348775283430137 406106154987544
727567161267633 704911937371759
285112597092454

348775283430137 406106154987544 727567161267633 704911937371759 285112597092454

ovvero:

Matematica per la crittografia

Gli scopi principali della crittografia sono quelli di garantire la possibilità di:

Gli scopi principali della crittografia sono quelli di garantire la possibilità di:

- trasmissione criptata di messaggi anche attraverso canali non sicuri
  (problema semplificato: deposito di dati per uso personale, cioè:
  'messaggi a se stesso')

Gli scopi principali della crittografia sono quelli di garantire la possibilità di:

- trasmissione criptata di messaggi anche attraverso canali non sicuri
  (problema semplificato: deposito di dati per uso personale, cioè:
  'messaggi a se stesso')

- autenticazione:

  ⋆ del mittente (firma digitale)
  ⋆ del testo (che non sia stato manomesso da interventi esterni)

Applicazioni quotidiane:

Oltre ad avere applicazioni non sempre gradevoli (militari, eccesso di difesa della proprietà intellettuale, crimine organizzato) tecniche crittografiche sono quotidianamente utilizzate, ad esempio, per:

Applicazioni quotidiane:

Oltre ad avere applicazioni non sempre gradevoli (militari, eccesso di difesa della proprietà intellettuale, crimine organizzato) tecniche crittografiche sono quotidianamente utilizzate, ad esempio, per:

- piccole transazioni finanziarie (bancomat, carte di credito, acquisti on-line . . .);

Applicazioni quotidiane:

Oltre ad avere applicazioni non sempre gradevoli (militari, eccesso di difesa della proprietà intellettuale, crimine organizzato) tecniche crittografiche sono quotidianamente utilizzate, ad esempio, per:

- piccole transazioni finanziarie (bancomat, carte di credito, acquisti on-line . . .);
- protezione dei sistemi informatici (passwords etc.) e delle comunicazioni in rete.

Sono correntemente in uso due principali schemi di trasmissione criptata di messaggi:

Sono correntemente in uso due principali schemi di trasmissione criptata di messaggi:

- a chiave privata (o *simmetrica*, crittografia classica)

Sono correntemente in uso due principali schemi di trasmissione criptata di messaggi:

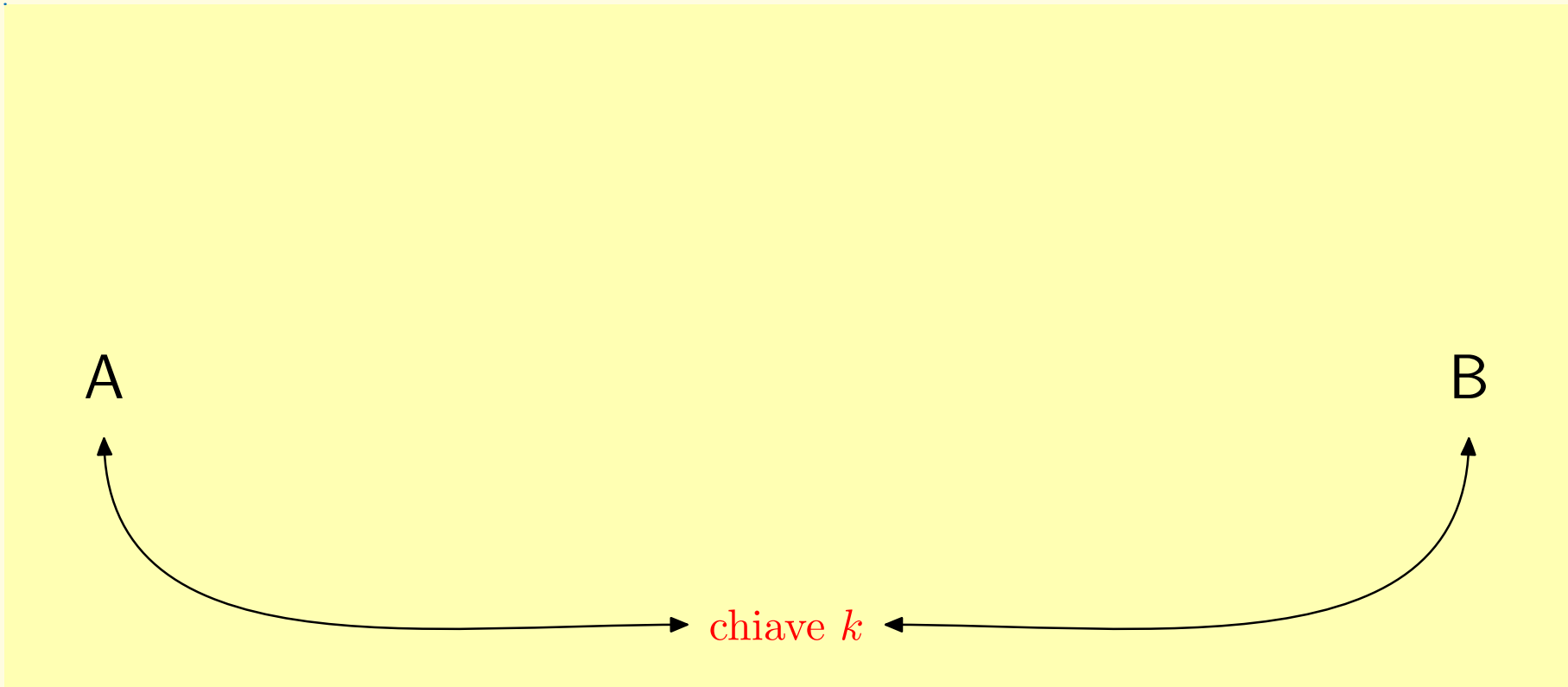
- a chiave privata (o *simmetrica*, crittografia classica)
- a chiave pubblica (o *asimmetrica*)

(in realtà, nella pratica vengono quasi sempre usati metodi ibridi)

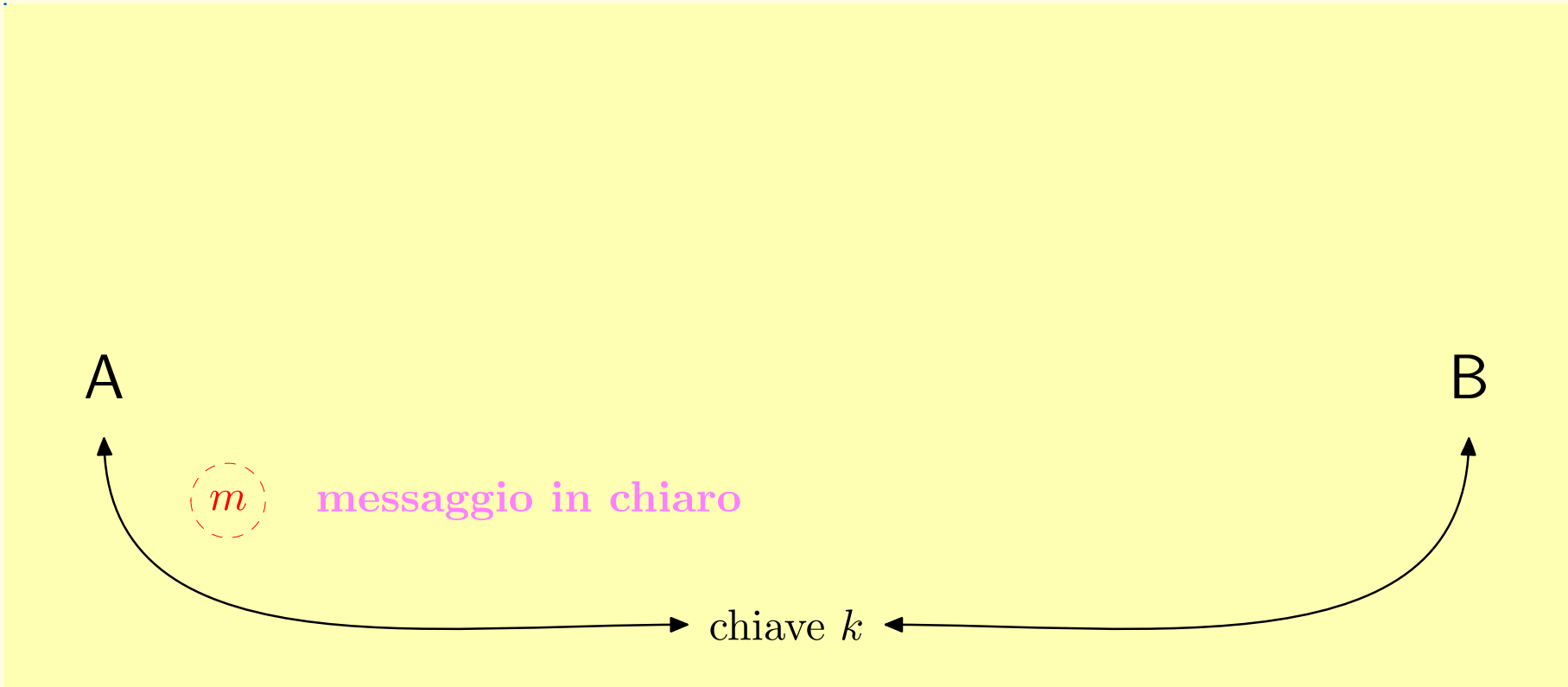
Schema di trasmissione a chiave privata:

Schema di trasmissione a chiave privata:

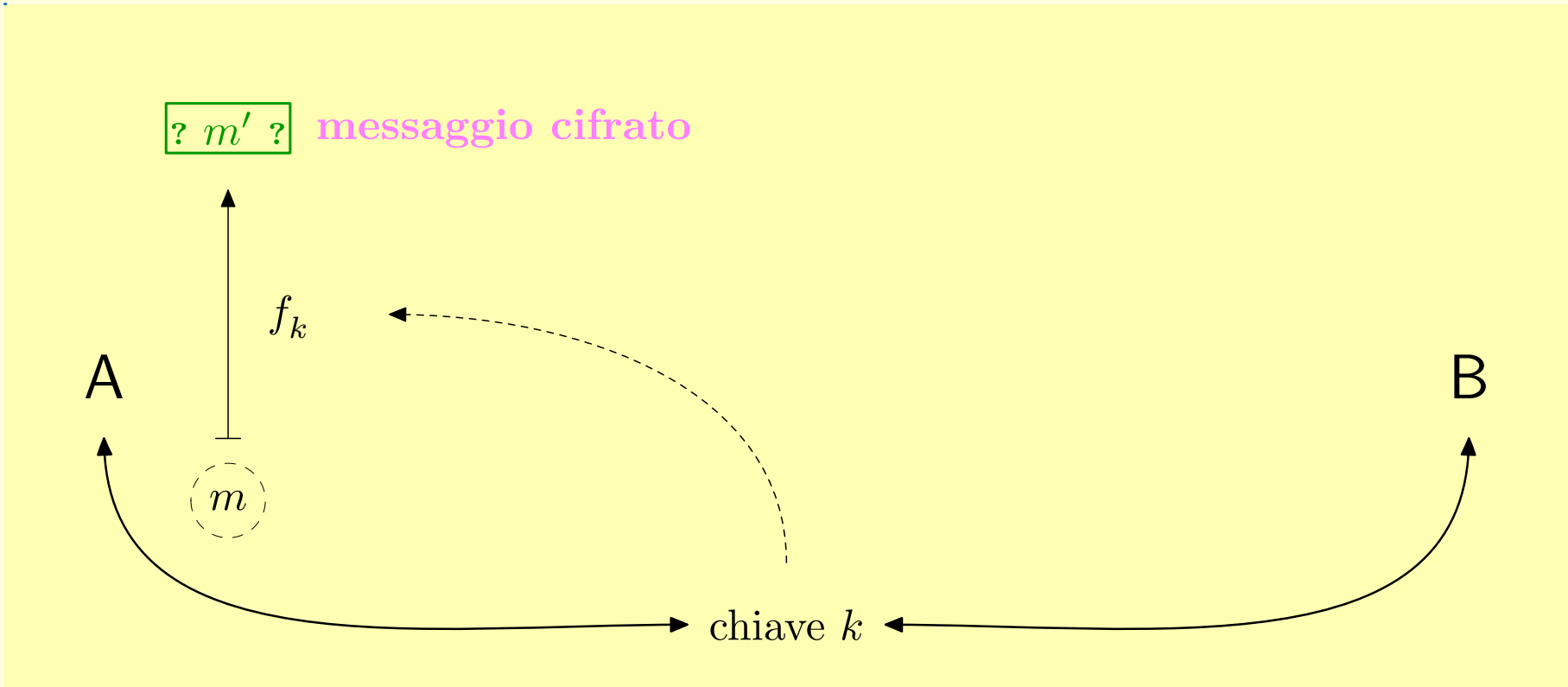
Schema di trasmissione a chiave privata:



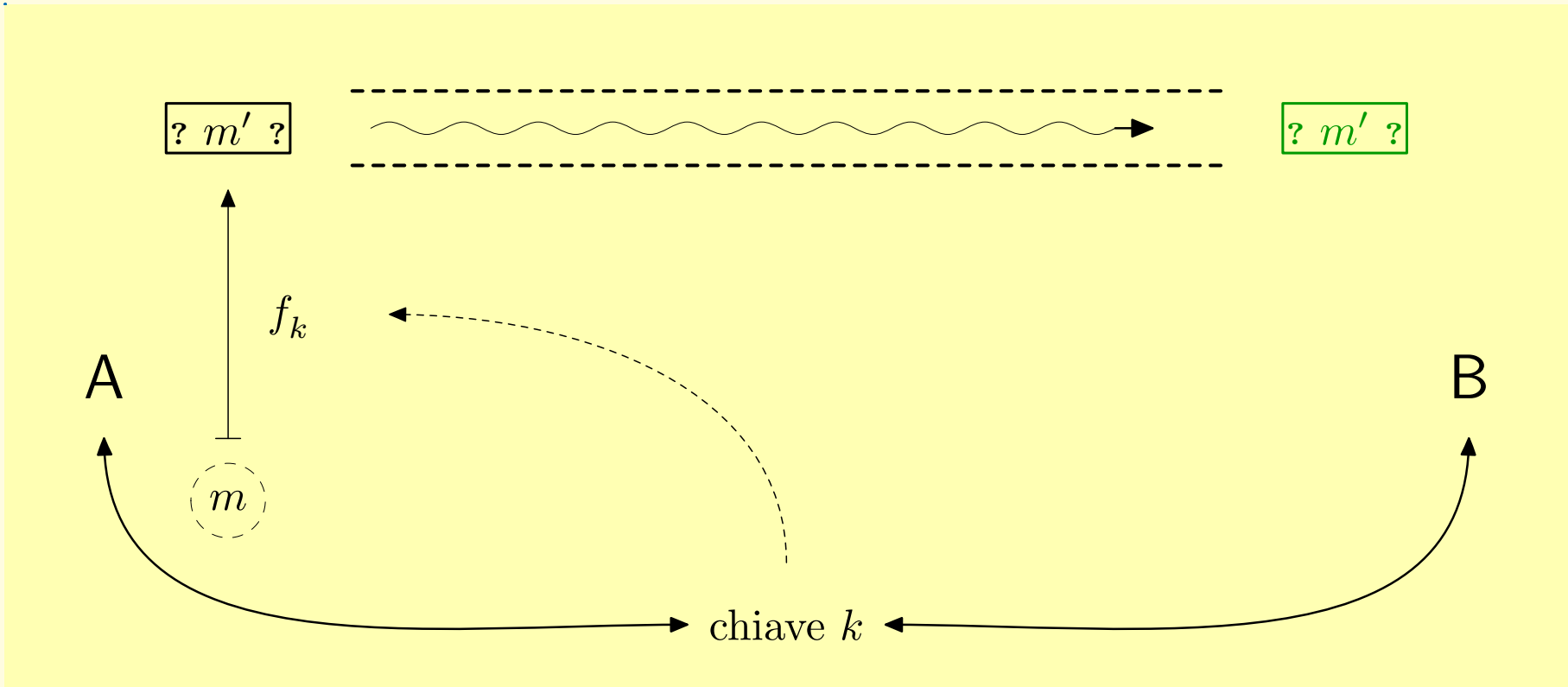
Schema di trasmissione a chiave privata:



