


IL CRITTOSISTEMA RSA

Università del Molise

Facoltà di Economia, Campobasso

Seminario sulla sicurezza informatica

28 Aprile, 2004


I due volti della Crittografia

I due volti della Crittografia

- **Chiave privata (o simmetrica)**
  - Lucifer
  - DES
  - AES

I due volti della Crittografia

- **Chiave privata (o simmetrica)**
  - Lucifer
  - DES
  - AES
- **Chiave pubblica**
  - RSA
  - Diffie–Hellmann
  - Knapsack
  - NTRU

RSA_{2048} = 25195908475657893494027183240048398571429282126204
032027777137836043662020707595556264018525880784406918290641249
515082189298559149176184502808489120072844992687392807287776735
971418347270261896375014971824691165077613379859095700097330459
748808428401797429100642458691817195118746121515172654632282216
869987549182422433637259085141865462043576798423387184774447920
739934236584823824281198163815010674810451660377306056201619676
256133844143603833904414952634432190114657544454178424020924616
515723350778707749817125772467962926386356373289912154831438167
899885040445364023527381951378636564391212010397122822120720357

RSA_{2048} = 25195908475657893494027183240048398571429282126204
032027777137836043662020707595556264018525880784406918290641249
515082189298559149176184502808489120072844992687392807287776735
971418347270261896375014971824691165077613379859095700097330459
748808428401797429100642458691817195118746121515172654632282216
869987549182422433637259085141865462043576798423387184774447920
739934236584823824281198163815010674810451660377306056201619676
256133844143603833904414952634432190114657544454178424020924616
515723350778707749817125772467962926386356373289912154831438167
899885040445364023527381951378636564391212010397122822120720357

RSA_{2048} è un numero con 617 cifre (decimali)

$$RSA_{2048} = 25195908475657893494027183240048398571429282126204$$
$$03202777713783604366202070759555626401852588078440691829064124951508218929855914917618450280848912007284499268739280728777673597141834727026189637501497182469116507761337985909570009733045974880842840179742910064245869181719511874612151517265463228221686998754918242243363725908514186546204357679842338718477444792073993423658482382428119816381501067481045166037730605620161967625613384414360383390441495263443219011465754445417842402092461651572335077870774981712577246796292638635637328991215483143816789988504044536402352738195137863656439121201039712282212072035$$

RSA_{2048} è un numero con 617 cifre (decimali)

http://www.rsa.com/rsalabs/challenges/factoring/numbers.html/

$$RSA_{2048} = p \cdot q, \qquad p, q \approx 10^{308}$$

$$RSA_{2048} = p \cdot q, \qquad p, q \approx 10^{308}$$

PROBLEMA: *Calcolare p e q*

$$RSA_{2048} = p \cdot q, \quad p, q \approx 10^{308}$$

PROBLEMA: *Calcolare p e q*

PREMIO: 200.000 US$ (~ 167,448€)!!

$$RSA_{2048}{=}p \cdot q, \quad p, q \approx 10^{308}$$

PROBLEMA: *Calcolare* p *e* q

PREMIO: 200.000 US$ ($\sim$ 167,448€)!!

Teorema. Sia $a \in \mathbb{N}$. Esistono unici $p_1 < p_2 < \cdots < p_k$ primi tali che $\quad a = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$

$$RSA_{2048}=p\cdot q, \quad p,q \approx 10^{308}$$

PROBLEMA: *Calcolare p e q*

PREMIO: 200.000 US$ (∼ 167,448€)!!

Teorema. Sia $a \in \mathbb{N}$. Esistono unici $p_1 < p_2 < \cdots < p_k$ primi tali che $a = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$

Sfortunatamente: RSAlabs ritiene che per fattorizzare in un anno un numero siano necessari:

numero	computers	memoria
RSA_{1620}	1.6×10^{15}	120 Tb
RSA_{1024}	$342,000,000$	170 Gb
RSA_{760}	215,000	4Gb.

http://www.rsa.com/rsalabs/challenges/factoring/numbers.html

http://www.rsa.com/rsalabs/challenges/factoring/numbers.html

Challenge Number	Premio ($US)
RSA_{576}	\$10,000
RSA_{640}	\$20,000
RSA_{704}	\$30,000
RSA_{768}	\$50,000
RSA_{896}	\$75,000
RSA_{1024}	\$100,000
RSA_{1536}	\$150,000
RSA_{2048}	\$200,000

http://www.rsa.com/rsalabs/challenges/factoring/numbers.html

Challenge Number	Premio ($US)	Stato
RSA_{576}	$10,000	Fattorizzato - Dicembre 2003
RSA_{640}	$20,000	Non Fattorizzato
RSA_{704}	$30,000	Non Fattorizzato
RSA_{768}	$50,000	Non Fattorizzato
RSA_{896}	$75,000	Non Fattorizzato
RSA_{1024}	$100,000	Non Fattorizzato
RSA_{1536}	$150,000	Non Fattorizzato
RSA_{2048}	$200,000	Non Fattorizzato

Storia dell'"Arte del Fattorizzare"

Storia dell'"Arte del Fattorizzare"

- 220 AC in Grecia (Eratostene da Cirene)

Storia dell'"Arte del Fattorizzare"

- 220 AC in Grecia (Eratostene da Cirene)
- 1730 Eulero $2^{2^5} + 1 = 641 \cdot 6700417$

Storia dell'"Arte del Fattorizzare"

- 220 AC in Grecia (Eratostene da Cirene)
- 1730 Eulero $2^{2^5} + 1 = 641 \cdot 6700417$
- 1750–1800 Fermat, Gauss (Crivelli e Tavole)

Storia dell'"Arte del Fattorizzare"

- 220 AC in Grecia (Eratostene da Cirene)
- 1730 Eulero $2^{2^5}+1=641\cdot 6700417$
- 1750–1800 Fermat, Gauss (Crivelli e Tavole)
- 1880 Landry & Le Lasseur:

$$2^{2^6}+1=274177\times 67280421310721$$

Storia dell'"Arte del Fattorizzare"

- 220 AC in Grecia (Eratostene da Cirene)
- 1730 Eulero $2^{2^5}+1=641\cdot 6700417$
- 1750–1800 Fermat, Gauss (Crivelli e Tavole)
- 1880 Landry & Le Lasseur:

$$2^{2^6}+1=274177\times 67280421310721$$

- 1919 Pierre e Eugène Carissan (Macchina per Fattorizzare)

Storia dell'"Arte del Fattorizzare"

- 220 AC in Grecia (Eratostene da Cirene)
- 1730 Eulero $2^{2^5}+1=641\cdot 6700417$
- 1750–1800 Fermat, Gauss (Crivelli e Tavole)
- 1880 Landry & Le Lasseur:

$$2^{2^6}+1=274177\times 67280421310721$$

- 1919 Pierre e Eugène Carissan (Macchina per Fattorizzare)
- 1970 Morrison & Brillhart

$$2^{2^7}+1=59649589127497217\times 5704689200685129054721$$

Storia dell'"Arte del Fattorizzare"

- 220 AC in Grecia (Eratostene da Cirene)
- 1730 Eulero $2^{2^5}+1 = 641 \cdot 6700417$
- 1750–1800 Fermat, Gauss (Crivelli e Tavole)
- 1880 Landry & Le Lasseur:

$$2^{2^6}+1 = 274177 \times 67280421310721$$

- 1919 Pierre e Eugène Carissan (Macchina per Fattorizzare)
- 1970 Morrison & Brillhart

$$2^{2^7}+1 = 59649589127497217 \times 5704689200685129054721$$

- 1982 Crivello quadratico **QS** Pomerance $\rightsquigarrow$ Crivello campo numerico **NFS**

Storia dell'"Arte del Fattorizzare"

- 220 AC in Grecia (Eratostene da Cirene)
- 1730 Eulero $2^{2^5}+1 = 641 \cdot 6700417$
- 1750–1800 Fermat, Gauss (Crivelli e Tavole)
- 1880 Landry & Le Lasseur:

$$2^{2^6}+1 = 274177 \times 67280421310721$$

- 1919 Pierre e Eugène Carissan (Macchina per Fattorizzare)
- 1970 Morrison & Brillhart

$$2^{2^7}+1 = 59649589127497217 \times 5704689200685129054721$$

- 1982 Crivello quadratico **QS** Pomerance $\rightsquigarrow$ Crivello campo numerico **NFS**
- 1987 Fattorizzazione con Curve ellittiche **ECT** (Lenstra)

