; **B**: $11^8 \equiv_{13} 9$

La segretezza della chiave è garantita dal fatto che, anche se p, a, $a_1 \equiv_p a^\alpha$ e $a_2 \equiv_p a^\beta$ sono noti (al solito spione), egli non ha a disposizione metodi per calcolare, ad esempio, α e quindi a_2^α (problema del logaritmo discreto)

Sistema crittografico ElGamal

Sistema crittografico ElGamal

1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e privata) in questo modo: viene scelto un “ambiente di calcolo” G (un gruppo), un elemento a di G, in modo che il PLD sia “molto complesso” per le potenze di a in G. Ciascun utente sceglie poi un intero d come chiave privata e, come chiave pubblica (G, a, a^d).

Sistema crittografico ElGamal

1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e privata) in questo modo: viene scelto un "ambiente di calcolo" G (un gruppo), un elemento a di G, in modo che il PLD sia "molto complesso" per le potenze di a in G. Ciascun utente sceglie poi un intero d come chiave privata e, come chiave pubblica (G, a, a^d).

2. Cifratura: **A** intende mandare un messaggio a **B**. Allora **A** prende atto della chiave pubblica (G, a, c) di **B**, codifica in un qualsiasi modo il messaggio come un elemento di G, sceglie un intero α ed invia a **B** il messaggio cifrato come $\left(a^\alpha, mc^\alpha\right)$.

Sistema crittografico ElGamal

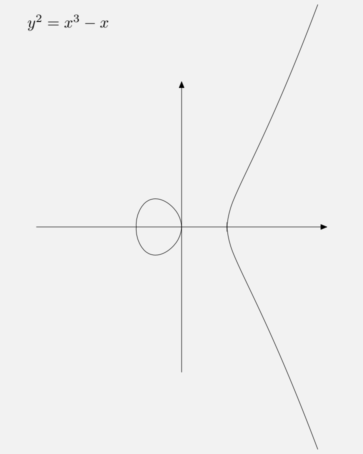
1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e privata) in questo modo: viene scelto un "ambiente di calcolo" G (un gruppo), un elemento a di G, in modo che il PLD sia "molto complesso" per le potenze di a in G. Ciascun utente sceglie poi un intero d come chiave privata e, come chiave pubblica (G, a, a^d).

2. Cifratura: **A** intende mandare un messaggio a **B**. Allora **A** prende atto della chiave pubblica (G, a, c) di **B**, codifica in un qualsiasi modo il messaggio come un elemento di G, sceglie un intero α ed invia a **B** il messaggio cifrato come $\left(a^{\alpha}, mc^{\alpha}\right)$.

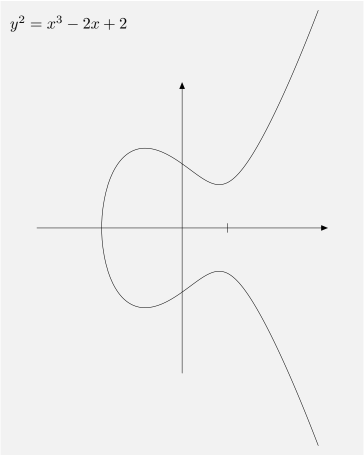
3. Decifrazione: **B** riceve $\left(a^{\alpha}, mc^{\alpha}\right)$, ovvero $\left(a^{\alpha}, ma^{\alpha d}\right)$, e, usando la sua chiave privata d, da a^{α} calcola $a^{\alpha d}$; di questo è facile calcolare l'inverso $\left(a^{\alpha d}\right)^{-1}$, quindi **B** può calcolare $m = \left(ma^{\alpha d}\right)\left(a^{\alpha d}\right)^{-1}$.