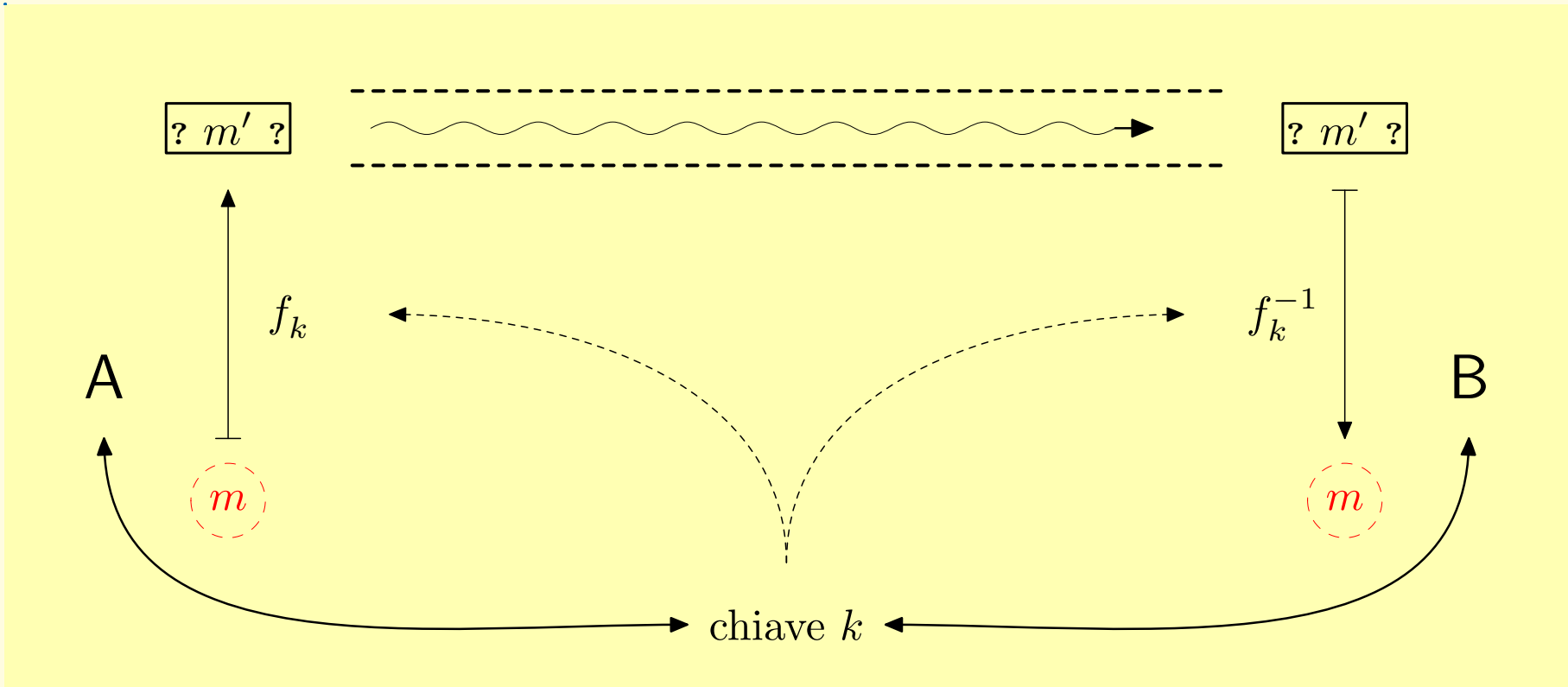
Schema di trasmissione a chiave privata:



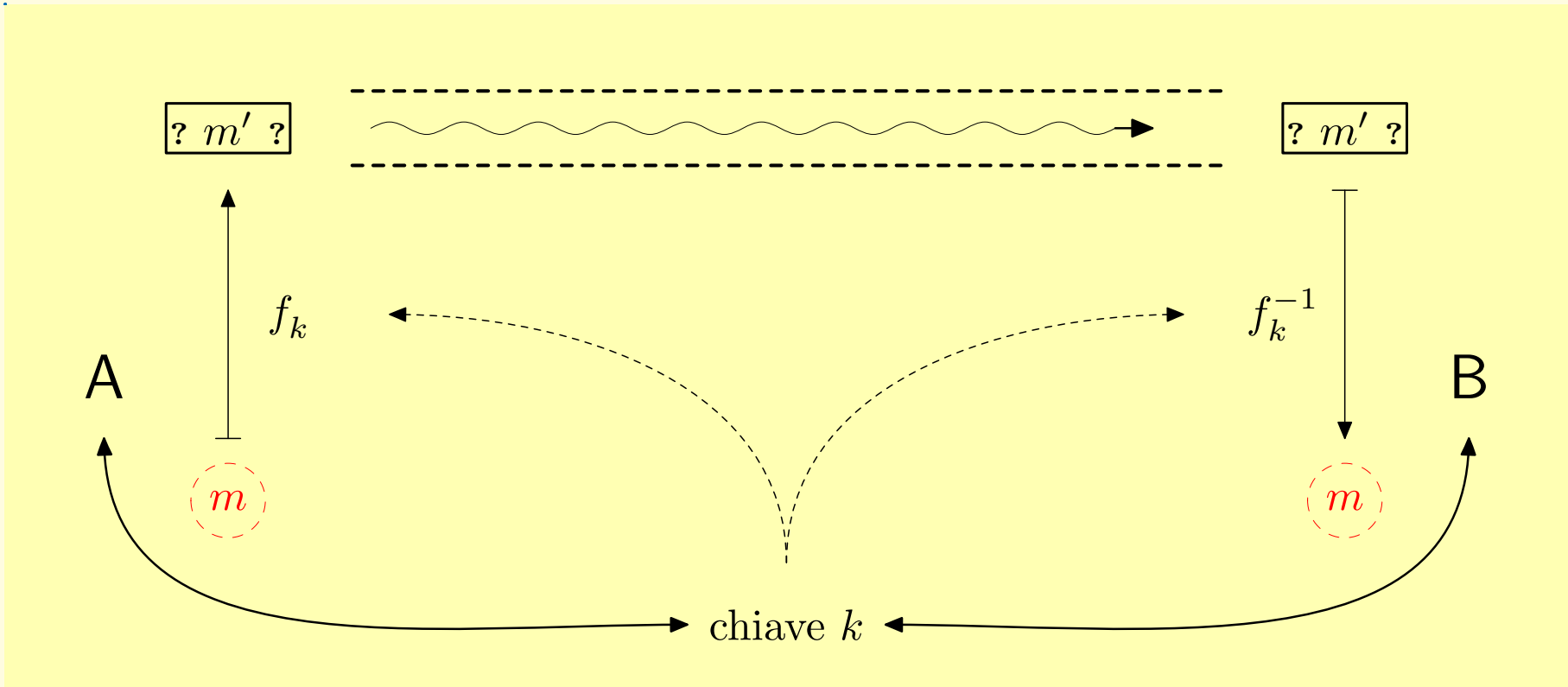
Schema di trasmissione a chiave privata:



Schema di trasmissione a chiave privata:

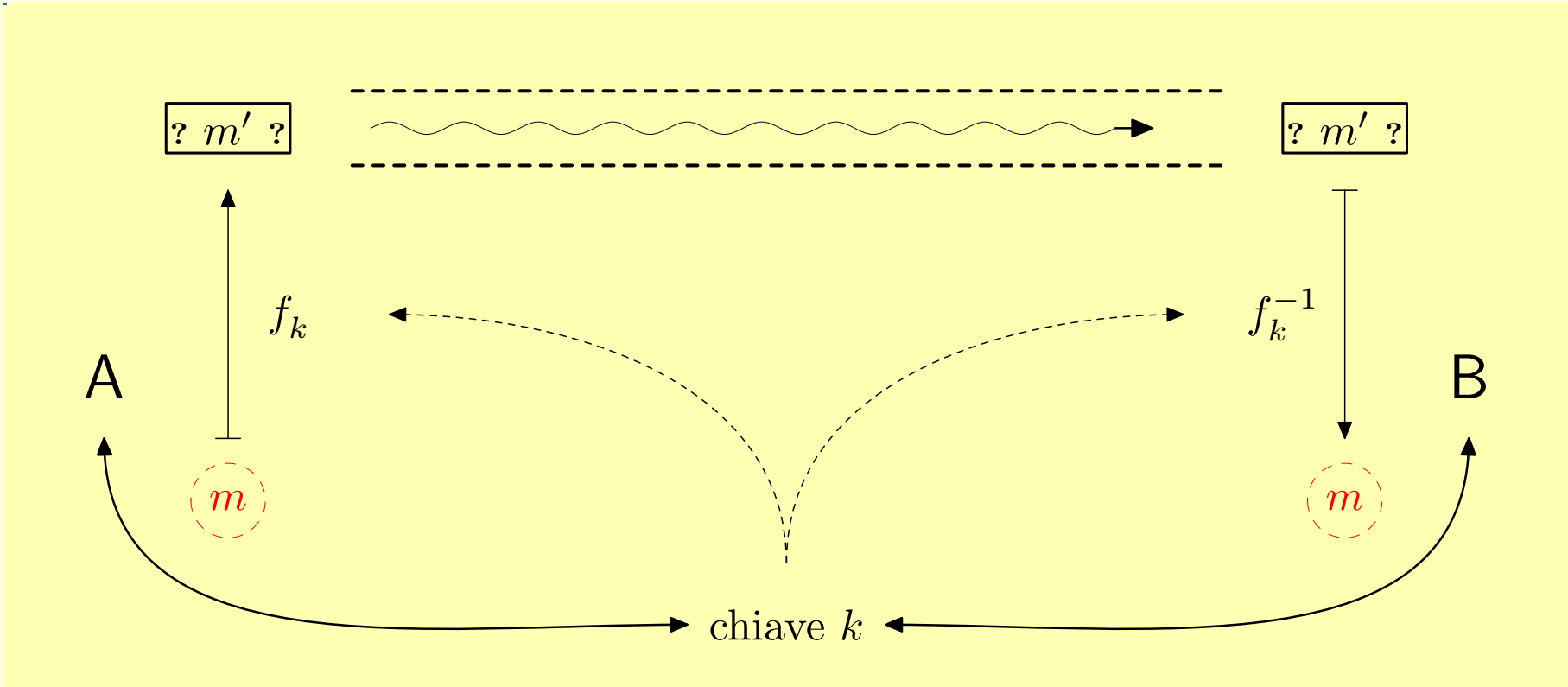


Schema di trasmissione a chiave privata:



Il principale punto debole di questo sistema è costituito dalle chiavi.

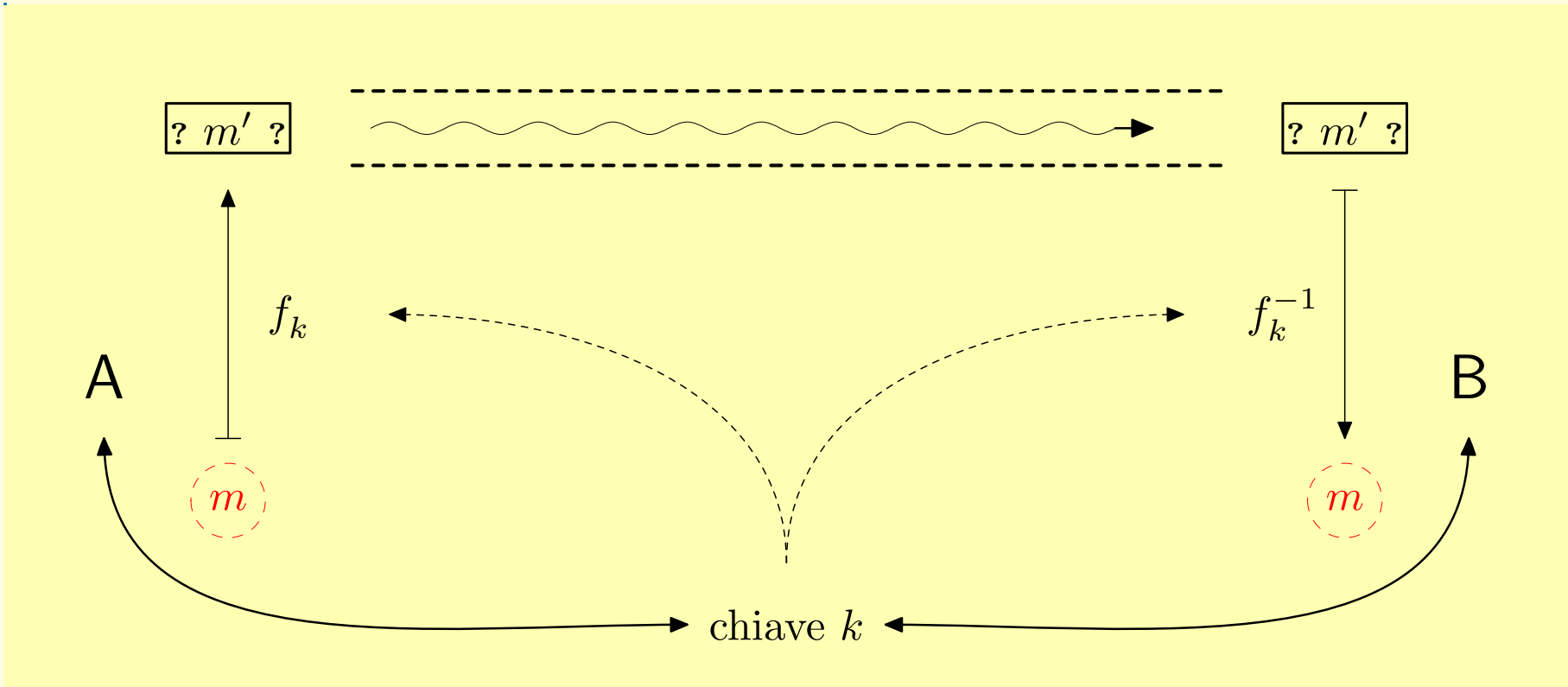
Schema di trasmissione a chiave privata:



Problema:
gestione delle chiavi

- servono chiavi lunghe per garantire la sicurezza;