L' antica macchina per Fattorizzare di Carissan

L' antica macchina per Fattorizzare di Carissan

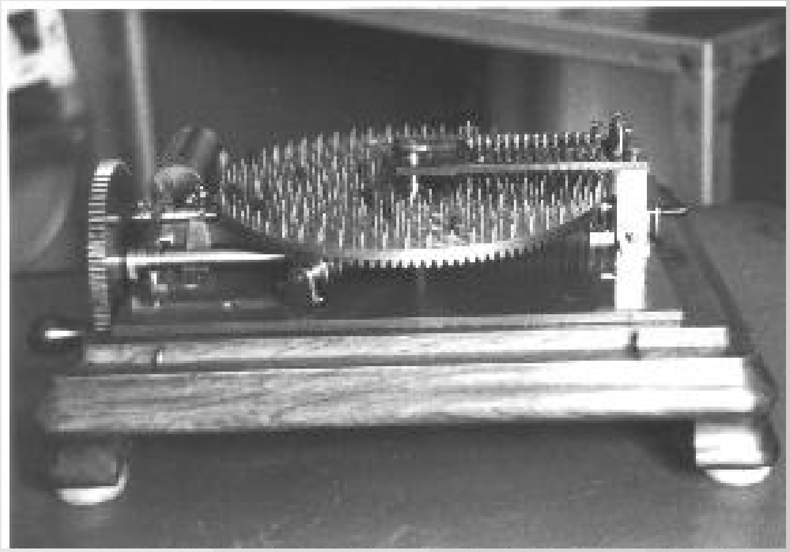


Figura 1: Conservatoire Nationale des Arts et Métiers a Parigi

L' antica macchina per Fattorizzare di Carissan

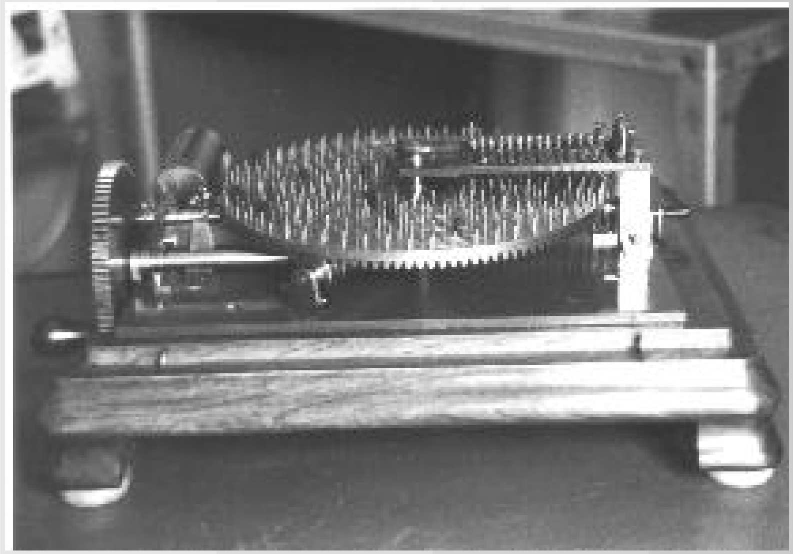


Figura 1: Conservatoire Nationale des Arts et Métiers a Parigi

`http://www.math.uwaterloo.ca/ shallit/Papers/carissan.html`

Figura 2: Tenente Eugène Carissan

Figura 2: Tenente Eugène Carissan

$$225058681 = 229 \times 982789 \qquad 2 \text{ minuti}$$

$$3450315521 = 1409 \times 2418769 \qquad 3 \text{ minuti}$$

$$3570537526921 = 841249 \times 4244329 \qquad 18 \text{ minuti}$$

Fattorizzazione Contemporanea

Fattorizzazione Contemporanea

❶ 1994, Crivello quadratico (QS): (8 mesi, 600 volontari, 20 nazioni)

D.Atkins, M. Graff, A. Lenstra, P. Leyland

$$RSA_{129} = 114381625757888867669235779976146612010218296721242362562561842935706$$
$$935245733897830597123563958705058989075147599290026879543541 =$$
$$= 3490529510847650949147849619903898133417764638493387843990820577 \times$$
$$32769132993266709549961988190834461413177642967992942539798288533$$

Fattorizzazione Contemporanea

❶ 1994, Crivello quadratico (QS): (8 mesi, 600 volontari, 20 nazioni)
D.Atkins, M. Graff, A. Lenstra, P. Leyland

$$RSA_{129} = 114381625757888867669235779976146612010218296721242362562561842935706 935245733897830597123563958705058989075147599290026879543541 =$$
$$= 3490529510847650949147849619903898133417764638493387843990820577 \times 32769132993266709549961988190834461413177642967992942539798288533$$

❷ (2 Febbraio 1999), Crivello campo numerico (NFS): (160 Sun, 4 mesi)

$$RSA_{155} = 109417386415705274218097073220403576120037329454492059909138421314763499842 88934784717997257891267332497625752899781833797076537244027146743531593354333897 =$$
$$= 102639592829741105772054196573991675900716567808038066803341933521790711307779 \times 106603488380168454820927220360012878679207958575989291522270608237193062808643$$

Fattorizzazione Contemporanea

❶ 1994, Crivello quadratico (QS): (8 mesi, 600 volontari, 20 nazioni)
D.Atkins, M. Graff, A. Lenstra, P. Leyland

$$\begin{aligned} RSA_{129} = {} & 114381625757888867669235779976146612010218296721242362562561842935706 \\ & 935245733897830597123563958705058989075147599290026879543541 = \\ = {} & 3490529510847650949147849619903898133417764638493387843990820577 \times \\ & 32769132993266709549961988190834461413177642967992942539798288533 \end{aligned}$$

❷ (2 Febbraio 1999), Crivello campo numerico (NFS): (160 Sun, 4 mesi)

$$\begin{aligned} RSA_{155} = {} & 109417386415705274218097073220403576120037329454492059909138421314763499842 \\ & 88934784717997257891267332497625752899781833797076537244027146743531593354333897 = \\ = {} & 102639592829741105772054196573991675900716567808038066803341933521790711307779 \times \\ & 106603488380168454820927220360012878679207958575989291522270608237193062808643 \end{aligned}$$

❸ (3 Dicembre, 2003) (NFS): J. Franke et al. (174 cifre decimali)

$$\begin{aligned} RSA_{576} = {} & 1881988129206079638386972394616504398071635633794173827007633564229888597152346 \\ & 65485319060606504743045317388011303396716199692321205734031879550656996221305168759307650257059 = \\ = {} & 398075086424064937397125500550386491199064362342526708406385189575946388957261768583317 \times \\ & 472772146107435302536223071973048224632914695302097116459852171130520711256363590397527 \end{aligned}$$

Fattorizzazione Contemporanea

❶ 1994, Crivello quadratico (QS): (8 mesi, 600 volontari, 20 nazioni)
D.Atkins, M. Graff, A. Lenstra, P. Leyland

$$RSA_{129} = 114381625757888867669235779976146612010218296721242362562561842935706 \\ 935245733897830597123563958705058989075147599290026879543541 = \\ = 3490529510847650949147849619903898133417764638493387843990820577 \times \\ 32769132993266709549961988190834461413177642967992942539798288533$$

❷ (2 Febbraio 1999), Crivello campo numerico (NFS): (160 Sun, 4 mesi)

$$RSA_{155} = 109417386415705274218097073220403576120037329454492059909138421314763499842 \\ 88934784717997257891267332497625752899781833797076537244027146743531593354333897 = \\ = 102639592829741105772054196573991675900716567808038066803341933521790711307779 \times \\ 106603488380168454820927220360012878679207958575989291522270608237193062808643$$

❸ (3 Dicembre, 2003) (NFS): J. Franke et al. (174 cifre decimali)

$$RSA_{576} = 1881988129206079638386972394616504398071635633794173827007633564229888597152346 \\ 65485319060606504743045317388011303396716199692321205734031879550656996221305168759307650257059 = \\ = 398075086424064937397125500550386491199064362342526708406385189575946388957261768583317 \times \\ 472772146107435302536223071973048224632914695302097116459852171130520711256363590397527$$

❹ La fattorizzazione con curve ellittiche introdotta da H. Lenstra. È adatta per trovare fattori con meno di 50 cifre decimali (piccoli)

Fattorizzazione Contemporanea

1. 1994, Crivello quadratico (QS): (8 mesi, 600 volontari, 20 nazioni)
   D.Atkins, M. Graff, A. Lenstra, P. Leyland

$$RSA_{129} = 114381625757888867669235779976146612010218296721242362562561842935706\\935245733897830597123563958705058989075147599290026879543541 =\\= 3490529510847650949147849619903898133417764638493387843990820577\times\\32769132993266709549961988190834461413177642967992942539798288533$$

2. (2 Febbraio 1999), Crivello campo numerico (NFS): (160 Sun, 4 mesi)

$$RSA_{155} = 109417386415705274218097073220403576120037329454492059909138421314763499842\\88934784717997257891267332497625752899781833797076537244027146743531593354333897 =\\= 102639592829741105772054196573991675900716567808038066803341933521790711307779\times\\106603488380168454820927220360012878679207958575989291522270608237193062808643$$

3. (3 Dicembre, 2003) (NFS): J. Franke et al. (174 cifre decimali)

$$RSA_{576} = 188198812920607963838697239461650439807163563379417382700763356422988859715234 6\\65485319060606504743045317388011303396716199692321205734031879550656996221305168759307650257059 =\\= 398075086424064937397125500550386491199064362342526708406385189575946388957261768583317\times\\472772146107435302536223071973048224632914695302097116459852171130520711256363590397527$$

4. La fattorizzazione con curve ellittiche introdotta da H. Lenstra. È adatta per trovare fattori con meno di 50 cifre decimali (piccoli)

Tutti: complessità sub–esponenziale

RSA

Adi Shamir, Ron L. Rivest, Leonard Adleman (1978)