Curve ellittiche

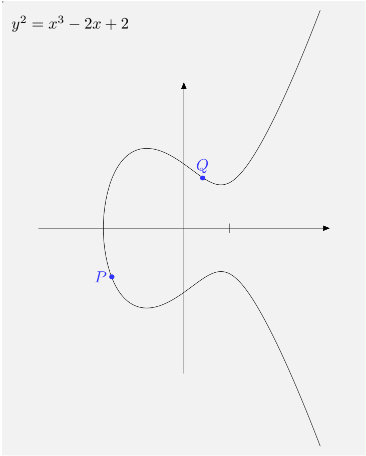
Curve ellittiche



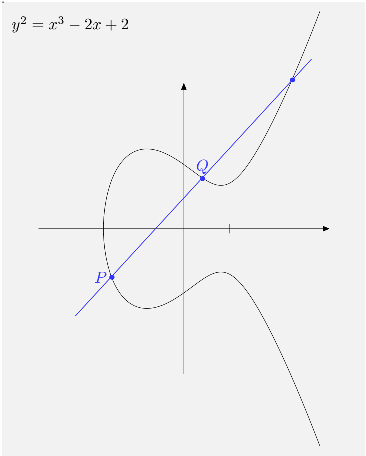
Curve ellittiche



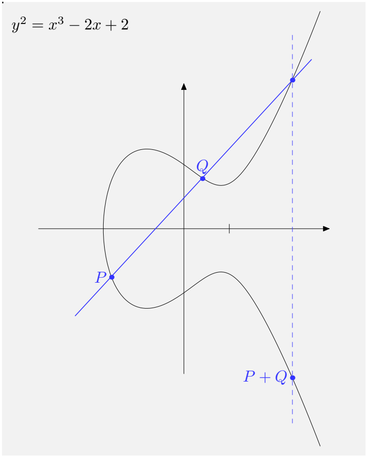
Curve ellittiche



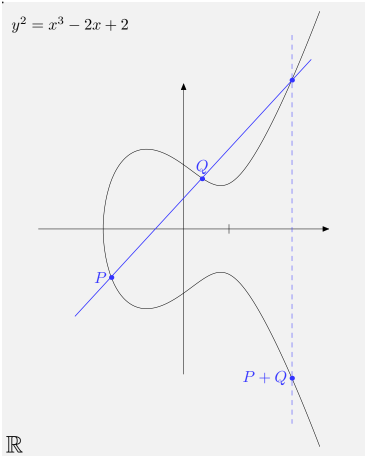
Curve ellittiche

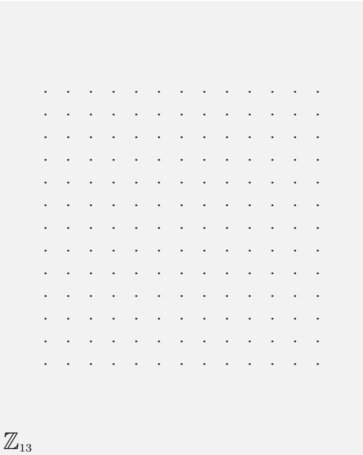


Curve ellittiche

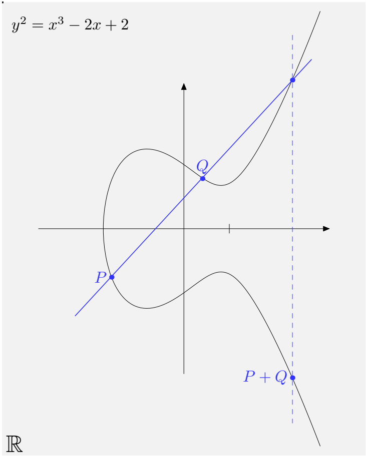


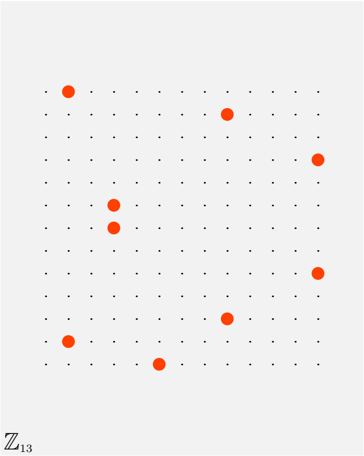
Curve ellittiche



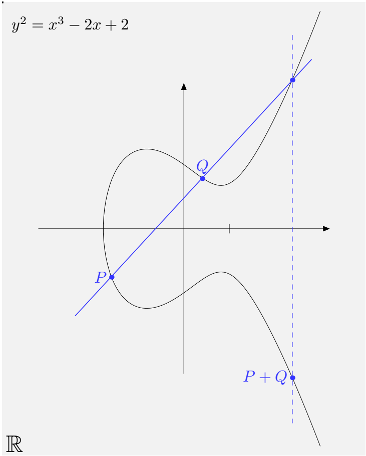


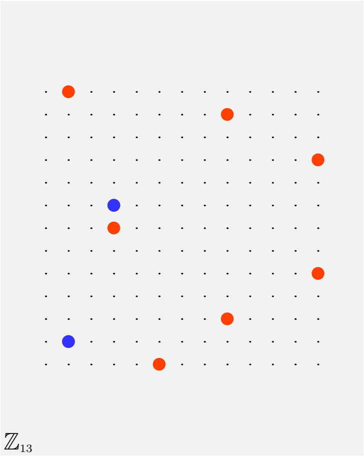
Curve ellittiche





Curve ellittiche





Curve ellittiche