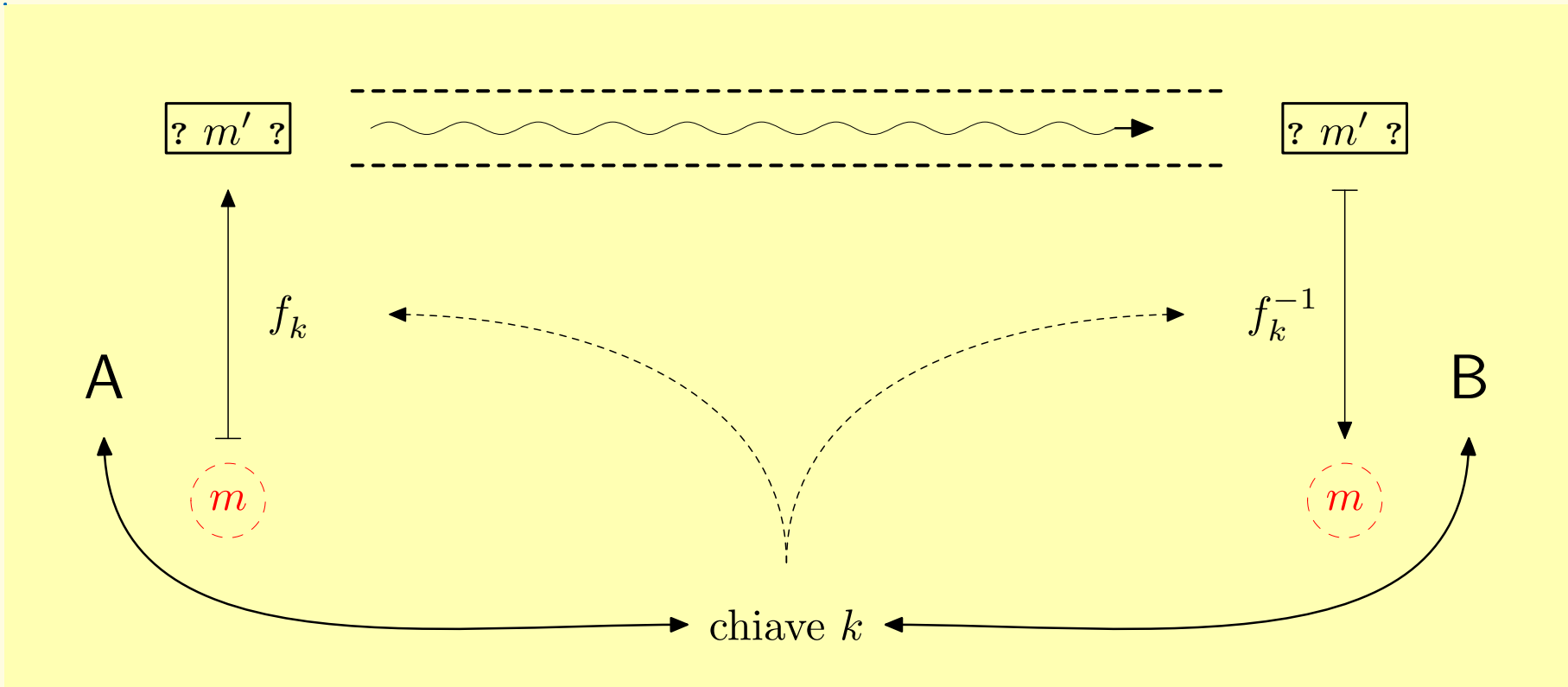
Schema di trasmissione a chiave privata:



Problema:
gestione delle chiavi

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

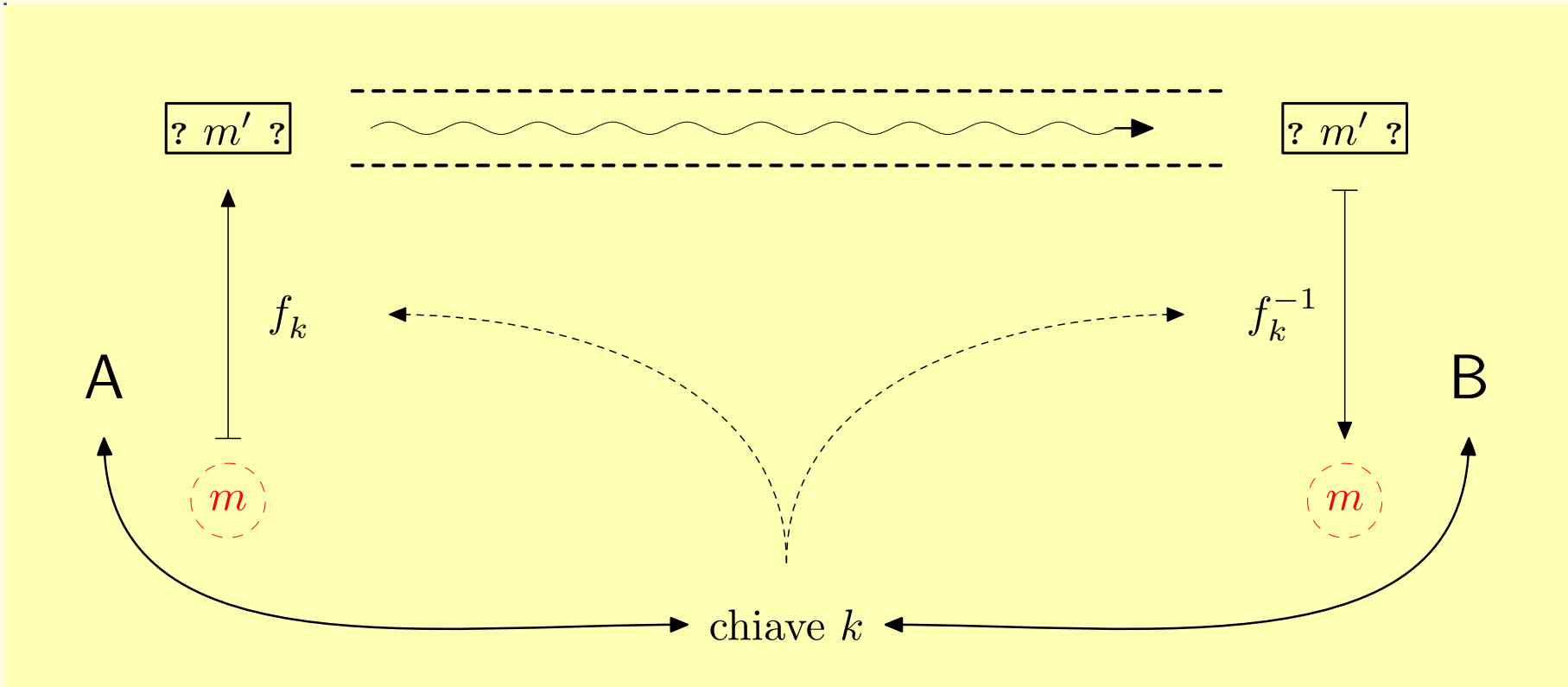
Schema di trasmissione a chiave privata:



Problema:
gestione delle chiavi

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

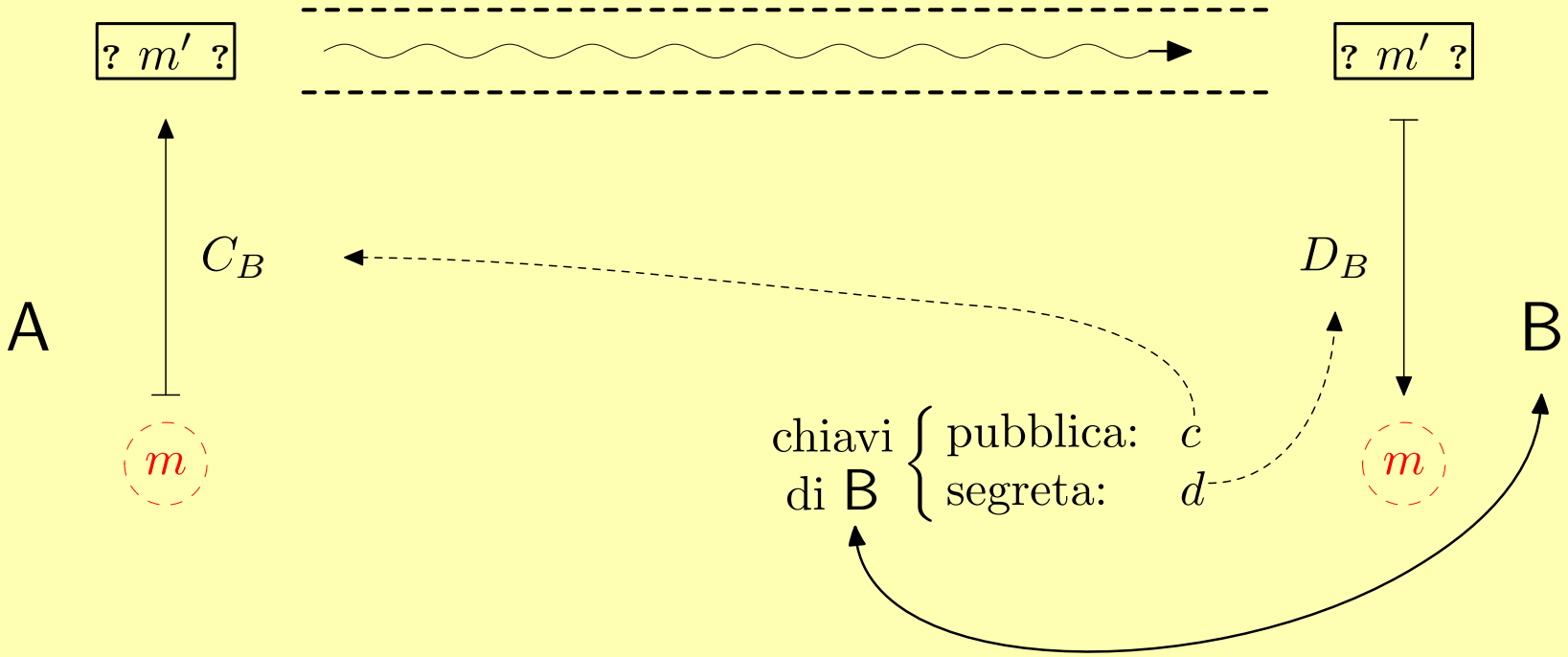
Schema di trasmissione a chiave privata:



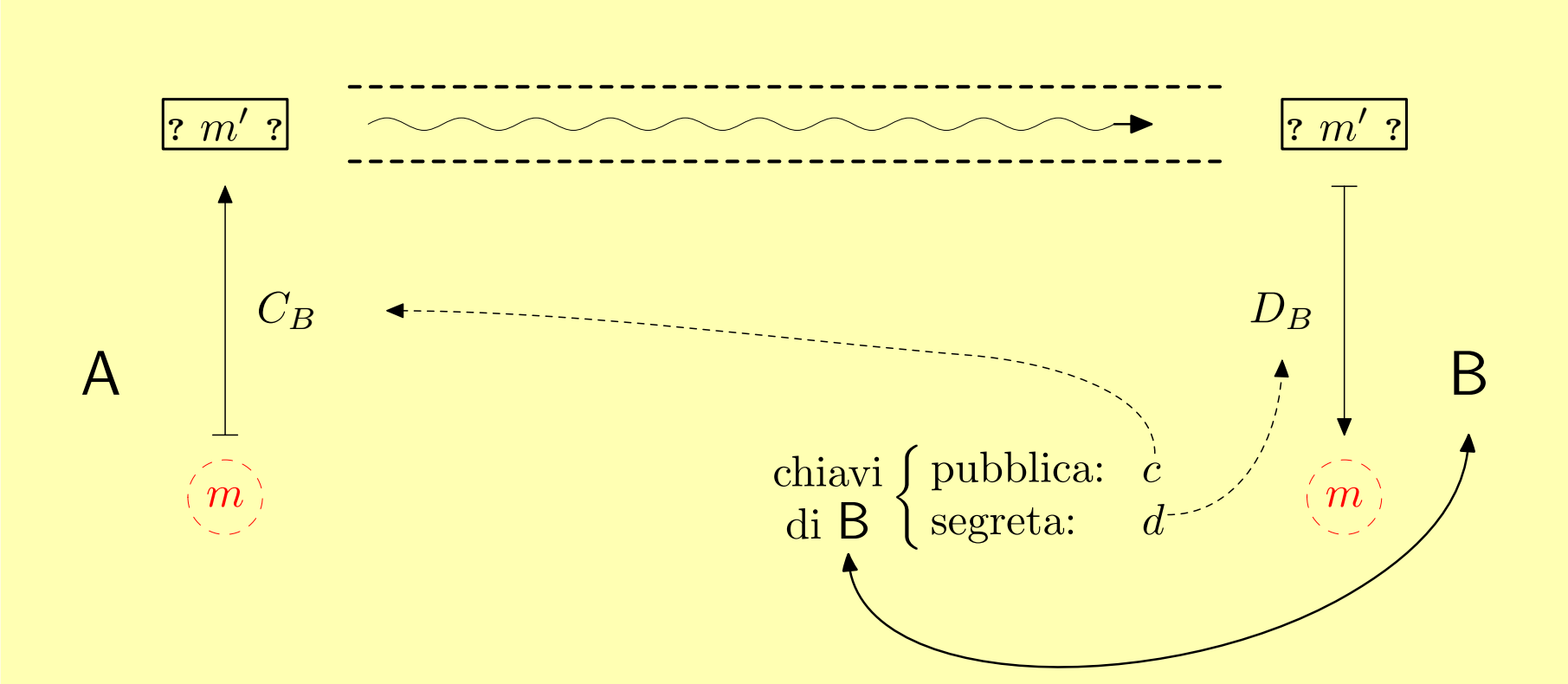
Problema:
gestione delle chiavi

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

Schema di trasmissione a chiave pubblica:

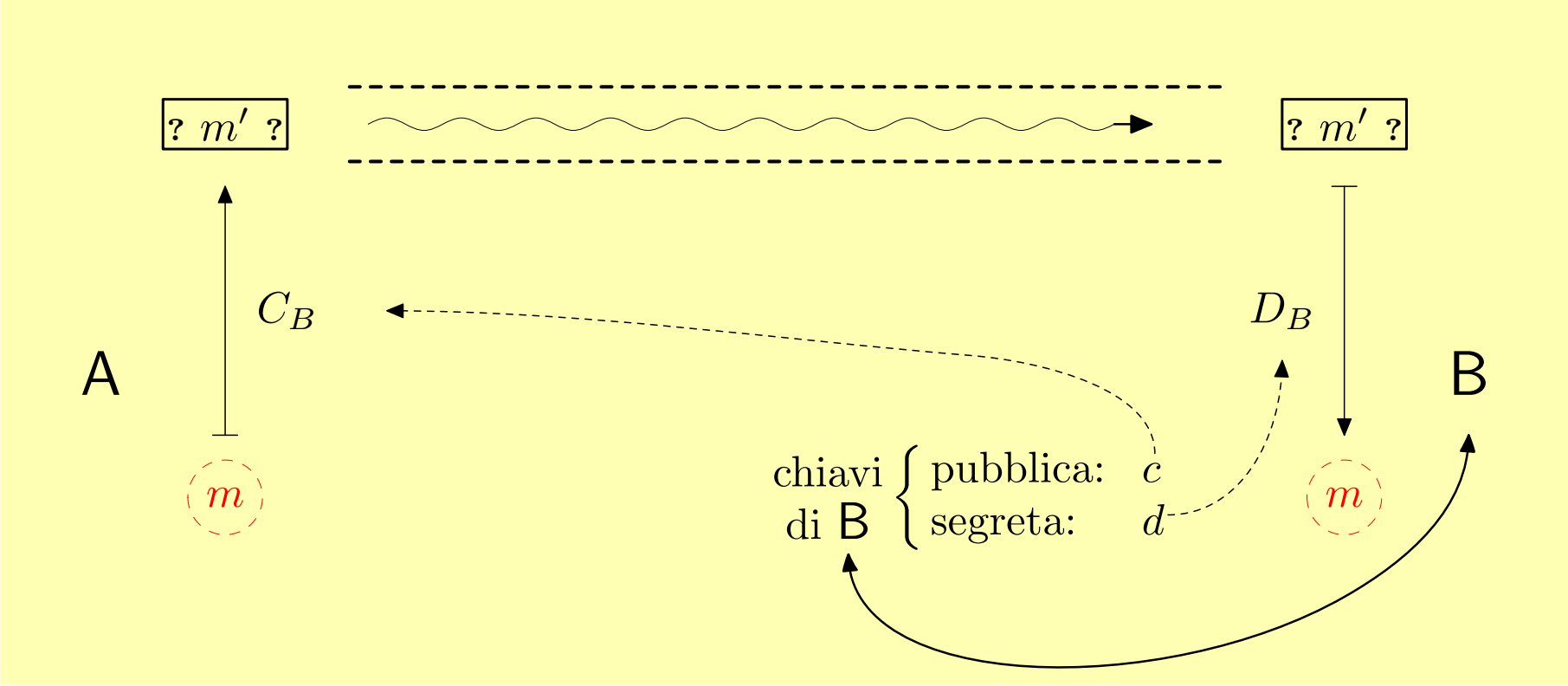


Schema di trasmissione a chiave pubblica:



Requisiti:

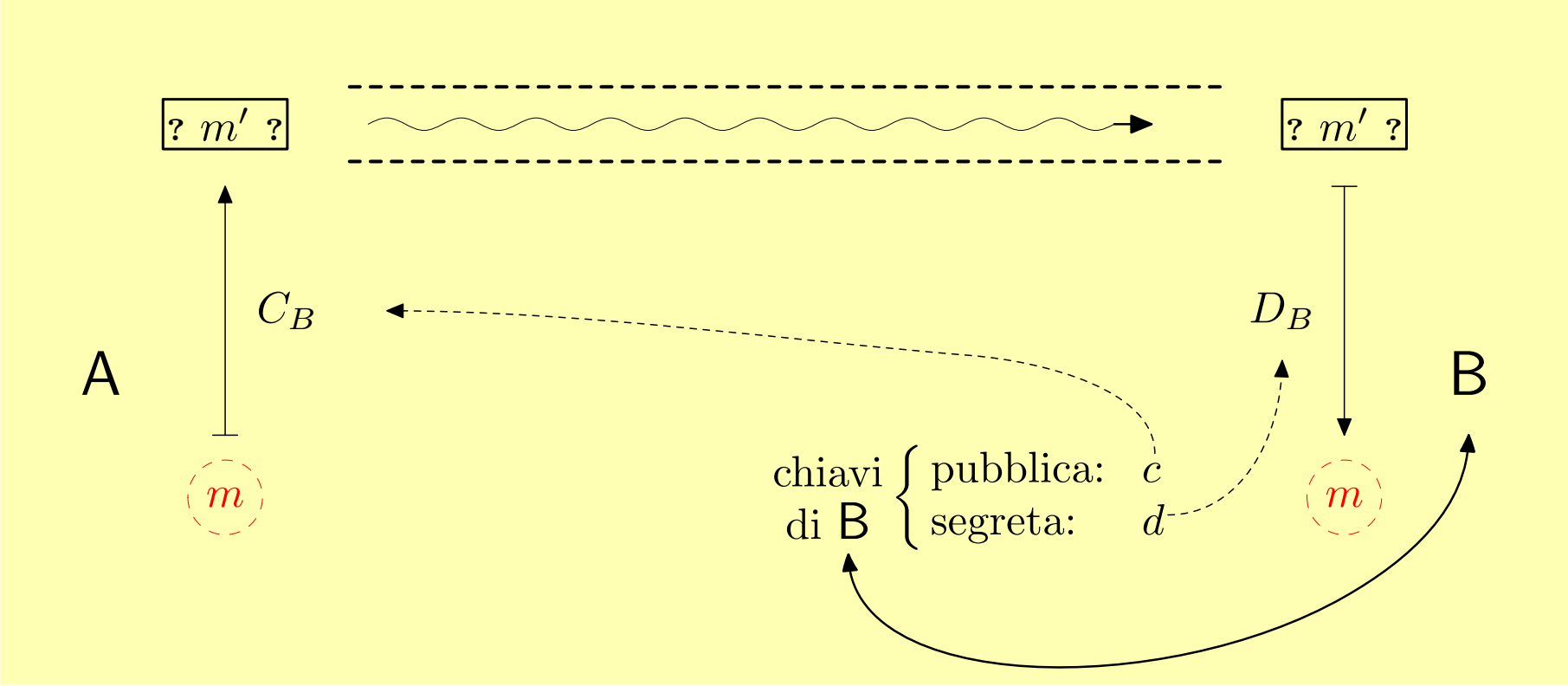
Schema di trasmissione a chiave pubblica:



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

Requisiti:

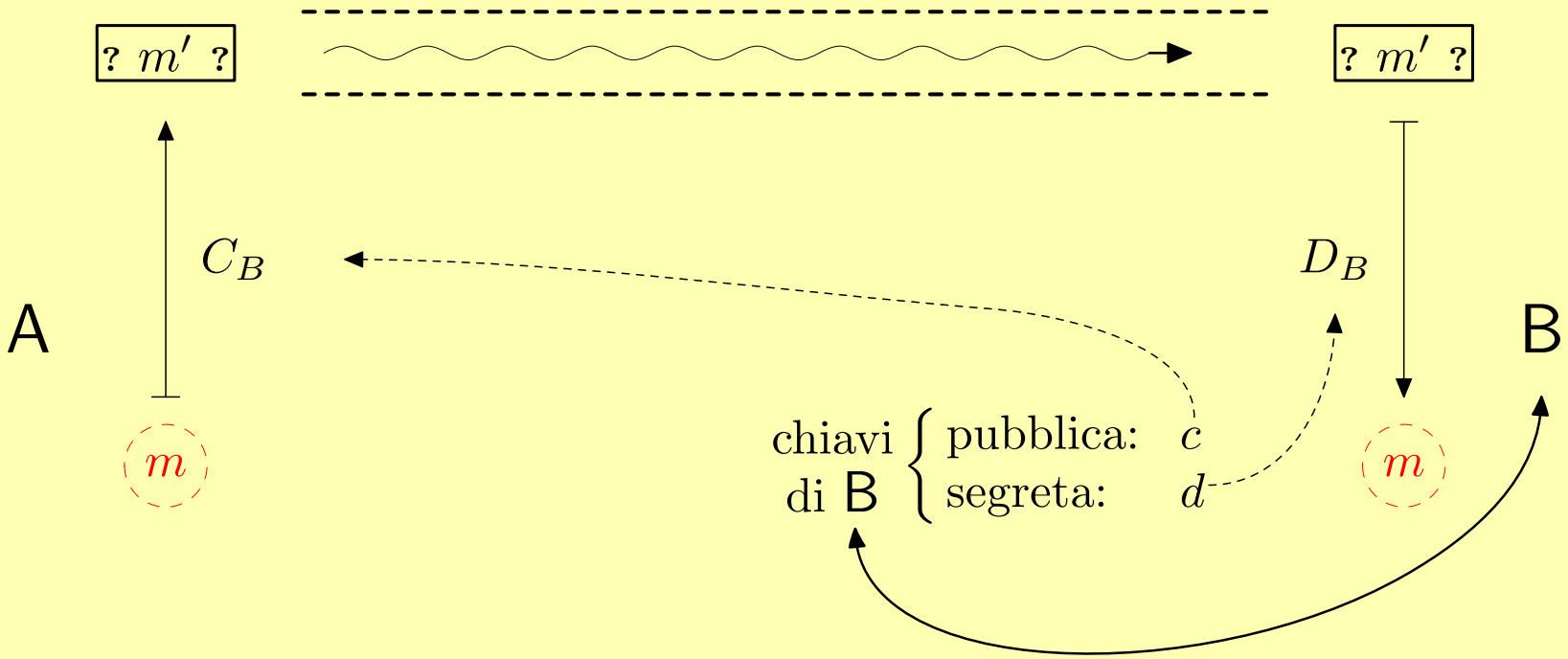
Schema di trasmissione a chiave pubblica:



Requisiti:

- D_B è l'inversa di C_B
  (non strettamente necessario, se non per l'autenticazione);
- dato m, è computazionalmente facile calcolare $C_B(m)$;

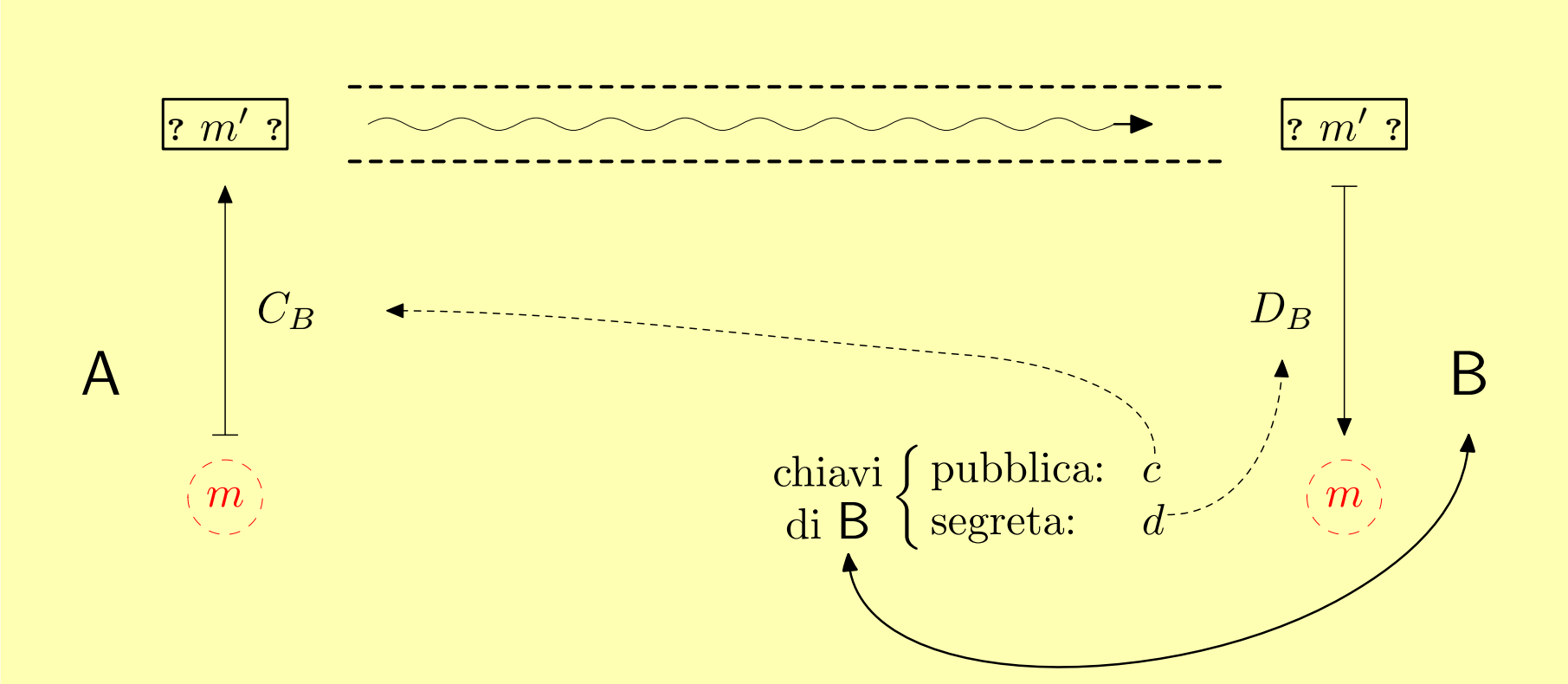
Schema di trasmissione a chiave pubblica:



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.

Schema di trasmissione a 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.

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;

...

$[n-1]_n$, l'insieme degli interi che, divisi per n, hanno resto $n-1$.

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

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

Compatibilità con le operazioni: fissato n, per ogni a, b, a', b' vale:

$$\begin{pmatrix} a \equiv_n a' \\ b \equiv_n b' \end{pmatrix} \implies \begin{pmatrix} a+b \equiv_n a'+b' \\ ab \equiv_n a'b' \end{pmatrix}$$

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 $[a]_n[b]_n = [ab]_n$, quindi una "aritmetica modulo n".

Compatibilità con le operazioni: fissato n, per ogni a, b, a', b' vale:

$$\begin{pmatrix} a \equiv_n a' \\ b \equiv_n b' \end{pmatrix} \implies \begin{pmatrix} a+b \equiv_n a'+b' \\ ab \equiv_n a'b' \end{pmatrix}$$

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 $[a]_n[b]_n = [ab]_n$, quindi una "aritmetica modulo n".

Esempio: Se $n = 2$, ponendo $P = [0]_2$ e $D = [1]_2$, le regole di calcolo sono:

$$P + P = D + D = P \qquad PP = PD = DP = P$$
$$P + D = D + P = D \qquad DD = D$$

che equivalgono a:

$$0 + 0 \equiv_2 1 + 1 \equiv_2 0 \qquad 0 \cdot 0 \equiv_2 0 \cdot 1 \equiv_2 1 \cdot 0 \equiv_2 0$$
$$0 + 1 \equiv_2 1 + 0 \equiv_2 1 \qquad 1 \cdot 1 \equiv_2 1$$

Due utili fatti (ancora abbastanza elementari):

Due utili fatti (ancora abbastanza elementari):

- 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 $aa' \equiv_n 1$.

  Questo a' è facile da calcolare grazie ad un algoritmo molto efficiente (l'algoritmo euclideo).

Due utili fatti (ancora abbastanza elementari):

- 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 $aa' \equiv_n 1$.

  Questo a' è facile da calcolare grazie ad un algoritmo molto efficiente (l'algoritmo euclideo).

- 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;
  ⋆ 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;

Schema di RSA:

Schema di RSA:

1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e segreta) in questo modo:

Schema di RSA:

1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e segreta) in questo modo:

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

Schema di RSA:

1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e segreta) 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 $\varphi := (p-1)(q-1)$;

Schema di RSA:

1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e segreta) 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 $\varphi := (p-1)(q-1)$;
- calcola l'inverso d di c modulo φ (dunque $cd \equiv_\varphi 1$).

Schema di RSA:

1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e segreta) 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 $\varphi := (p-1)(q-1)$;
- calcola l'inverso d di c modulo φ (dunque $cd \equiv_\varphi 1$).
- chiavi:
  - ⋆ chiave pubblica: (n, c);
  - ⋆ chiave segreta: d.

Schema di RSA:

1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e segreta) 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 $\varphi := (p-1)(q-1)$;
- calcola l'inverso d di c modulo φ (dunque $cd \equiv_\varphi 1$).
- chiavi:
  - ⋆ chiave pubblica: (n, c);
  - ⋆ chiave segreta: d.

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

Schema di RSA:

2. 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 rapidi per farlo, molto più rapidi che calcolarsi prima m^c e poi il resto di questo numero intero). Il messaggio cifrato sarà appunto questo resto, m'.

Schema di RSA:

2. 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 rapidi per farlo, molto più rapidi che calcolarsi prima m^c e poi il resto di questo numero intero). Il messaggio cifrato sarà appunto questo resto, m'.

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

Schema di RSA:

2. 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 rapidi per farlo, molto più rapidi che calcolarsi prima m^c e poi il resto di questo numero intero). Il messaggio cifrato sarà appunto questo resto, m'.

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

Perché funziona?

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

Schema di RSA:

2. 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 rapidi per farlo, molto più rapidi che calcolarsi prima m^c e poi il resto di questo numero intero). Il messaggio cifrato sarà appunto questo resto, m'.

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

E lo spione?

Se una persona diversa da **B** sapesse fattorizzare n, allora saprebbe anche calcolare $\varphi = (p-1)(q-1)$ 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. 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 (presumibilmente) altissimo. Ciò non esclude che esista qualche metodo per scardinare RSA che prescinda dalla fattorizzazione di n. Al momento (forse, speriamo) non sono noti tali metodi.

Se una persona diversa da **B** sapesse fattorizzare n, allora saprebbe anche calcolare $\varphi = (p-1)(q-1)$ 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. 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 (presumibilmente) altissimo. Ciò non esclude che esista qualche metodo per scardinare RSA che prescinda dalla fattorizzazione di n. Al momento (forse, speriamo) non sono noti tali metodi.

RSA fornisce anche un metodo sicuro di autenticazione.

Se una persona diversa da **B** sapesse fattorizzare n, allora saprebbe anche calcolare $\varphi = (p-1)(q-1)$ 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. 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 (presumibilmente) altissimo. Ciò non esclude che esista qualche metodo per scardinare RSA che prescinda dalla fattorizzazione di n. Al momento (forse, speriamo) non sono noti tali metodi.

RSA fornisce anche un metodo sicuro di autenticazione.