Il crittosistema RSA

Il crittosistema RSA

1978 R. L. Rivest, A. Shamir, L. Adleman (Brevetto scaduto nel 1998)

Il crittosistema RSA

1978 R. L. Rivest, A. Shamir, L. Adleman (Brevetto scaduto nel 1998)

Problema: Alice vuole spedire il messaggio $\mathcal{P}$ a Bernardo in modo tale che Carlo non possa leggerlo

Il crittosistema RSA

1978 R. L. Rivest, A. Shamir, L. Adleman (Brevetto scaduto nel 1998)

Problema: Alice vuole spedire il messaggio $\mathcal{P}$ a Bernardo in modo tale che Carlo non possa leggerlo

$$\begin{array}{ccc} A\ (Alice) & \longrightarrow & B\ (Bernardo) \\ & \uparrow & \\ & C\ (Carlo) & \end{array}$$

Il crittosistema RSA

1978 R. L. Rivest, A. Shamir, L. Adleman (Brevetto scaduto nel 1998)

Problema: Alice vuole spedire il messaggio $\mathcal{P}$ a Bernardo in modo tale che Carlo non possa leggerlo

$$\begin{array}{ccc} A\ (Alice) & \longrightarrow & B\ (Bernardo) \\ & \uparrow & \\ & C\ (Carlo) & \end{array}$$

❶

❷

❸

❹

Il crittosistema RSA

1978 R. L. Rivest, A. Shamir, L. Adleman (Brevetto scaduto nel 1998)

Problema: Alice vuole spedire il messaggio $\mathcal{P}$ a Bernardo in modo tale che Carlo non possa leggerlo

$$\begin{array}{ccc} A\ (Alice) & \longrightarrow & B\ (Bernardo) \\ & \uparrow & \\ & C\ (Carlo) & \end{array}$$

❶ Generazione della Chiave — deve farla Bernardo

❷

❸

❹

Il crittosistema RSA

1978 R. L. Rivest, A. Shamir, L. Adleman (Brevetto scaduto nel 1998)

Problema: Alice vuole spedire il messaggio $\mathcal{P}$ a Bernardo in modo tale che Carlo non possa leggerlo

$$\begin{array}{ccc} A\ (Alice) & \longrightarrow & B\ (Bernardo) \\ & \uparrow & \\ & C\ (Carlo) & \end{array}$$

1. Generazione della Chiave — deve farla Bernardo
2. Cifratura — deve farla Alice
3.
4.

Il crittosistema RSA

1978 R. L. Rivest, A. Shamir, L. Adleman (Brevetto scaduto nel 1998)

Problema: Alice vuole spedire il messaggio $\mathcal{P}$ a Bernardo in modo tale che Carlo non possa leggerlo

$$\begin{array}{ccc} A\ (Alice) & \longrightarrow & B\ (Bernardo) \\ & \uparrow & \\ & C\ (Carlo) & \end{array}$$

1. GENERAZIONE DELLA CHIAVE — deve farla Bernardo
2. CIFRATURA — deve farla Alice
3. DECIFRATURA — deve farla Bernardo
4.

Il crittosistema RSA

1978 R. L. Rivest, A. Shamir, L. Adleman (Brevetto scaduto nel 1998)

Problema: Alice vuole spedire il messaggio $\mathcal{P}$ a Bernardo in modo tale che Carlo non possa leggerlo

$$\begin{array}{ccc} A\ (Alice) & \longrightarrow & B\ (Bernardo) \\ & \uparrow & \\ & C\ (Carlo) & \end{array}$$

1. Generazione della Chiave — deve farla Bernardo
2. Cifratura — deve farla Alice
3. Decifratura — deve farla Bernardo
4. Attacco — Carlo vorrebbe farlo

Bernardo: Generazione della Chiave

Bernardo: Generazione della Chiave

✍

✍

✍

✍

✍

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $(p, q \approx 10^{100})$

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $(p, q \approx 10^{100})$
- Calcola $M = p \times q$, $\varphi(M) = (p-1) \times (q-1)$
-
-
-

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $\qquad (p, q \approx 10^{100})$
- Calcola $\quad M = p \times q,\ \varphi(M) = (p-1) \times (q-1)$
- Sceglie un intero e tale che
-
-

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $(p, q \approx 10^{100})$
- Calcola $M = p \times q$, $\varphi(M) = (p-1) \times (q-1)$
- Sceglie un intero e tale che

$$0 \leq e \leq \varphi(M) \text{ e } \gcd(e, \varphi(M)) = 1$$

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $\qquad (p, q \approx 10^{100})$
- Calcola $\quad M = p \times q,\ \varphi(M) = (p-1) \times (q-1)$
- Sceglie un intero e tale che

$$0 \leq e \leq \varphi(M) \ \text{ e } \ \gcd(e, \varphi(M)) = 1$$

NOTA. Se si prende $e = 3$ e $p \equiv q \equiv 2 \bmod 3$

-
-

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $(p, q \approx 10^{100})$
- Calcola $M = p \times q$, $\varphi(M) = (p-1) \times (q-1)$
- Sceglie un intero e tale che

$$0 \le e \le \varphi(M) \;\text{ e }\; \gcd(e, \varphi(M)) = 1$$

Nota. Se si prende $e = 3$ e $p \equiv q \equiv 2 \bmod 3$

Gli esperti raccomandano $e = 2^{16} + 1$

-
-

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $\qquad (p, q \approx 10^{100})$
- Calcola $\quad M = p \times q$, $\varphi(M) = (p-1) \times (q-1)$
- Sceglie un intero e tale che

  $$0 \leq e \leq \varphi(M) \text{ e } \gcd(e, \varphi(M)) = 1$$

  NOTA. Se si prende $e = 3$ e $p \equiv q \equiv 2 \bmod 3$

  Gli esperti raccomandano $e = 2^{16} + 1$
- Calcola l'inverso aritmetico d di e modulo $\varphi(M)$
-

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $\qquad (p, q \approx 10^{100})$
- Calcola $\quad M = p \times q,\ \varphi(M) = (p-1) \times (q-1)$
- Sceglie un intero e tale che

  $$0 \leq e \leq \varphi(M) \ \text{ e } \ \gcd(e, \varphi(M)) = 1$$

  NOTA. Se si prende $e = 3$ e $p \equiv q \equiv 2 \bmod 3$

  Gli esperti raccomandano $e = 2^{16} + 1$
- Calcola l'inverso aritmetico d di e modulo $\varphi(M)$

  (i.e. $d \in \mathbb{N}$ (unico $\leq \varphi(M)$) tale che $e \times d \equiv 1 \pmod{\varphi(M)}$)
-

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $\qquad (p, q \approx 10^{100})$
- Calcola $\quad M = p \times q, \; \varphi(M) = (p-1) \times (q-1)$
- Sceglie un intero e tale che

  $$0 \leq e \leq \varphi(M) \;\text{ e }\; \gcd(e, \varphi(M)) = 1$$

  NOTA. Se si prende $e = 3$ e $p \equiv q \equiv 2 \bmod 3$

  Gli esperti raccomandano $e = 2^{16} + 1$
- Calcola l'inverso aritmetico d di e modulo $\varphi(M)$

  (i.e. $d \in \mathbb{N}$ (unico $\leq \varphi(M)$) tale che $e \times d \equiv 1 \pmod{\varphi(M)}$)
- Pubblica (M, e) **Chiave pubblica** e nasconde **chiave segreta** d

Bernardo: Generazione della Chiave

- Sceglie in modo casuale p e q primi $(p, q \approx 10^{100})$
- Calcola $M = p \times q$, $\varphi(M) = (p-1) \times (q-1)$
- Sceglie un intero e tale che

  $$0 \leq e \leq \varphi(M) \;\text{ e }\; \gcd(e, \varphi(M)) = 1$$

  NOTA. Se si prende $e = 3$ e $p \equiv q \equiv 2 \bmod 3$

  Gli esperti raccomandano $e = 2^{16} + 1$
- Calcola l'inverso aritmetico d di e modulo $\varphi(M)$

  (i.e. $d \in \mathbb{N}$ (unico $\leq \varphi(M)$) tale che $e \times d \equiv 1 \pmod{\varphi(M)}$)
- Pubblica (M, e) **Chiave pubblica** e nasconde **chiave segreta** d

Problema: Come fa Bernardo a fare tutto ciò?- Ci torneremo!