Una morale . . .

il problema del logaritmo discreto: Dati gli interi a, b, n (con $n > 0$), se $b \equiv_n a^l$ 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 difficoltà 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.

il problema del logaritmo discreto: Dati gli interi a, b, n (con $n > 0$), se $b \equiv_n a^l$ 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 difficoltà 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
2	2916
3	815
4	1266
5	1651

6	3731
7	292
8	4182
9	1630
10	1

11	196
12	2081
13	552
14	3208
15	2466

16	2532
17	213
18	4546
19	2143
20	2917

21	1107
22	3112
23	2659
24	431
25	3302

$\cdots$

come A e B si possono accordare su una chiave senza trasmetterla (protocollo Diffie-Hellman):

come A e B si possono accordare su una chiave senza trasmetterla (protocollo Diffie-Hellman):

- **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, il resto di a^α modulo p;

come A e B si possono accordare su una chiave senza trasmetterla (protocollo Diffie-Hellman):

- **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, il resto di a^α modulo p; Esempio: **A**: $[\alpha = 5]$ $\boxed{p = 13,\ a = 7,\ a_1 = 11}$ $\rightarrow$ **B**

come A e B si possono accordare su una chiave senza trasmetterla (protocollo Diffie-Hellman):

- **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, il resto di a^α modulo 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** il resto, a_2, di a^β modulo p;

come A e B si possono accordare su una chiave senza trasmetterla (protocollo Diffie-Hellman):

- **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, il resto di a^α modulo 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** il resto, a_2, di a^β modulo p; **B**: $[\beta = 8]$ $\boxed{a_2 = 3}$ $\rightarrow$ **A**

come A e B si possono accordare su una chiave senza trasmetterla (protocollo Diffie-Hellman):

- **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, il resto di a^α modulo 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** il resto, a_2, di a^β modulo p; **B**: $[\beta = 8]$ $\boxed{a_2 = 3}$ $\rightarrow$ **A**
- A questo punto **A** e **B** possono calcolarsi la loro chiave comune k: il resto di $a_2^\alpha \equiv_p a_1^\beta \equiv_p a^{\alpha\beta}$ modulo p.

come A e B si possono accordare su una chiave senza trasmetterla (protocollo Diffie-Hellman):

- **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, il resto di a^α modulo 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** il resto, a_2, di a^β modulo p; **B**: $[\beta = 8]$ $\boxed{a_2 = 3}$ $\rightarrow$ **A**
- A questo punto **A** e **B** possono calcolarsi 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**: $11^8 \equiv_{13} 9$

come A e B si possono accordare su una chiave senza trasmetterla (protocollo Diffie-Hellman):

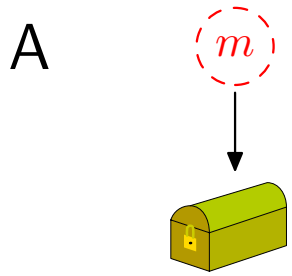
- **A** fissa un primo p, ed un opportuno intero a tale che $0 < a < p$, ed un intero (segreto) α, e trasmette a **B**: p, a e a_1, il resto di a^α modulo p; Esempio: **A**: $[\alpha = 5]$ $\boxed{p = 13,\ a = 7,\ a_1 = 11}$ $\rightarrow$ **B**
- **B** fissa un intero segreto β e trasmette ad **A** il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

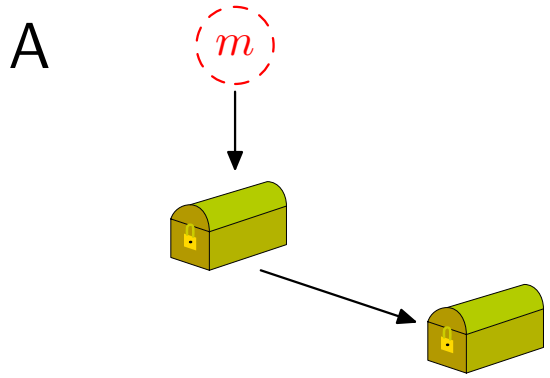
estensione: scambio di messaggi senza chiavi:

estensione: scambio di messaggi senza chiavi:

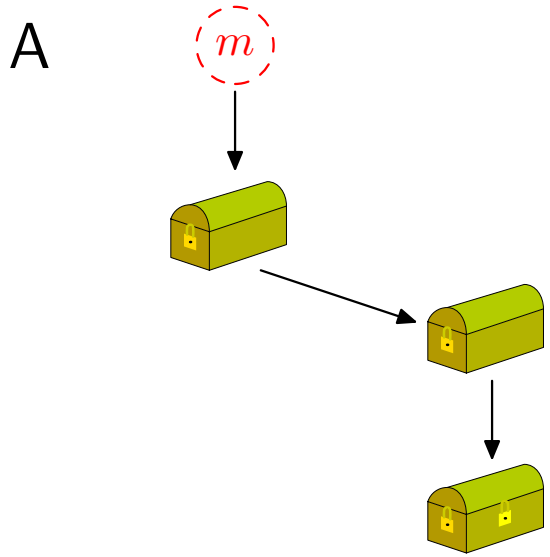
estensione: scambio di messaggi senza chiavi:



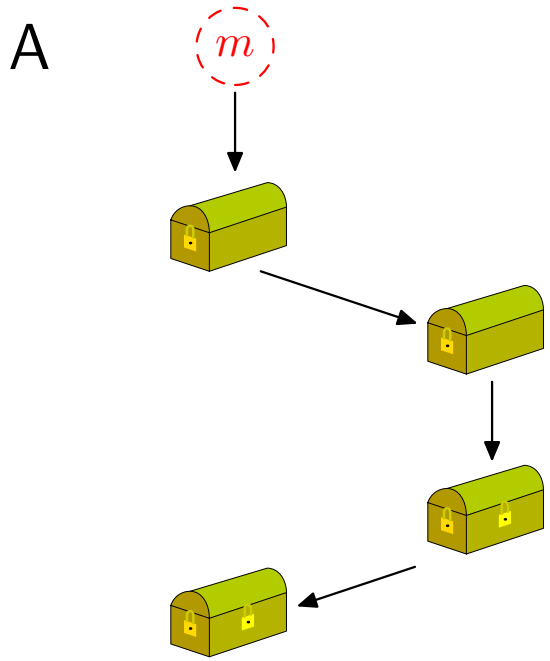
estensione: scambio di messaggi senza chiavi:



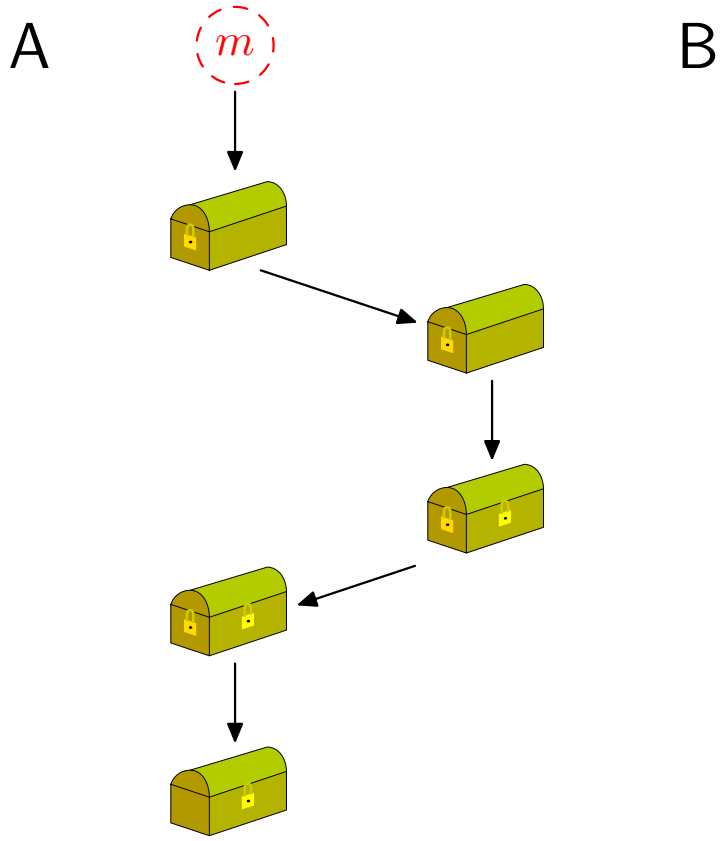
estensione: scambio di messaggi senza chiavi:



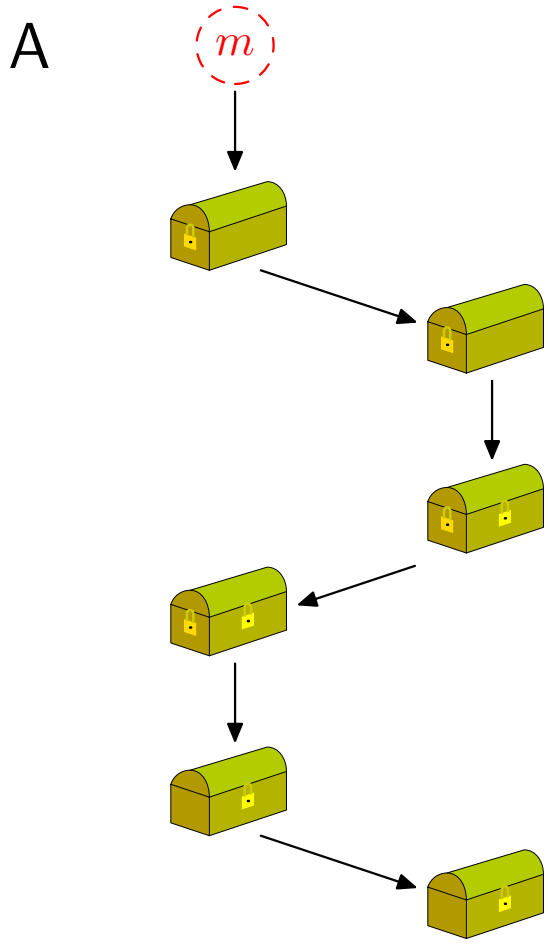
estensione: scambio di messaggi senza chiavi:



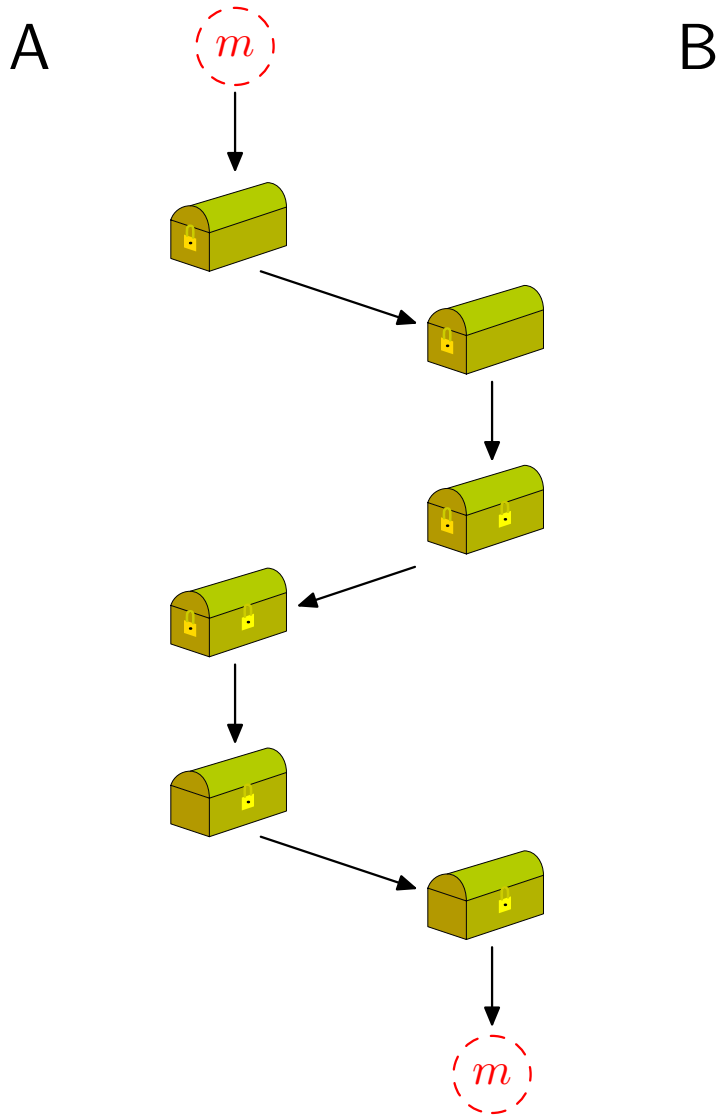
estensione: scambio di messaggi senza chiavi:



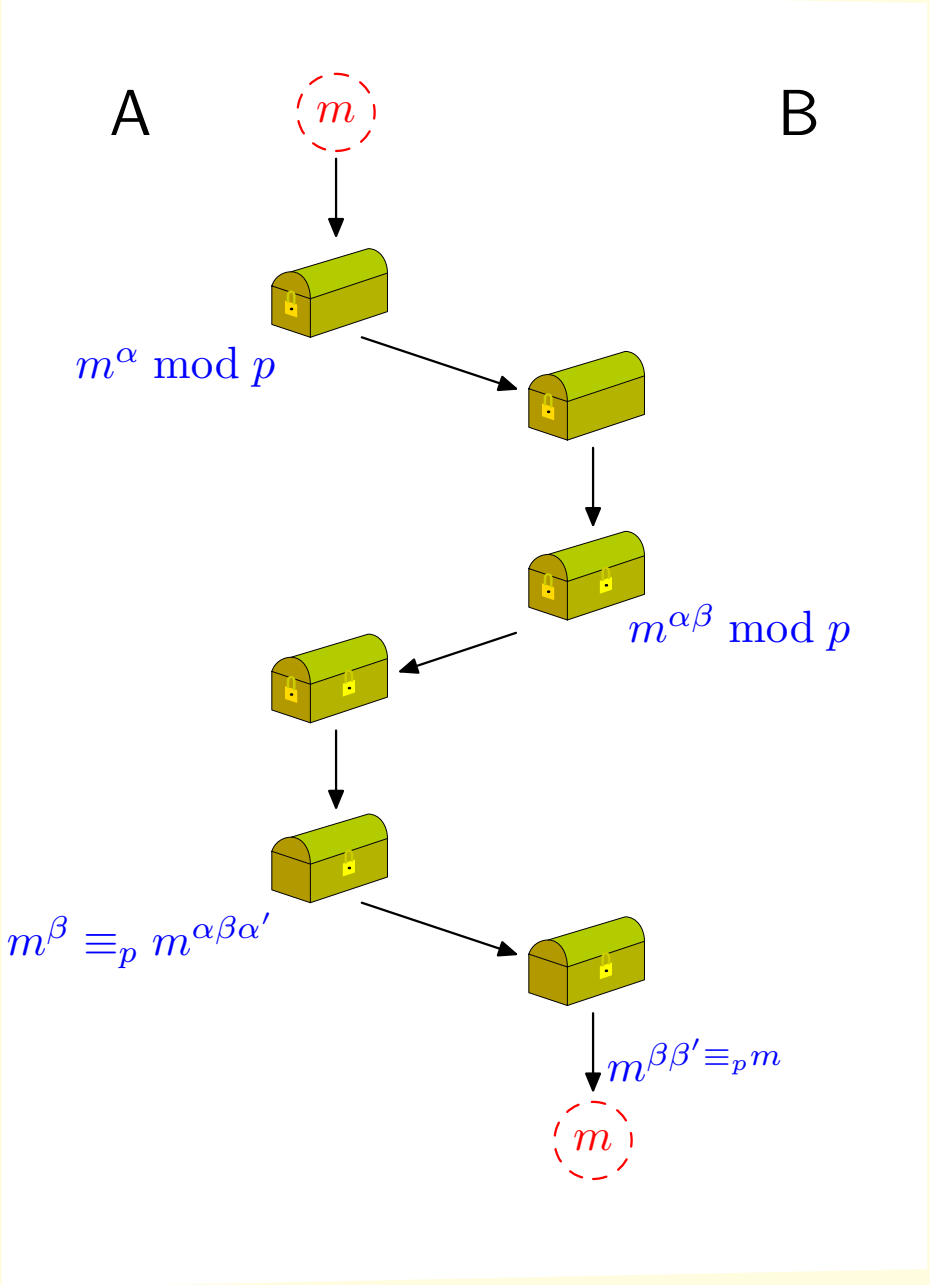
estensione: scambio di messaggi senza chiavi:



estensione: scambio di messaggi senza chiavi:



estensione: scambio di messaggi senza chiavi:



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

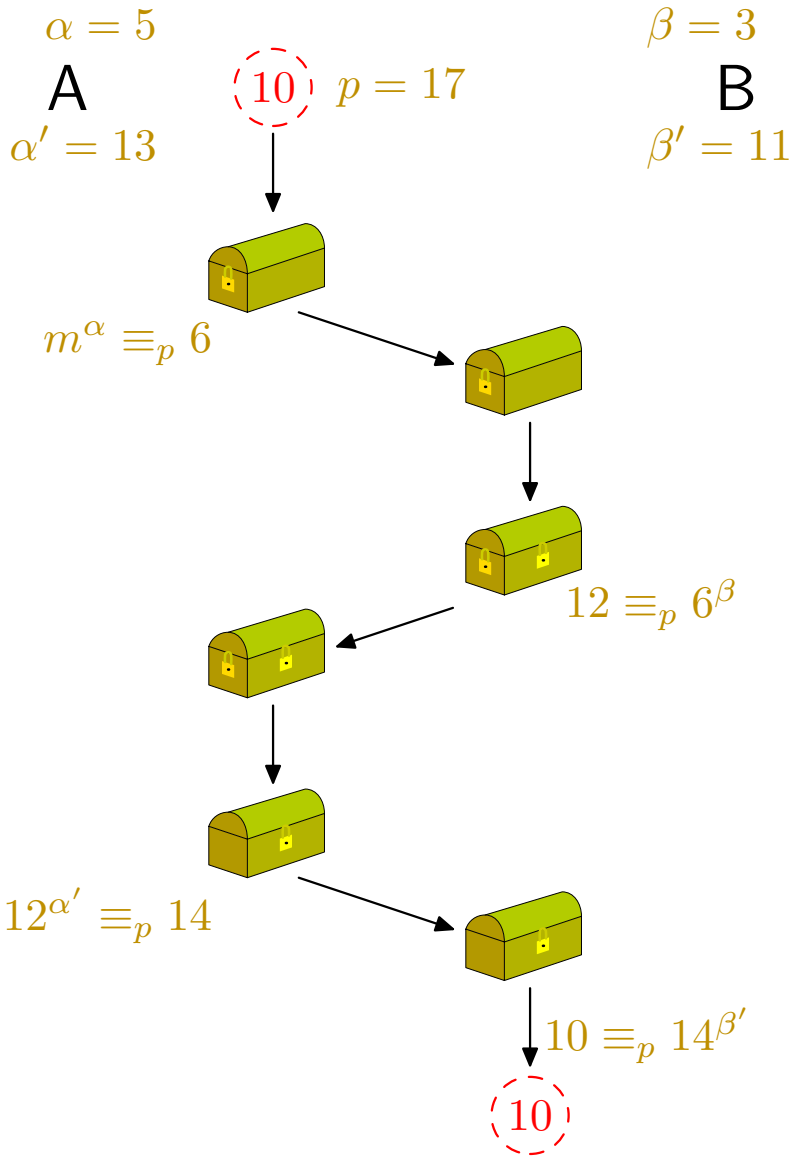
A può calcolare α' tale che

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

B può calcolare β' tale che

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

estensione: scambio di messaggi senza chiavi:



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

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

Schema di ElGamal:

Schema di ElGamal:

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

Schema di ElGamal:

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

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 $\left(a^\alpha, mc^\alpha\right)$.

Schema di ElGamal:

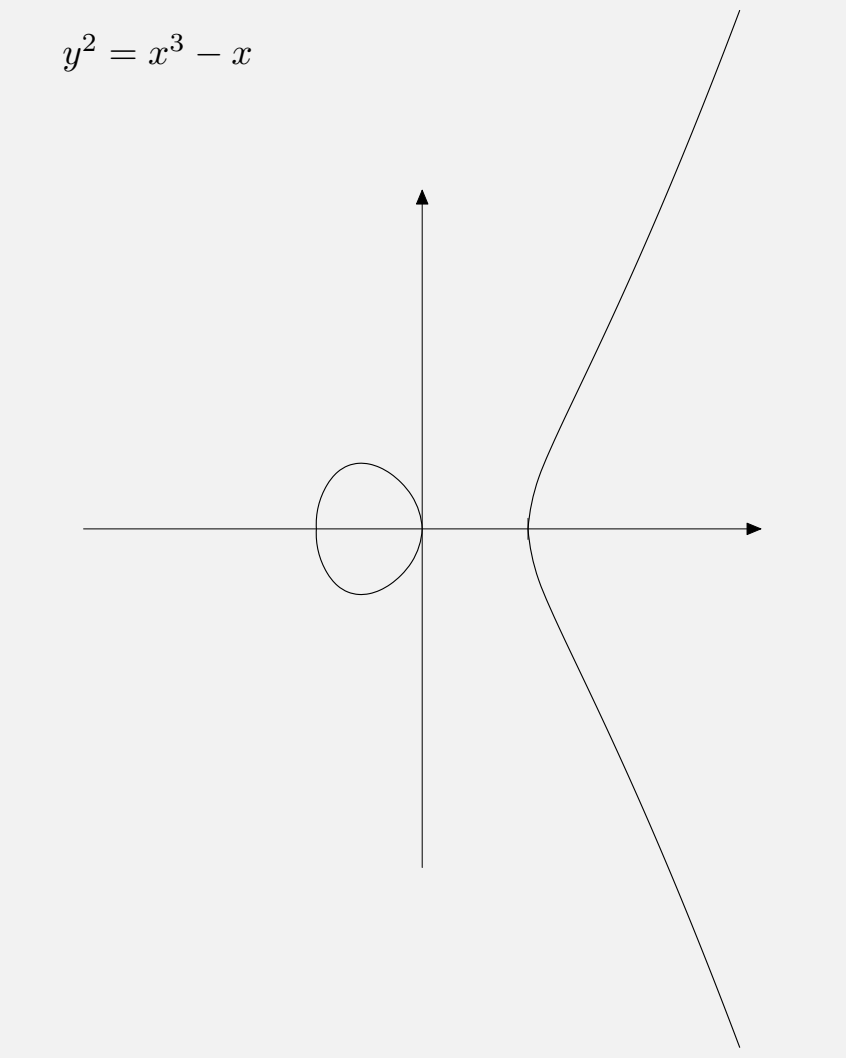
1. Scelta delle chiavi: ogni utente sceglie le sue chiavi (pubblica e segreta) in questo modo: viene scelto un "ambiente di calcolo" G (un gruppo), un elemento a di G, in modo che il PLD sia "molto difficile" per le potenze di a in G. Ciascun utente sceglie poi un intero d come chiave segreta e (G, a,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