Alice: Cifratura

Alice: Cifratura

Rappresentare il messaggio $\mathcal{P}$ come un elemento di $\mathbb{Z}/M\mathbb{Z}$

Alice: Cifratura

Rappresentare il messaggio $\mathcal{P}$ come un elemento di $\mathbb{Z}/M\mathbb{Z}$

(per esempio) $\boxed{A \leftrightarrow 1 \quad B \leftrightarrow 2 \quad C \leftrightarrow 3 \quad \ldots \quad Z \leftrightarrow 26 \quad AA \leftrightarrow 27 \ldots}$

Alice: Cifratura

Rappresentare il messaggio $\mathcal{P}$ come un elemento di $\mathbb{Z}/M\mathbb{Z}$

(per esempio) $\boxed{A \leftrightarrow 1 \quad B \leftrightarrow 2 \quad C \leftrightarrow 3 \quad \ldots \quad Z \leftrightarrow 26 \quad AA \leftrightarrow 27 \ldots}$

$$\texttt{MOLISE} \leftrightarrow 11 \cdot 26^5 + 13 \cdot 26^4 + 10 \cdot 26^3 + 9 \cdot 26^2 + 17 \cdot 26 + 5 = 136818115$$

Nota. È meglio se il testo non è troppo corto. Altrimenti si fa il *padding*

Alice: Cifratura

Rappresentare il messaggio $\mathcal{P}$ come un elemento di $\mathbb{Z}/M\mathbb{Z}$

(per esempio) $\boxed{A \leftrightarrow 1 \quad B \leftrightarrow 2 \quad C \leftrightarrow 3 \quad \ldots \quad Z \leftrightarrow 26 \quad AA \leftrightarrow 27 \ldots}$

$$\texttt{MOLISE} \leftrightarrow 11 \cdot 26^5 + 13 \cdot 26^4 + 10 \cdot 26^3 + 9 \cdot 26^2 + 17 \cdot 26 + 5 = 136818115$$

Nota. È meglio se il testo non è troppo corto. Altrimenti si fa il *padding*

$$\boxed{\mathcal{C} = E(\mathcal{P}) = \mathcal{P}^e \pmod{M}}$$

Alice: Cifratura

Rappresentare il messaggio $\mathcal{P}$ come un elemento di $\mathbb{Z}/M\mathbb{Z}$

(per esempio) $\boxed{A \leftrightarrow 1 \quad B \leftrightarrow 2 \quad C \leftrightarrow 3 \quad \ldots \quad Z \leftrightarrow 26 \quad AA \leftrightarrow 27 \ldots}$

$$\texttt{MOLISE} \leftrightarrow 11 \cdot 26^5 + 13 \cdot 26^4 + 10 \cdot 26^3 + 9 \cdot 26^2 + 17 \cdot 26 + 5 = 136818115$$

Nota. È meglio se il testo non è troppo corto. Altrimenti si fa il *padding*

$$\boxed{\mathcal{C} = E(\mathcal{P}) = \mathcal{P}^e \pmod{M}}$$

Esempio: $p = 9049465727$, $q = 8789181607$, $M = 79537397720925283289$, $e = 2^{16} + 1 = 65537$, $\mathcal{P} = \texttt{MOLISE}$:

Alice: Cifratura

Rappresentare il messaggio $\mathcal{P}$ come un elemento di $\mathbb{Z}/M\mathbb{Z}$

(per esempio) $\boxed{A \leftrightarrow 1 \quad B \leftrightarrow 2 \quad C \leftrightarrow 3 \quad \ldots \quad Z \leftrightarrow 26 \quad AA \leftrightarrow 27 \ldots}$

$$\texttt{MOLISE} \leftrightarrow 11 \cdot 26^5 + 13 \cdot 26^4 + 10 \cdot 26^3 + 9 \cdot 26^2 + 17 \cdot 26 + 5 = 136818115$$

Nota. È meglio se il testo non è troppo corto. Altrimenti si fa il *padding*

$$\boxed{\mathcal{C} = E(\mathcal{P}) = \mathcal{P}^e \pmod{M}}$$

Esempio: $p = 9049465727$, $q = 8789181607$, $M = 79537397720925283289$, $e = 2^{16} + 1 = 65537$, $\mathcal{P} = \texttt{MOLISE}$:

$$E(\texttt{MOLISE}) = 136818115^{65537} \pmod{79537397720925283289}$$

$$= 53971574720895588999 = \mathcal{C} = \texttt{UUDPHSGXBATNSU}$$

Bernardo: Decifratura

Bernardo: Decifratura

$$\boxed{\mathcal{P} = D(\mathcal{C}) = \mathcal{C}^d \pmod{M}}$$

Bernardo: Decifratura

$$\boxed{\mathcal{P} = D(\mathcal{C}) = \mathcal{C}^d \pmod{M}}$$

Nota. Bernardo può decifrare perchè è l'unico che conosce d.

Bernardo: Decifratura

$$\mathcal{P} = D(\mathcal{C}) = \mathcal{C}^d \pmod{M}$$

Nota. Bernardo può decifrare perchè è l'unico che conosce d.

Teorema di Eulero. Se $a, m \in \mathbb{N}$, $\gcd(a, m) = 1$,

$$a^{\varphi(m)} \equiv 1 \pmod{m}.$$

Se $n_1 \equiv n_2 \bmod \varphi(m)$ then $a^{n_1} \equiv a^{n_2} \bmod m$.

Bernardo: Decifratura

$$\boxed{\mathcal{P} = D(\mathcal{C}) = \mathcal{C}^d \pmod{M}}$$

Nota. Bernardo può decifrare perchè è l'unico che conosce d.

Teorema di Eulero. Se $a, m \in \mathbb{N}$, $\gcd(a, m) = 1$,

$$a^{\varphi(m)} \equiv 1 \pmod{m}.$$

Se $n_1 \equiv n_2 \bmod \varphi(m)$ then $a^{n_1} \equiv a^{n_2} \bmod m$.

Quindi $(ed \equiv 1 \bmod \varphi(M))$

$$\boxed{D(E(\mathcal{P})) = \mathcal{P}^{ed} \equiv \mathcal{P} \bmod M}$$

Bernardo: Decifratura

$$\boxed{\mathcal{P} = D(\mathcal{C}) = \mathcal{C}^d \pmod{M}}$$

Nota. Bernardo può decifrare perchè è l'unico che conosce d.

Teorema di Eulero. Se $a, m \in \mathbb{N}$, $\gcd(a, m) = 1$,

$$a^{\varphi(m)} \equiv 1 \pmod{m}.$$

Se $n_1 \equiv n_2 \bmod \varphi(m)$ then $a^{n_1} \equiv a^{n_2} \bmod m$.

Quindi $(ed \equiv 1 \bmod \varphi(M))$

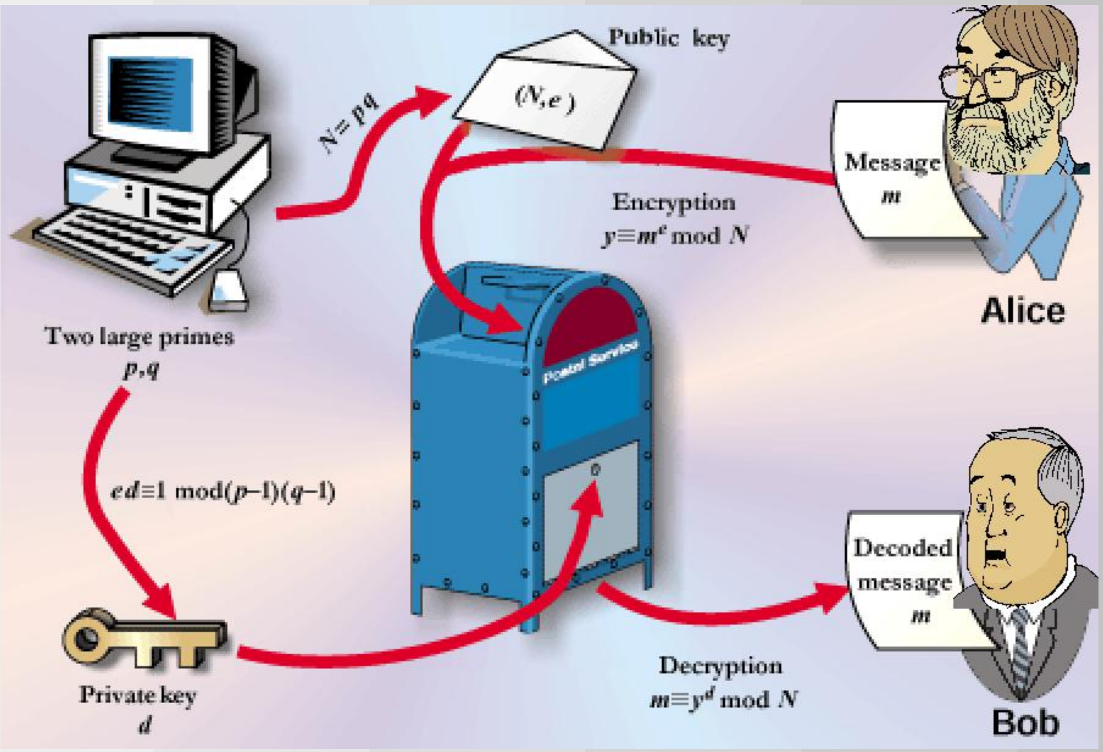
$$\boxed{D(E(\mathcal{P})) = \mathcal{P}^{ed} \equiv \mathcal{P} \bmod M}$$

Esempio(cont.):$d = 65537^{-1} \bmod \varphi(9049465727 \cdot 8789181607) = 57173914060643780153$

$D(\texttt{UUDPHSGXBATNSU}) =$

$53971574720895588999^{57173914060643780153} (\bmod 79537397720925283289) =$ `MOLISE`

RSA in funzione



L'algoritmo dei quadrati successivi

L'algoritmo dei quadrati successivi

Problema: Come si calcola $a^b \bmod c$?

L'algoritmo dei quadrati successivi

Problema: Come si calcola $a^b \bmod c$?

$$562594241074019544 43^{5717391406064378015 3}$$

$$56259424107401954443^{57173914060643780153} \pmod{79537397720925283289}$$

L'algoritmo dei quadrati successivi

Problema: Come si calcola $a^b \bmod c$?

$$5625942410740195444 3^{5717391406064378015 3} \pmod{7953739772092528328 9}$$

✍

✍

✍

L'algoritmo dei quadrati successivi

Problema: Come si calcola $a^b \bmod c$?

$$5625942410740195444 3^{5717391406064378015 3} \pmod{79537397720925283289}$$

- ✍ <u>Calcolare l'espansione binaria</u> $b = \sum_{j=0}^{[\log_2 b]} \epsilon_j 2^j$
- ✍
- ✍

L'algoritmo dei quadrati successivi

Problema: Come si calcola $a^b \bmod c$?

$$562594241074019544443^{571739140606437801 53}\pmod{795373977209252832 89}$$

✍ Calcolare l'espansione binaria $b = \sum_{j=0}^{[\log_2 b]} \epsilon_j 2^j$

$57173914060643780153 = 110001100101110010100010111110101011110011011000100100011000111001$

✍

✍

L'algoritmo dei quadrati successivi

Problema: Come si calcola $a^b \bmod c$?

$$562594241074019544 43^{57173914060643780153} \pmod{79537397720925283289}$$

- Calcolare l'espansione binaria $b = \sum_{j=0}^{[\log_2 b]} \epsilon_j 2^j$

$57173914060643780153 = 110001100101110010100010111110101011110011011000100100011000111001$

- Calcolare ricorsivamente $a^{2^j} \bmod c, j = 1, \ldots, [\log_2 b]$:

-

L'algoritmo dei quadrati successivi

Problema: Come si calcola $a^b \bmod c$?

$$562594241074019544 43^{5717391406064378015 3} \pmod{7953739772092528328 9}$$

- <u>Calcolare l'espansione binaria</u> $b = \sum_{j=0}^{[\log_2 b]} \epsilon_j 2^j$

  $5717391406064378015 3 = 110001100101110010100010111110101011110011011000100100011000111001$

- <u>Calcolare ricorsivamente</u> $a^{2^j} \bmod c, j = 1, \ldots, [\log_2 b]$:

  $$a^{2^j} \bmod c = \left(a^{2^{j-1}} \bmod c\right)^2 \bmod c$$

-

L'algoritmo dei quadrati successivi

Problema: Come si calcola $a^b \bmod c$?

$$562594241074019544 43^{5717391406064378 0153} \pmod{7953739772092528 3289}$$

- <u>Calcolare l'espansione binaria</u> $b = \sum_{j=0}^{[\log_2 b]} \epsilon_j 2^j$

57173914060643780153=1100011001011100101000101111101010111100110110001001000110001110 01

- <u>Calcolare ricorsivamente</u> $a^{2^j} \bmod c, j = 1, \ldots, [\log_2 b]$:

$$a^{2^j} \bmod c = \left(a^{2^{j-1}} \bmod c\right)^2 \bmod c$$

- <u>Moltiplicare</u> $a^{2^j} \bmod c$ con $\epsilon_j = 1$

L'algoritmo dei quadrati successivi

Problema: Come si calcola $a^b \bmod c$?

$$562594241074019544443^{5717391406064378015 3}(\bmod 795373977209252832 89)$$

- <u>Calcolare l'espansione binaria</u> $b = \sum_{j=0}^{[\log_2 b]} \epsilon_j 2^j$

  $5717391406064378015 3 = 110001100101110010100010111110101011110011011000100100011000111001$

- <u>Calcolare ricorsivamente</u> $a^{2^j} \bmod c, j = 1, \ldots, [\log_2 b]$:

  $$a^{2^j} \bmod c = \left(a^{2^{j-1}} \bmod c\right)^2 \bmod c$$

- <u>Moltiplicare</u> $a^{2^j} \bmod c$ con $\epsilon_j = 1$

  $$a^b \bmod c = \left(\prod_{j=0,\epsilon_j=1}^{[\log_2 b]} a^{2^j} \bmod c\right) \bmod c$$

$\#\{$**oper. in** $\mathbb{Z}/c\mathbb{Z}$ **per calcolare** $a^b \bmod c\} \leq 2\log_2 b$

$\#\{\textbf{oper. in } \mathbb{Z}/c\mathbb{Z} \textbf{ per calcolare } a^b \bmod c\} \le 2\log_2 b$

`QULMHYHXQRBNQV` si decifra con 131 operazioni in

$$\mathbb{Z}/795373977209252832 89\mathbb{Z}$$

$\#\{$**oper. in** $\mathbb{Z}/c\mathbb{Z}$ **per calcolare** $a^b \bmod c\} \leq 2\log_2 b$

`QULMHYHXQRBNQV` si decifra con 131 operazioni in

$$\mathbb{Z}/79537397720925283289\mathbb{Z}$$

Pseudo codice: $e_c(a,b) = a^b \bmod c$

$\#\{$**oper. in** $\mathbb{Z}/c\mathbb{Z}$ **per calcolare** $a^b \bmod c\} \leq 2\log_2 b$

`QULMHYHXQRBNQV` si decifra con 131 operazioni in

$$\mathbb{Z}/795373977209252832 89\mathbb{Z}$$

PSEUDO CODICE: $e_c(a,b) = a^b \bmod c$

$e_c(a,b)$	`=`	`if`	$b=1$	`then`	$a \bmod c$
		`if`	$2 \mid b$	`then`	$e_c(a, \frac{b}{2})^2 \bmod c$
		`else`			$a * e_c(a, \frac{b-1}{2})^2 \bmod c$

$\#\{$**oper. in** $\mathbb{Z}/c\mathbb{Z}$ **per calcolare** $a^b \bmod c\} \leq 2\log_2 b$

`QULMHYHXQRBNQV` si decifra con 131 operazioni in

$$\mathbb{Z}/795373977209252832 89\mathbb{Z}$$

PSEUDO CODICE: $e_c(a,b) = a^b \bmod c$

$$
\boxed{\begin{array}{llllll}
e_c(a,b) & \texttt{=} & \texttt{if} & b=1 & \texttt{then} & a \bmod c \\
 & & \texttt{if} & 2|b & \texttt{then} & e_c(a,\frac{b}{2})^2 \bmod c \\
 & & \texttt{else} & & & a * e_c(a,\frac{b-1}{2})^2 \bmod c
\end{array}}
$$

Per cifrare con $e = 2^{16}+1$, sono sufficienti solo 17 operazioni in $\mathbb{Z}/M\mathbb{Z}$.

Generazione della Chiave

Generazione della Chiave

Problema. Produrre un numero primo $p \approx 10^{100}$ in modo casuale

	Algoritmo Probabilistico (di tipo Las Vegas)
1.	Let $p = \textsc{Random}(10^{100})$
2.	If isprimo(p)=1 then Output=p else goto 1

Generazione della Chiave

Problema. Produrre un numero primo $p \approx 10^{100}$ in modo casuale

	`Algoritmo Probabilistico (di tipo Las Vegas)`
1.	`Let` $p = \textsc{Random}(10^{100})$
2.	`If ISPRIMO(`p`)=1 then OUTPUT=`p` else goto 1`

sotto–problemi:

Generazione della Chiave

Problema. Produrre un numero primo $p \approx 10^{100}$ in modo casuale

```
     Algoritmo Probabilistico (di tipo Las Vegas)
1.   Let p = RANDOM(10^100)
2.   If ISPRIMO(p)=1 then OUTPUT=p else goto 1
```

sotto–problemi:

A. Quante iterazioni sono necessarie?

(i.e. Come sono distribuiti i numeri primi?)

Generazione della Chiave

Problema. Produrre un numero primo $p \approx 10^{100}$ in modo casuale

	Algoritmo Probabilistico (di tipo Las Vegas)
1.	Let $p = \textsc{Random}(10^{100})$
2.	If ISPRIMO(p)=1 then OUTPUT=p else goto 1

sotto–problemi:

A. Quante iterazioni sono necessarie?

(i.e. Come sono distribuiti i numeri primi?)

B. Come si controlla se p è un primo?

(i.e. come si calcola la funzione ISPRIMO(p)?) $\rightsquigarrow$ Test di Primalità

Generazione della Chiave

Problema. Produrre un numero primo $p \approx 10^{100}$ in modo casuale

Algoritmo Probabilistico (di tipo Las Vegas)
1. Let $p = \textsc{Random}(10^{100})$
2. If ISPRIMO(p)=1 then OUTPUT=p else goto 1

sotto–problemi:

A. Quante iterazioni sono necessarie?

(i.e. Come sono distribuiti i numeri primi?)