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

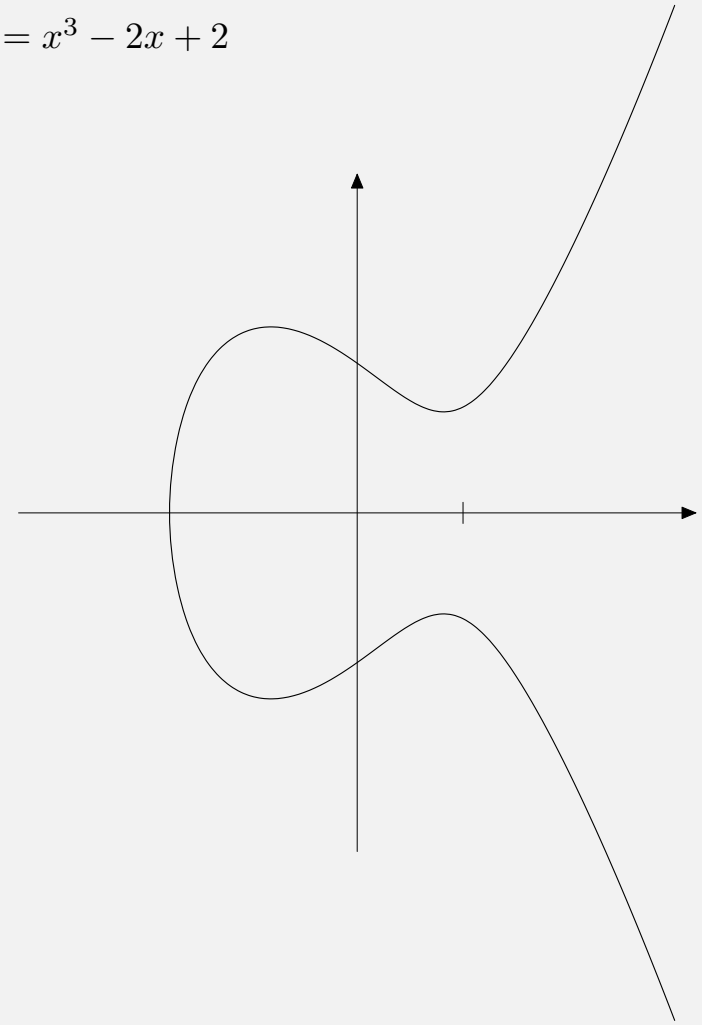
Curve ellittiche

Curve ellittiche

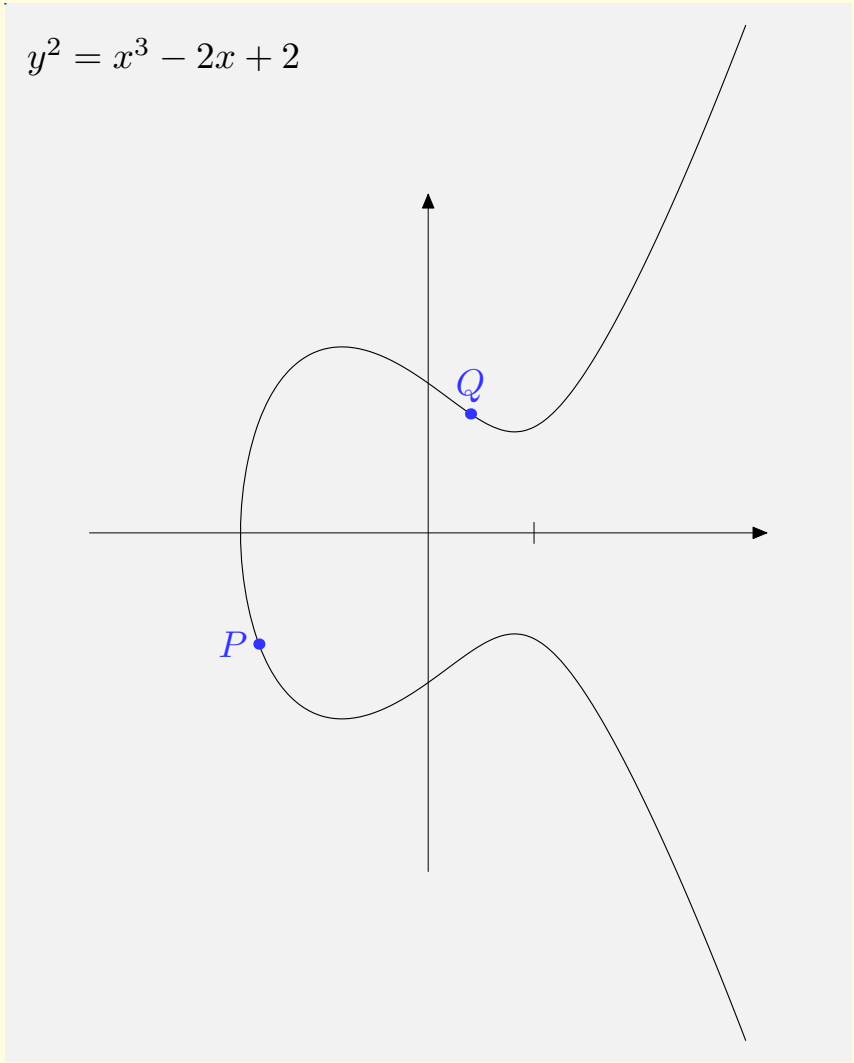


Curve ellittiche

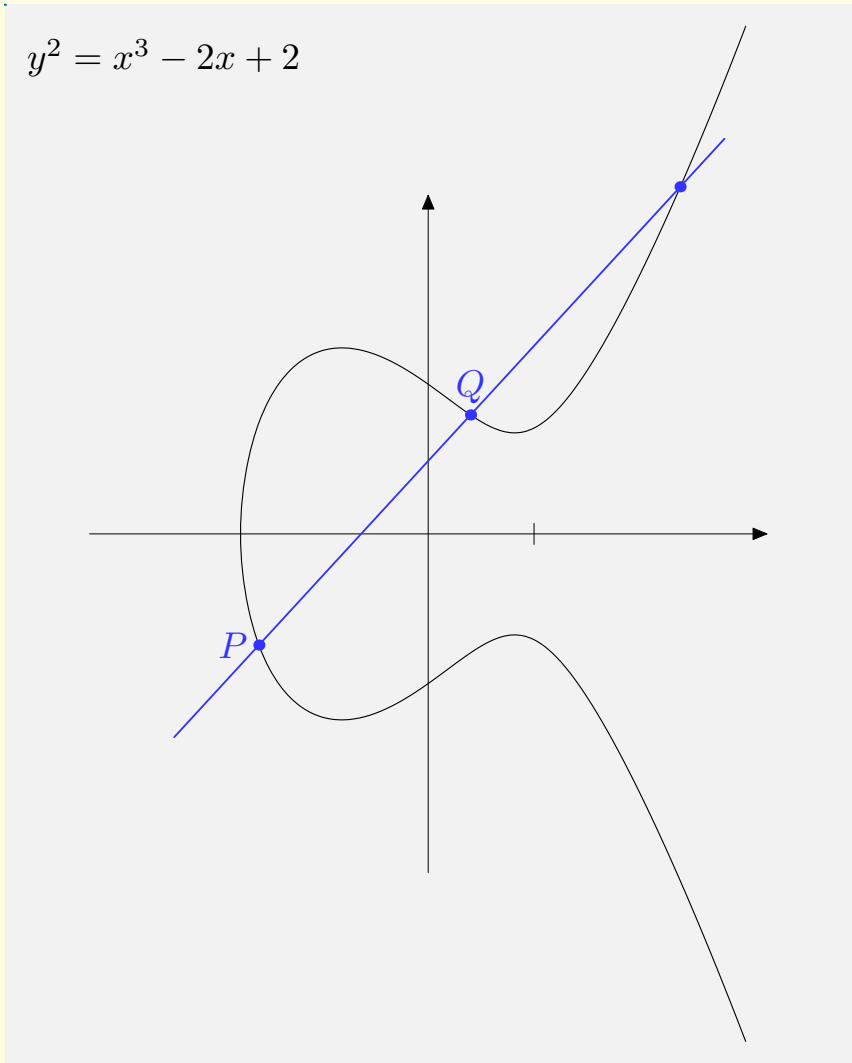
$y^2 = x^3 - 2x + 2$



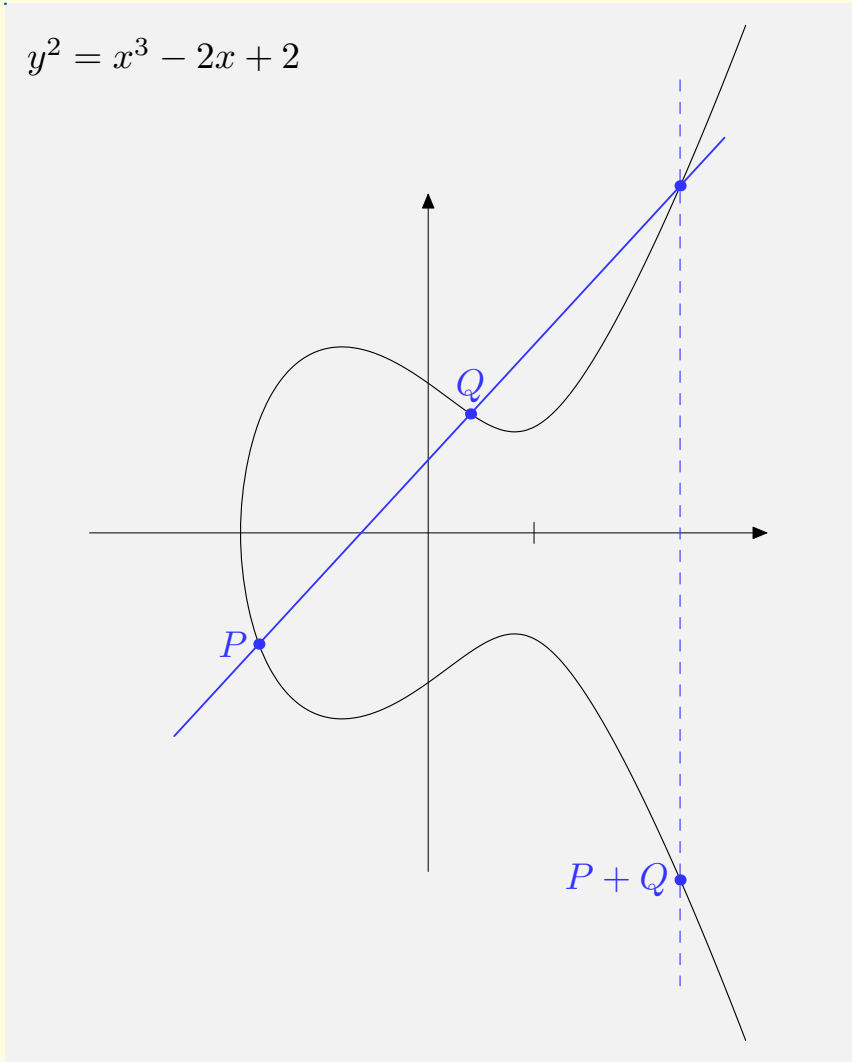
Curve ellittiche



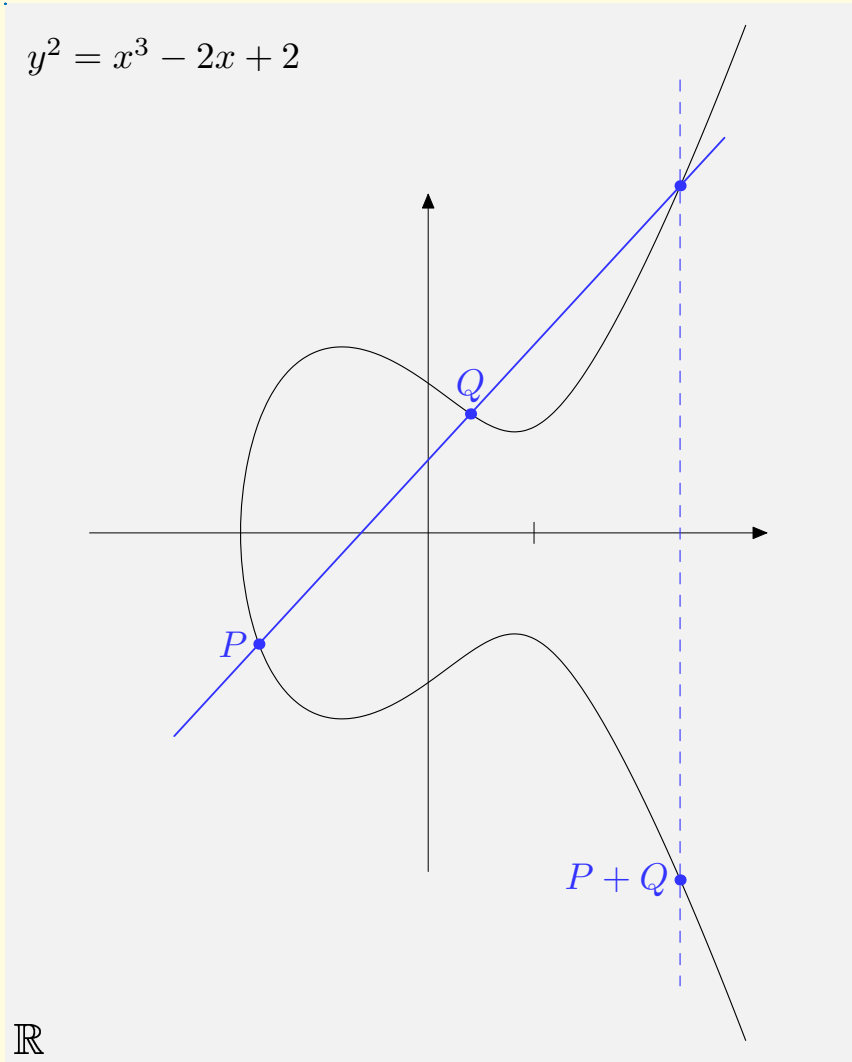
Curve ellittiche

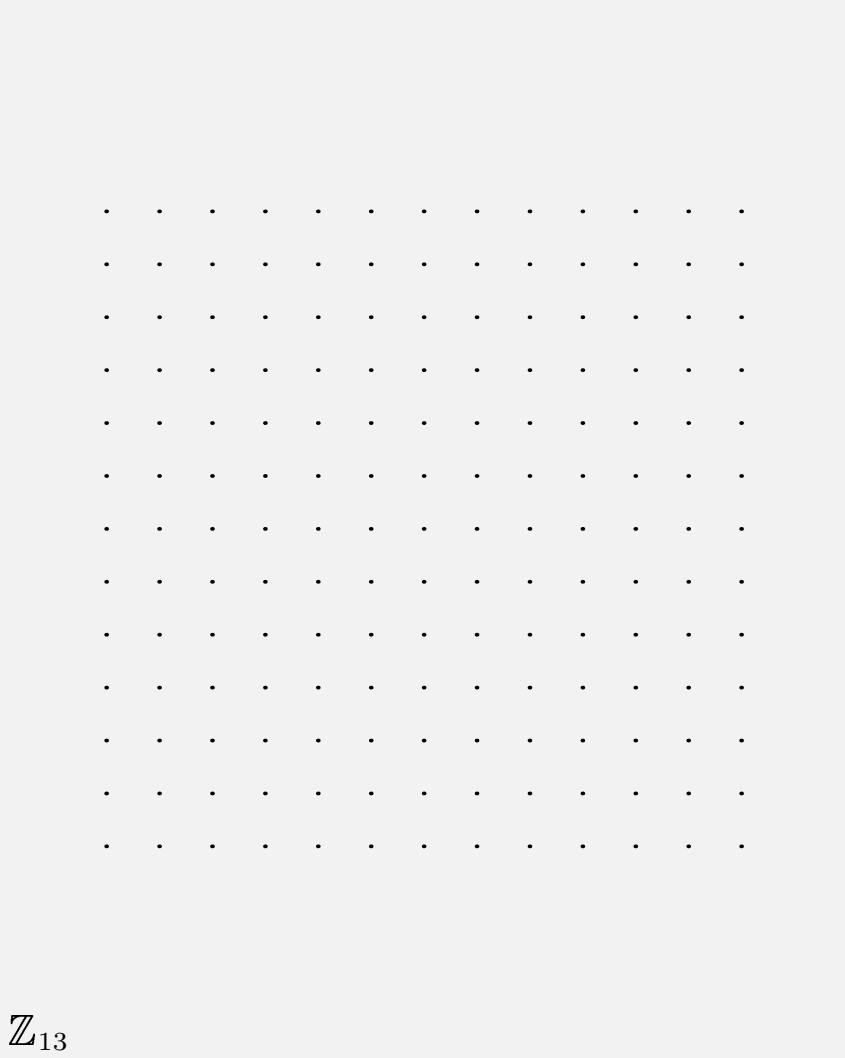


Curve ellittiche

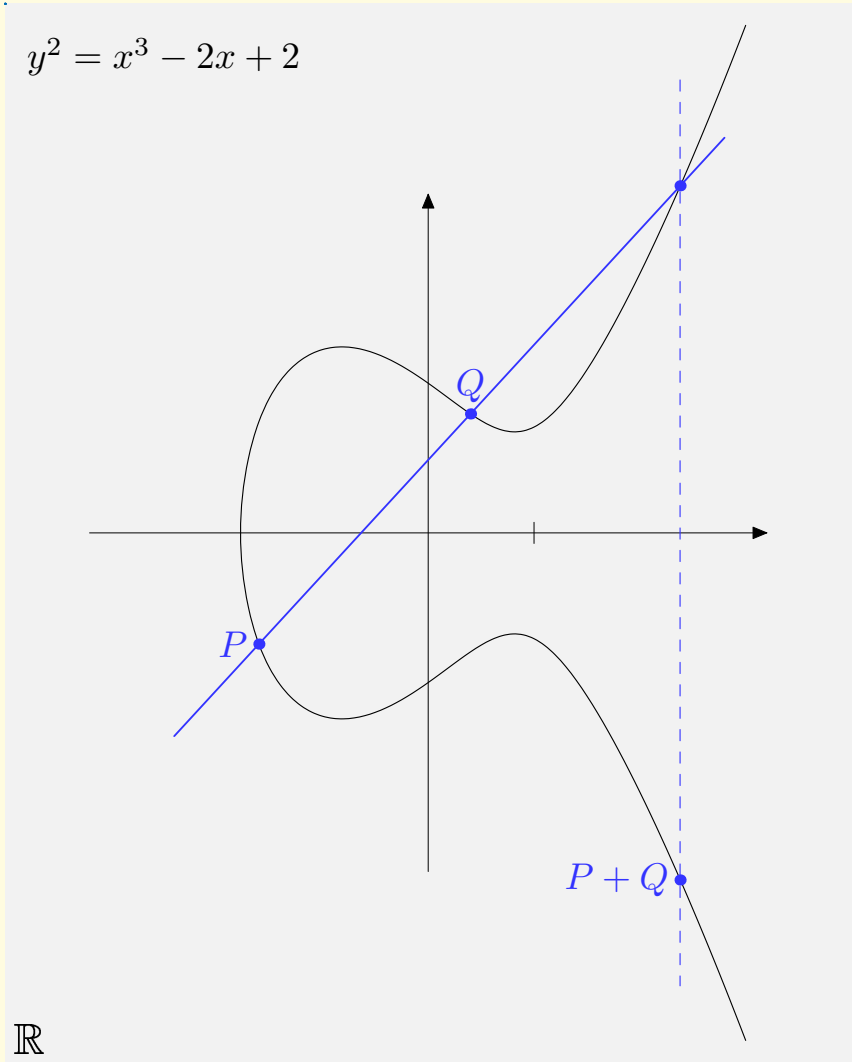


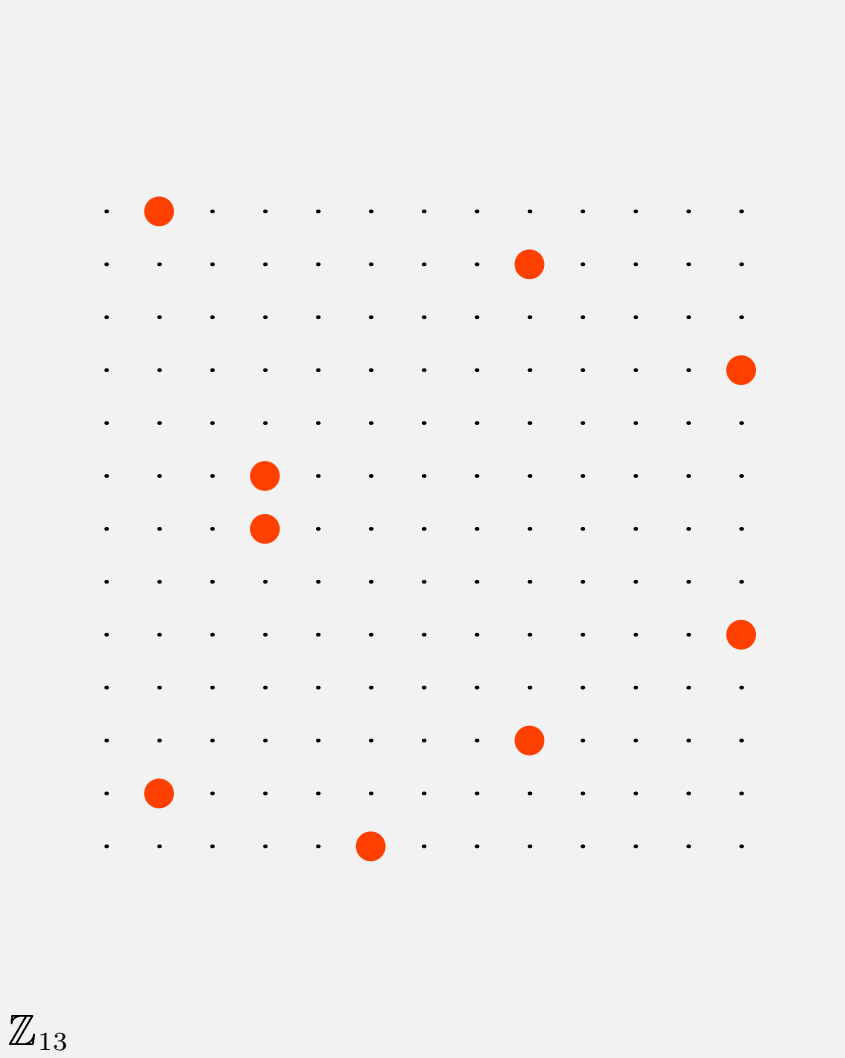
Curve ellittiche



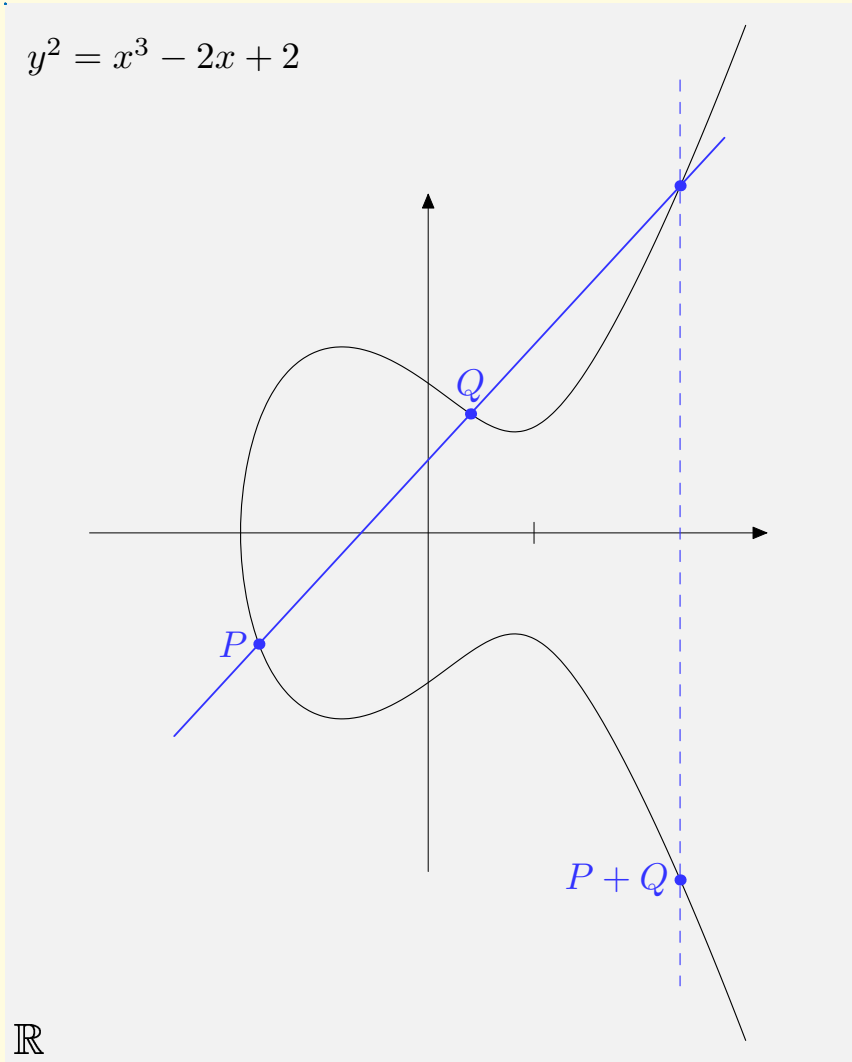


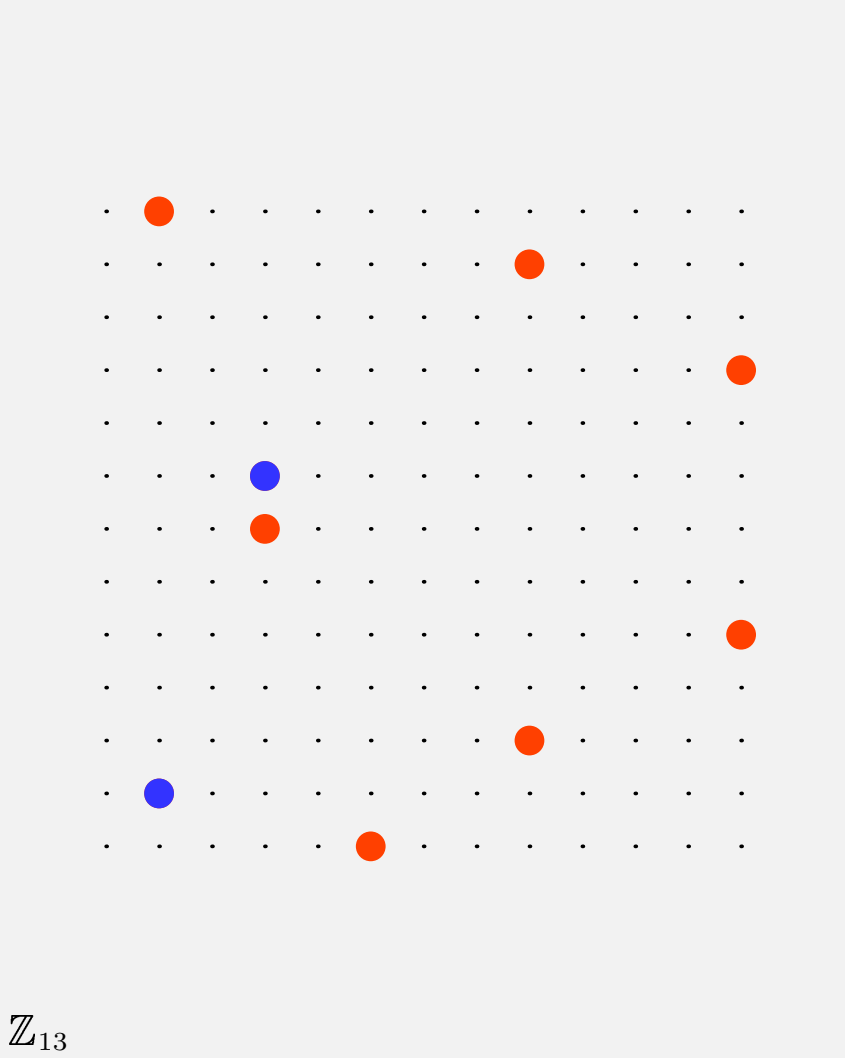
Curve ellittiche



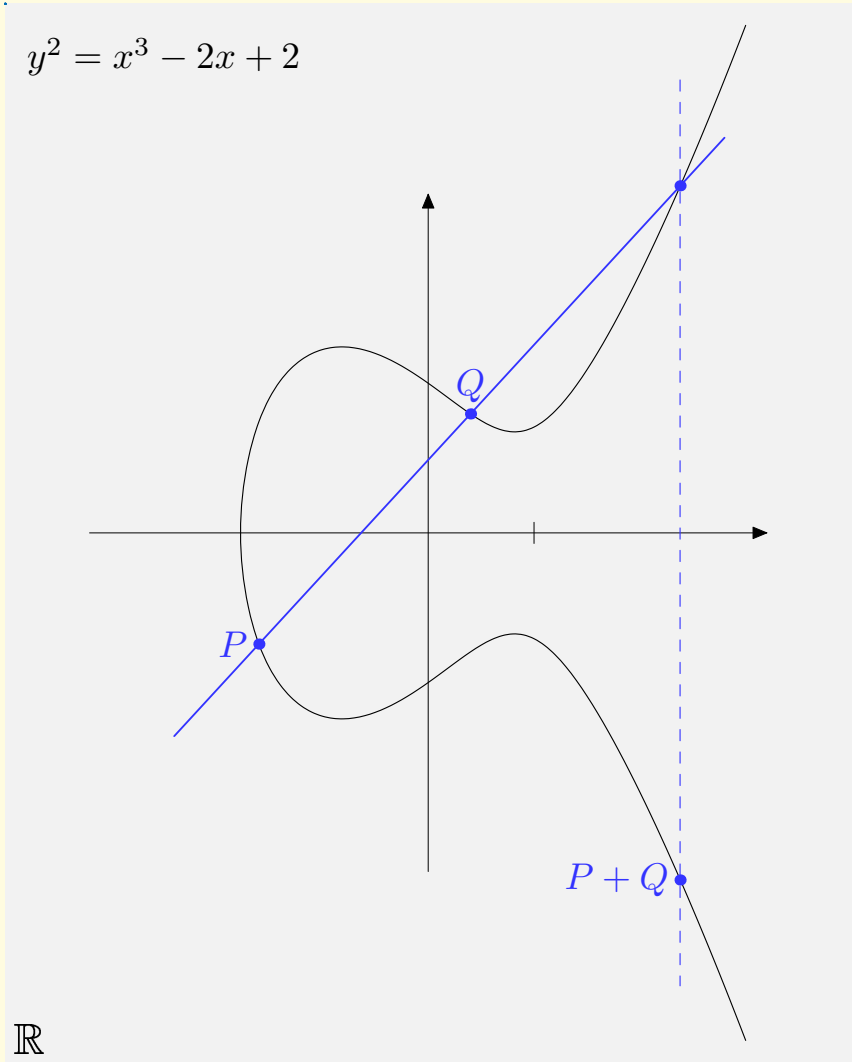


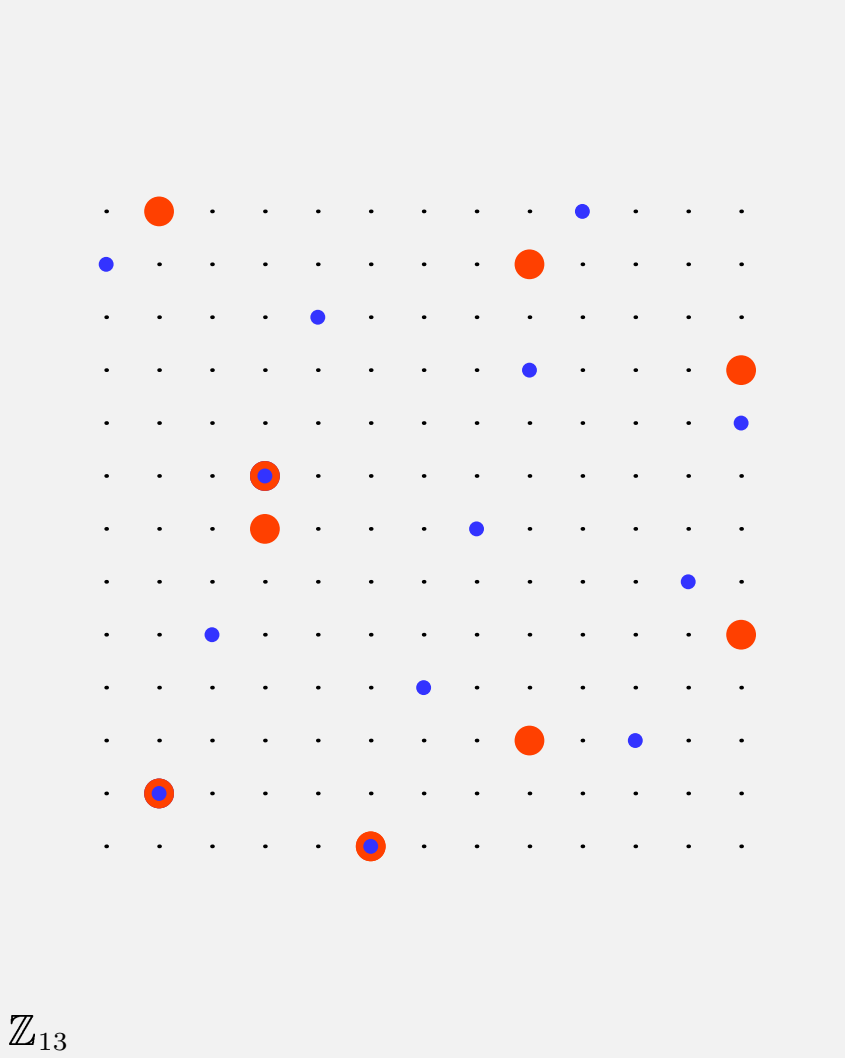
Curve ellittiche



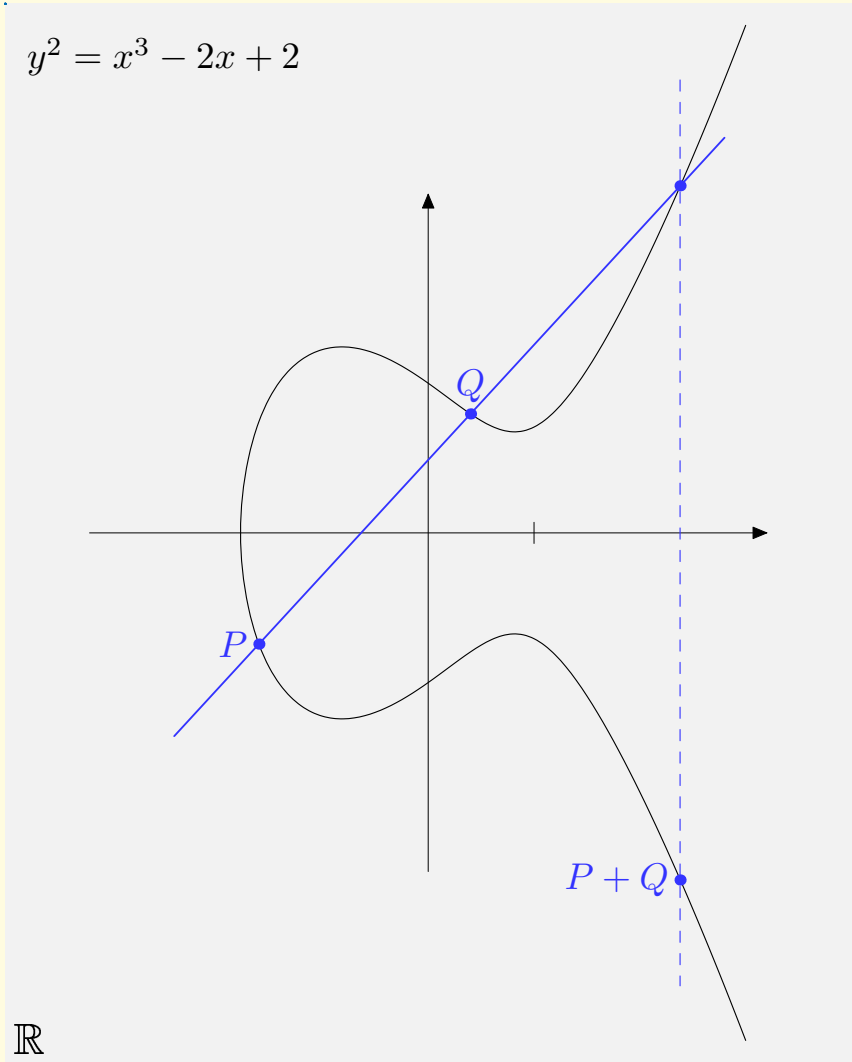


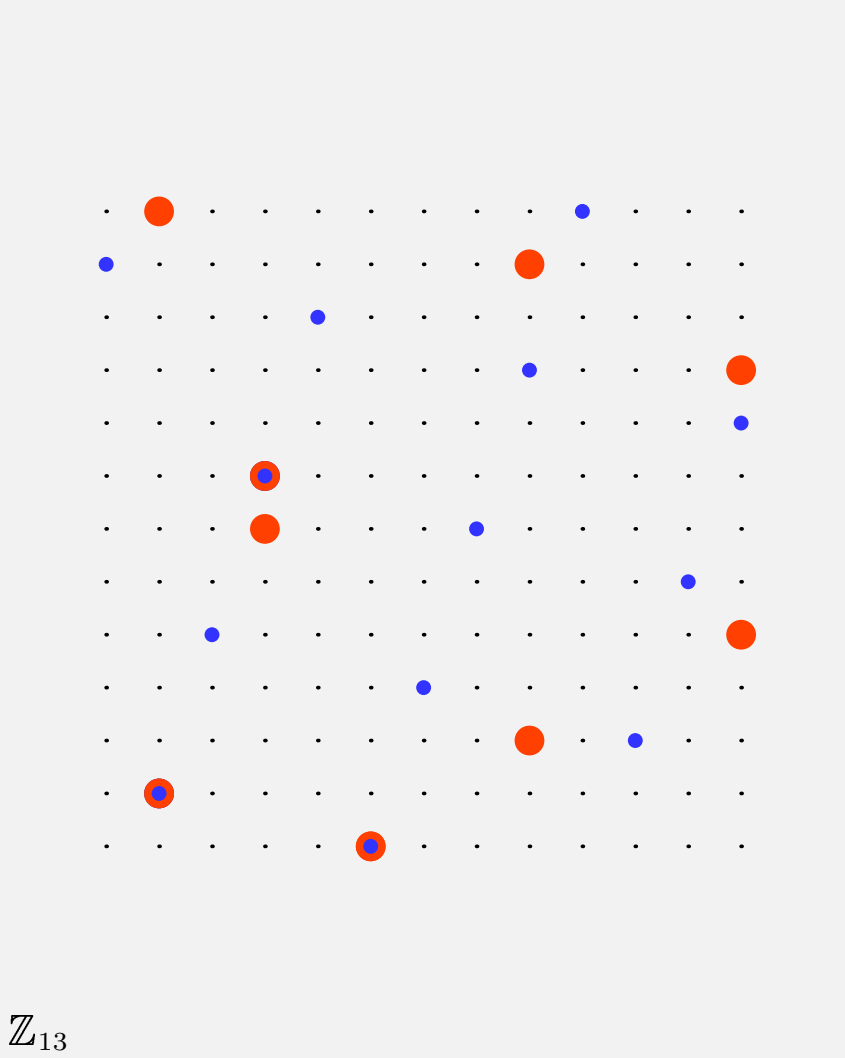
Curve ellittiche





Curve ellittiche





Appendice. Esempio: la cifratura RSA del titolo

Appendice. Esempio: la cifratura RSA del titolo

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.

Appendice. Esempio: la cifratura RSA del titolo

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.

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

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

Appendice. Esempio: la cifratura RSA del titolo

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.

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

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

Codifica del testo: ogni lettera maiuscola 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 per giustapposizione da questio codici:

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

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

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

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"
7765846977658 4	7367653280698 2	3276653267827 3	8484797182657 0	7365

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"
7765846977658 4	7367653280698 2	3276653267827 3	8484797182657 0	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".

Indice