B. Come si controlla se p è un primo?

(i.e. come si calcola la funzione ISPRIMO(p)?) $\rightsquigarrow$ Test di Primalità

Falsa Leggenda Metropolitana: Controllare la primalità è equivalente a fattorizzare

A. Distribuzione dei numeri primi

A. Distribuzione dei numeri primi

$$\pi(x) = \#\{p \leq x \text{ t. c. } p \text{ è primo}\}$$

A. Distribuzione dei numeri primi

$$\pi(x) = \#\{p \leq x \text{ t. c. } p \text{ è primo}\}$$

Teorema. (Hadamard - de la vallee Pussen - 1897)

$$\pi(x) \sim \frac{x}{\log x}$$

A. Distribuzione dei numeri primi

$$\pi(x) = \#\{p \leq x \text{ t. c. } p \text{ è primo}\}$$

Teorema. (Hadamard - de la vallee Pussen - 1897)

$$\pi(x) \sim \frac{x}{\log x}$$

Versione quantitativa:

Teorema. (Rosser - Schoenfeld) se $x \geq 67$

$$\frac{x}{\log x - 1/2} < \pi(x) < \frac{x}{\log x - 3/2}$$

A. Distribuzione dei numeri primi

$$\pi(x) = \#\{p \leq x \text{ t. c. } p \text{ è primo}\}$$

Teorema. (Hadamard - de la vallee Pussen - 1897)

$$\pi(x) \sim \frac{x}{\log x}$$

Versione quantitativa:

Teorema. (Rosser - Schoenfeld) se $x \geq 67$

$$\frac{x}{\log x - 1/2} < \pi(x) < \frac{x}{\log x - 3/2}$$

Quindi

$$0.0043523959267 < Prob\left((\textsc{Random}(10^{100}) = \texttt{primo}\right) < 0.004371422086$$

Se P_k denota la probabilità che tra k numeri random $\leq 10^{100}$ ce ne sia uno primo, allora

Se P_k denota la probabilità che tra k numeri random $\leq 10^{100}$ ce ne sia uno primo, allora

$$P_k = 1 - \left(1 - \frac{\pi(10^{100})}{10^{100}}\right)^k$$

Se P_k denota la probabilità che tra k numeri random $\leq 10^{100}$ ce ne sia uno primo, allora

$$P_k = 1 - \left(1 - \frac{\pi(10^{100})}{10^{100}}\right)^k$$

Quindi

$$0.663942 < P_{250} < 0.66554440$$

Se P_k denota la probabilità che tra k numeri random $\leq 10^{100}$ ce ne sia uno primo, allora

$$P_k = 1 - \left(1 - \frac{\pi(10^{100})}{10^{100}}\right)^k$$

Quindi

$$0.663942 < P_{250} < 0.66554440$$

Per accelerare il processo: uno può considerare solo numeri dispari non divisibili per 3 o per 5.

Se P_k denota la probabilità che tra k numeri random $\leq 10^{100}$ ce ne sia uno primo, allora

$$P_k = 1 - \left(1 - \frac{\pi(10^{100})}{10^{100}}\right)^k$$

Quindi

$$0.663942 < P_{250} < 0.66554440$$

Per accelerare il processo: uno può considerare solo numeri dispari non divisibili per 3 o per 5.

Sia

$$\Psi(x, 30) = \#\left\{n \leq x \text{ tali che} \gcd(n, 30) = 1\right\}$$

Per accelerare il processo: uno può considerare solo numeri dispari non divisibili per 3 o per 5.

Per accelerare il processo: uno può considerare solo numeri dispari non divisibili per 3 o per 5.

Sia

$$\Psi(x, 30) = \#\{n \leq x \text{ tali che} \gcd(n, 30) = 1\}$$

allora

Per accelerare il processo: uno può considerare solo numeri dispari non divisibili per 3 o per 5.

Sia

$$\Psi(x,30) = \#\{n \le x \text{ tali che} \gcd(n,30)=1\}$$

allora

$$\frac{4}{15}x - 4 < \Psi(x,30) < \frac{4}{15}x + 4$$

Per accelerare il processo: uno può considerare solo numeri dispari non divisibili per 3 o per 5.

Sia

$$\Psi(x, 30) = \#\left\{n \leq x \text{ tali che} \gcd(n, 30) = 1\right\}$$

allora

$$\frac{4}{15}x - 4 < \Psi(x, 30) < \frac{4}{15}x + 4$$

Quindi, se P'_k indica la probabilità che tra k numeri random coprimi con 30, ce ne sia uno primo, allora

Per accelerare il processo: uno può considerare solo numeri dispari non divisibili per 3 o per 5.

Sia

$$\Psi(x, 30) = \#\{n \leq x \text{ tali che} \gcd(n, 30) = 1\}$$

allora

$$\frac{4}{15}x - 4 < \Psi(x, 30) < \frac{4}{15}x + 4$$

Quindi, se P'_k indica la probabilità che tra k numeri random coprimi con 30, ce ne sia uno primo, allora

$$P'_k = 1 - \left(1 - \frac{\pi(10^{100})}{\Psi(10^{100}, 30)}\right)^k$$

Per accelerare il processo: uno può considerare solo numeri dispari non divisibili per 3 o per 5.

Sia

$$\Psi(x, 30) = \#\{n \leq x \text{ tali che} \gcd(n, 30) = 1\}$$

allora

$$\frac{4}{15}x - 4 < \Psi(x, 30) < \frac{4}{15}x + 4$$

Quindi, se P'_k indica la probabilità che tra k numeri random coprimi con 30, ce ne sia uno primo, allora

$$P'_k = 1 - \left(1 - \frac{\pi(10^{100})}{\Psi(10^{100}, 30)}\right)^k$$

e

$$0.98365832 < P'_{250} < 0.98395199$$

B. Test di primalità

B. Test di primalità

Piccolo Teorema di Fermat. Se p è primo, $p \nmid a \in \mathbb{N}$

$$a^{p-1} \equiv 1 \bmod p$$

B. Test di primalità

Piccolo Teorema di Fermat. Se p è primo, $p \nmid a \in \mathbb{N}$

$$a^{p-1} \equiv 1 \bmod p$$

Test di NON-primalità

$$M \in \mathbb{Z}, \quad 2^{M-1} \not\equiv 1 \bmod M \Rightarrow \quad M \text{composto!}$$

B. Test di primalità

Piccolo Teorema di Fermat. Se p è primo, $p \nmid a \in \mathbb{N}$

$$a^{p-1} \equiv 1 \bmod p$$

Test di NON-primalità

$$M \in \mathbb{Z}, \quad 2^{M-1} \not\equiv 1 \bmod M \Longrightarrow \quad M \text{composto!}$$

Esempio: $2^{RSA_{2048}-1} \not\equiv 1 \bmod RSA_{2048}$

Quindi RSA_{2048} è composto!

B. Test di primalità

Piccolo Teorema di Fermat. Se p è primo, $p \nmid a \in \mathbb{N}$

$$a^{p-1} \equiv 1 \bmod p$$

Test di NON-primalità

$$M \in \mathbb{Z}, \quad 2^{M-1} \not\equiv 1 \bmod M \Rightarrow \quad M \text{composto!}$$

Esempio: $2^{RSA_{2048}-1} \not\equiv 1 \bmod RSA_{2048}$

Quindi RSA_{2048} è composto!

Il piccolo Teorema di Fermat non si inverte. In fatti

B. Test di primalità

Piccolo Teorema di Fermat. Se p è primo, $p \nmid a \in \mathbb{N}$

$$a^{p-1} \equiv 1 \bmod p$$

Test di NON-primalità

$$M \in \mathbb{Z}, \quad 2^{M-1} \not\equiv 1 \bmod M \Longrightarrow \quad M \text{composto!}$$

Esempio: $2^{RSA_{2048}-1} \not\equiv 1 \bmod RSA_{2048}$

Quindi RSA_{2048} è composto!

Il piccolo Teorema di Fermat non si inverte. In fatti

$$2^{93960} \equiv 1 \pmod{93961} \quad \text{but} \quad 93961 = 7 \times 31 \times 433$$

Pseudo primi forti

Pseudo primi forti

Da ora in poi sia $m \equiv 3 \bmod 4$ (solo per semplificare la notazione)

Pseudo primi forti

Da ora in poi sia $m \equiv 3 \bmod 4$ (solo per semplificare la notazione)

Definizione. $m \in \mathbb{N}$, $m \equiv 3 \bmod 4$, composto è detto presudo primo forte (SPSP) in base a se

$$a^{(m-1)/2} \equiv \pm 1 \pmod{m}.$$

Pseudo primi forti

Da ora in poi sia $m \equiv 3 \bmod 4$ (solo per semplificare la notazione)

Definizione. $m \in \mathbb{N}$, $m \equiv 3 \bmod 4$, composto è detto presudo primo forte (SPSP) in base a se

$$a^{(m-1)/2} \equiv \pm 1 \pmod{m}.$$

Nota. Se $p > 2$ primo $\implies$ $a^{(p-1)/2} \equiv \pm 1 \pmod{p}$

Sia $\boxed{\mathcal{S} = \{a \in \mathbb{Z}/m\mathbb{Z} \text{ tale che } \gcd(m,a) = 1, a^{(m-1)/2} \equiv \pm 1 \pmod{m}\}}$

Pseudo primi forti

Da ora in poi sia $m \equiv 3 \bmod 4$ (solo per semplificare la notazione)

Definizione. $m \in \mathbb{N}$, $m \equiv 3 \bmod 4$, composto è detto presudo primo forte (SPSP) in base a se

$$a^{(m-1)/2} \equiv \pm 1 \pmod{m}.$$

Nota. Se $p > 2$ primo $\Longrightarrow \quad a^{(p-1)/2} \equiv \pm 1 \pmod{p}$

Sia $\boxed{\mathcal{S} = \{a \in \mathbb{Z}/m\mathbb{Z} \text{ tale che } \gcd(m,a) = 1, a^{(m-1)/2} \equiv \pm 1 \pmod{m}\}}$

①

②

③

④

Pseudo primi forti

Da ora in poi sia $m \equiv 3 \bmod 4$ (solo per semplificare la notazione)

Definizione. $m \in \mathbb{N}$, $m \equiv 3 \bmod 4$, composto è detto presudo primo forte (SPSP) in base a se

$$a^{(m-1)/2} \equiv \pm 1 \pmod{m}.$$

Nota. Se $p > 2$ primo $\Longrightarrow \quad a^{(p-1)/2} \equiv \pm 1 \pmod{p}$

Sia $\boxed{\mathcal{S} = \{a \in \mathbb{Z}/m\mathbb{Z} \text{ tale che } \gcd(m, a) = 1, a^{(m-1)/2} \equiv \pm 1 \pmod{m}\}}$

① $\mathcal{S} \subseteq (\mathbb{Z}/m\mathbb{Z})^*$ è un sottogruppo

②

③

④

Pseudo primi forti

Da ora in poi sia $m \equiv 3 \bmod 4$ (solo per semplificare la notazione)

Definizione. $m \in \mathbb{N}$, $m \equiv 3 \bmod 4$, composto è detto presudo primo forte (SPSP) in base a se

$$a^{(m-1)/2} \equiv \pm 1 \pmod{m}.$$

Nota. Se $p > 2$ primo $\Longrightarrow \quad a^{(p-1)/2} \equiv \pm 1 \pmod{p}$

Sia $\boxed{\mathcal{S} = \{a \in \mathbb{Z}/m\mathbb{Z} \text{ tale che } \gcd(m,a) = 1, a^{(m-1)/2} \equiv \pm 1 \pmod{m}\}}$

① $\mathcal{S} \subseteq (\mathbb{Z}/m\mathbb{Z})^*$ è un sottogruppo

② Se m è composto $\Longrightarrow$ è un sottogruppo proprio

③

④

Pseudo primi forti

Da ora in poi sia $m \equiv 3 \bmod 4$ (solo per semplificare la notazione)

Definizione. $m \in \mathbb{N}$, $m \equiv 3 \bmod 4$, composto è detto presudo primo forte (SPSP) in base a se

$$a^{(m-1)/2} \equiv \pm 1 \pmod{m}.$$

Nota. Se $p > 2$ primo $\Longrightarrow$ $a^{(p-1)/2} \equiv \pm 1 \pmod{p}$

Sia $\boxed{\mathcal{S} = \{a \in \mathbb{Z}/m\mathbb{Z} \text{ tale che } \gcd(m, a) = 1, a^{(m-1)/2} \equiv \pm 1 \pmod{m}\}}$

① $\mathcal{S} \subseteq (\mathbb{Z}/m\mathbb{Z})^*$ è un sottogruppo

② Se m è composto $\Longrightarrow$ è un sottogruppo proprio

③ Se m è composto $\Longrightarrow$ $\#\mathcal{S} \leq \frac{\varphi(m)}{4}$

④

Pseudo primi forti

Da ora in poi sia $m \equiv 3 \bmod 4$ (solo per semplificare la notazione)

Definizione. $m \in \mathbb{N}$, $m \equiv 3 \bmod 4$, composto è detto presudo primo forte (SPSP) in base a se

$$a^{(m-1)/2} \equiv \pm 1 \pmod{m}.$$

Nota. Se $p > 2$ primo $\Longrightarrow$ $a^{(p-1)/2} \equiv \pm 1 \pmod{p}$

Sia $\boxed{\mathcal{S} = \{a \in \mathbb{Z}/m\mathbb{Z} \text{ tale che } \gcd(m,a) = 1, a^{(m-1)/2} \equiv \pm 1 \pmod{m}\}}$

① $\mathcal{S} \subseteq (\mathbb{Z}/m\mathbb{Z})^*$ è un sottogruppo

② Se m è composto $\Longrightarrow$ è un sottogruppo proprio

③ Se m è composto $\Longrightarrow$ $\#\mathcal{S} \leq \frac{\varphi(m)}{4}$

④ Se m è composto $\Longrightarrow$ $Prob(m \text{ PSPF in base } a) \leq 0,25$

Test di primalità di Miller–Rabin

Test di primalità di Miller–Rabin

Sia $m \equiv 3 \bmod 4$

Test di primalità di Miller–Rabin

Sia $m \equiv 3 \bmod 4$

ALGORITMO DI MILLER RABIN CON k ITERAZIONI
$N = (m-1)/2$
`for` $j = 0$ `to` k `do` $a =$`Random`(m)
`if` $a^N \not\equiv \pm 1 \bmod m$ `then Output=(`m `composito): END`
`endfor Output=(`m `primo)`

Test di primalità di Miller–Rabin

Sia $m \equiv 3 \bmod 4$

ALGORITMO DI MILLER RABIN CON k ITERAZIONI
$N = (m-1)/2$
`for` $j = 0$ `to` k `do` $a =$`Random`(m)
`if` $a^N \not\equiv \pm 1 \bmod m$ `then Output=(`m `composito): END`
`endfor Output=(`m `primo)`

Test di primalitò di tipo Monte Carlo

Test di primalità di Miller–Rabin

Sia $m \equiv 3 \bmod 4$

ALGORITMO DI MILLER RABIN CON k ITERAZIONI
$N = (m-1)/2$
`for` $j = 0$ `to` k `do` $a =$`Random`(m)
`if` $a^N \not\equiv \pm 1 \bmod m$ `then Output=(`m `composito): END`
`endfor Output=(`m `primo)`

Test di primalitò di tipo Monte Carlo

$Prob$(Miller Rabin dichiari m `primo` quando m è composto) $\lessapprox \frac{1}{4^k}$

Test di primalità di Miller–Rabin

Sia $m \equiv 3 \bmod 4$

ALGORITMO DI MILLER RABIN CON k ITERAZIONI
$N = (m-1)/2$
`for` $j = 0$ `to` k `do` $a =$`Random`(m)
`if` $a^N \not\equiv \pm 1 \bmod m$ `then Output=(`m `composito): END`
`endfor Output=(`m `primo)`

Test di primalitò di tipo Monte Carlo

$Prob$(Miller Rabin dichiari m `primo` quando m è composto) $\lessapprox \frac{1}{4^k}$

Nel mondo reale il software usa Miller Rabin con $k = 10$

Test di primalità deterministici

Test di primalità deterministici

Teorema. (Miller, Bach) Se m è composto, allora

$$\textbf{GRH} \Longrightarrow \ \exists a \le 2\log^2 m \text{ tali che } \ a^{(m-1)/2} \not\equiv \pm 1 \pmod{m}.$$

(i.e. *m non è SPSP in base a.*)

Test di primalità deterministici

Teorema. (Miller, Bach) Se m è composto, allora

$$\textbf{GRH} \Rightarrow \ \exists a \leq 2\log^2 m \text{ tali che } \ a^{(m-1)/2} \not\equiv \pm 1 \pmod{m}.$$

(i.e. *m non è SPSP in base a.*)

Conseguenza: "Miller–Rabin si de–randomizza sotto GRH" ($m \equiv 3 \bmod 4$)

Test di primalità deterministici

Teorema. (Miller, Bach) Se m è composto, allora

$$\textbf{GRH} \Rightarrow \ \exists a \leq 2\log^2 m \text{ tali che } \ a^{(m-1)/2} \not\equiv \pm 1 \pmod{m}.$$

(i.e. *m non è SPSP in base a.*)

Conseguenza: "Miller–Rabin si de–randomizza sotto GRH" ($m \equiv 3 \bmod 4$)

`for`	$a = 2$ `to` $2\log^2 m$	`do`
	`if` $a^{(m-1)/2} \not\equiv \pm 1 \bmod m$	`then`
		`OUPUT=(`m `composto):   END`
`endfor`		`OUTPUT=(`m `primo)`

Test di primalità deterministici

Teorema. (Miller, Bach) Se m è composto, allora

$$\textbf{GRH} \Rightarrow \ \exists a \leq 2\log^2 m \text{ tali che } \ a^{(m-1)/2} \not\equiv \pm 1 \pmod{m}.$$

(i.e. *m non è SPSP in base a.*)

Conseguenza: "Miller–Rabin si de–randomizza sotto GRH" ($m \equiv 3 \bmod 4$)

`for`	$a = 2$ `to` $2\log^2 m$	`do`	
	`if` $a^{(m-1)/2} \not\equiv \pm 1 \bmod m$	`then`	
		`OUPUT=(`m `composto):`	`END`
`endfor`		`OUTPUT=(`m `primo)`	

Algoritmo deterministico polinomiale

Test di primalità deterministici

Teorema. (Miller, Bach) Se m è composto, allora

$$\mathbf{GRH} \Rightarrow \exists a \leq 2\log^2 m \text{ tali che } a^{(m-1)/2} \not\equiv \pm 1 \pmod{m}.$$

(i.e. *m non è SPSP in base a.*)

Conseguenza: "Miller–Rabin si de–randomizza sotto GRH" ($m \equiv 3 \bmod 4$)

`for`	$a = 2$ `to` $2\log^2 m$	`do`
	`if` $a^{(m-1)/2} \not\equiv \pm 1 \bmod m$	`then`
		`OUPUT=(`m `composto):   END`
`endfor`		`OUTPUT=(`m `primo)`

Algoritmo deterministico polinomiale

Per girare richiede $O(\log^5 m)$ operazioni in $\mathbb{Z}/m\mathbb{Z}$.

Record di primi certificati

Record di primi certificati

Record di primi certificati

- $2^{20996011}-1$, 6320430 cifre (scoperto nel 2003)
-
-
-
-
-
-
-

Record di primi certificati

- $2^{20996011} - 1$, 6320430 cifre (scoperto nel 2003)
- $2^{13466917} - 1$, 4053946 digits (discovered in 2001)

Record di primi certificati

- $2^{20996011} - 1$, 6320430 cifre (scoperto nel 2003)
- $2^{13466917} - 1$, 4053946 digits (discovered in 2001)
- $2^{6972593} - 1$, 2098960 cifre (scoperto nel 1999)

Record di primi certificati

- $2^{20996011} - 1$, 6320430 cifre (scoperto nel 2003)
- $2^{13466917} - 1$, 4053946 digits (discovered in 2001)
- $2^{6972593} - 1$, 2098960 cifre (scoperto nel 1999)
- $5359 \times 2^{5054502} + 1$, 1521561 cifre (scoperto nel 2003)

Record di primi certificati

- $2^{20996011}-1$, 6320430 cifre (scoperto nel 2003)
- $2^{13466917}-1$, 4053946 digits (discovered in 2001)
- $2^{6972593}-1$, 2098960 cifre (scoperto nel 1999)
- $5359 \times 2^{5054502}+1$, 1521561 cifre (scoperto nel 2003)
- $2^{3021377}-1$, 909526 cifre (scoperto nel 1998)

Record di primi certificati

- $2^{20996011} - 1$, 6320430 cifre (scoperto nel 2003)
- $2^{13466917} - 1$, 4053946 digits (discovered in 2001)
- $2^{6972593} - 1$, 2098960 cifre (scoperto nel 1999)
- $5359 \times 2^{5054502} + 1$, 1521561 cifre (scoperto nel 2003)
- $2^{3021377} - 1$, 909526 cifre (scoperto nel 1998)
- $2^{2976221} - 1$, 895932 cifre (scoperto nel 1997)

Record di primi certificati

- $2^{20996011} - 1$, 6320430 cifre (scoperto nel 2003)
- $2^{13466917} - 1$, 4053946 digits (discovered in 2001)
- $2^{6972593} - 1$, 2098960 cifre (scoperto nel 1999)
- $5359 \times 2^{5054502} + 1$, 1521561 cifre (scoperto nel 2003)
- $2^{3021377} - 1$, 909526 cifre (scoperto nel 1998)
- $2^{2976221} - 1$, 895932 cifre (scoperto nel 1997)
- $1372930^{131072} + 1$, 804474 cifre (scoperto nel 2003)
-

Record di primi certificati

- $2^{20996011} - 1$, 6320430 cifre (scoperto nel 2003)
- $2^{13466917} - 1$, 4053946 digits (discovered in 2001)
- $2^{6972593} - 1$, 2098960 cifre (scoperto nel 1999)
- $5359 \times 2^{5054502} + 1$, 1521561 cifre (scoperto nel 2003)
- $2^{3021377} - 1$, 909526 cifre (scoperto nel 1998)
- $2^{2976221} - 1$, 895932 cifre (scoperto nel 1997)
- $1372930^{131072} + 1$, 804474 cifre (scoperto nel 2003)
- $1176694^{131072} + 1$, 795695 cifre (scoperto nel 2003)

Il test di primalità deterministico AKS

Il test di primalità deterministico AKS

Department di Computer Science & Engineering,
I.I.T. Kanpur, Agost 8, 2002.

Il test di primalità deterministico AKS

Department di Computer Science & Engineering,
I.I.T. Kanpur, Agost 8, 2002.

Nitin Saxena, Neeraj Kayal e Manindra Agarwal

Il test di primalità deterministico AKS

Department di Computer Science & Engineering,
I.I.T. Kanpur, Agost 8, 2002.

Nitin Saxena, Neeraj Kayal e Manindra Agarwal

Nuovo test di primalità, deterministico e polinomiale

Il test di primalità deterministico AKS

Department di Computer Science & Engineering,
I.I.T. Kanpur, Agost 8, 2002.

Nitin Saxena, Neeraj Kayal e Manindra Agarwal

Nuovo test di primalità, deterministico e polinomiale

Risolve il problema #1 in Teoria computazionale dei numeri.

Il test di primalità deterministico AKS

Department di Computer Science & Engineering,
I.I.T. Kanpur, Agost 8, 2002.

Nitin Saxena, Neeraj Kayal e Manindra Agarwal

Nuovo test di primalità, deterministico e polinomiale

Risolve il problema #1 in Teoria computazionale dei numeri.

http://www.cse.iitk.ac.in/news/primality.html

Come funziona AKS?

Come funziona AKS?

Teorema. (AKS) Sia $n \in \mathbb{N}$. Siano q, r primi, $S \subseteq \mathbb{N}$ finito:

- $q|r-1$;
- $n^{(r-1)/q} \bmod r \notin \{0,1\}$;
- $\gcd(n, b-b') = 1, \quad \forall b, b' \in S$ (distinti);
- $\binom{q+\#S-1}{\#S} \geq n^{2\lfloor\sqrt{r}\rfloor}$;
- $(x+b)^n = x^n + b$ in $\mathbb{Z}/n\mathbb{Z}[x]/(x^r-1), \quad \forall b \in S$;

Allora n è una potenza di un primo

formulazione di Bernstein

Come funziona AKS?

Teorema. (AKS) Sia $n \in \mathbb{N}$. Siano q, r primi, $S \subseteq \mathbb{N}$ finito:

- $q | r - 1$;
- $n^{(r-1)/q} \bmod r \notin \{0, 1\}$;
- $\gcd(n, b - b') = 1, \quad \forall b, b' \in S$ (distinti);
- $\binom{q+\#S-1}{\#S} \geq n^{2\lfloor\sqrt{r}\rfloor}$;
- $(x + b)^n = x^n + b$ in $\mathbb{Z}/n\mathbb{Z}[x]/(x^r - 1), \quad \forall b \in S$;

Allora n è una potenza di un primo

formulazione di Bernstein

Teorema di Fouvry (1985) $\Longrightarrow \quad \exists r \approx \log^6 n, s \approx \log^4 n$

Come funziona AKS?

Teorema. (AKS) Sia $n \in \mathbb{N}$. Siano q, r primi, $S \subseteq \mathbb{N}$ finito:

- $q | r - 1$;
- $n^{(r-1)/q} \bmod r \notin \{0, 1\}$;
- $\gcd(n, b - b') = 1, \quad \forall b, b' \in S$ (distinti);
- $\binom{q+\#S-1}{\#S} \geq n^{2\lfloor\sqrt{r}\rfloor}$;
- $(x + b)^n = x^n + b$ in $\mathbb{Z}/n\mathbb{Z}[x]/(x^r - 1), \quad \forall b \in S$;

Allora n è una potenza di un primo

formulazione di Bernstein

Teorema di Fouvry (1985) $\Longrightarrow \quad \exists r \approx \log^6 n, s \approx \log^4 n$

$\Longrightarrow$ AKS richiede $O(\log^{17} n)$ operazioni in $\mathbb{Z}/n\mathbb{Z}$.

Come funziona AKS?

Teorema. (AKS) Sia $n \in \mathbb{N}$. Siano q, r primi, $S \subseteq \mathbb{N}$ finito:

- $q | r - 1$;
- $n^{(r-1)/q} \bmod r \notin \{0, 1\}$;
- $\gcd(n, b - b') = 1, \quad \forall b, b' \in S$ (distinti);
- $\binom{q+\#S-1}{\#S} \geq n^{2\lfloor\sqrt{r}\rfloor}$;
- $(x + b)^n = x^n + b$ in $\mathbb{Z}/n\mathbb{Z}[x]/(x^r - 1), \quad \forall b \in S$;

Allora n è una potenza di un primo   formulazione di Bernstein

Teorema di Fouvry (1985) $\Longrightarrow \quad \exists r \approx \log^6 n, s \approx \log^4 n$

$\Longrightarrow$ AKS richiede $O(\log^{17} n)$ operazioni in $\mathbb{Z}/n\mathbb{Z}$.

Svariate semplificazioni e miglioramenti: Bernstein, Lenstra, Pomerance.....

Perchè RSA è sicuro?

Perchè RSA è sicuro?

☞

☞

☞

Perchè RSA è sicuro?

- È chiaro che se Carlo riesce a fattorizzare M,
-
-

Perchè RSA è sicuro?

- È chiaro che se Carlo riesce a fattorizzare M, allora può anche calcolare $\varphi(M)$, d e quindi decifrare i messaggi
-
-

Perchè RSA è sicuro?

- È chiaro che se Carlo riesce a fattorizzare M, allora può anche calcolare $\varphi(M)$, d e quindi decifrare i messaggi
- Calcolare $\varphi(M)$ è equivalente a fattorizzare M. In fatti
-

Perchè RSA è sicuro?

- È chiaro che se Carlo riesce a fattorizzare M,
  allora può anche calcolare $\varphi(M)$, d e quindi decifrare i messaggi
- Calcolare $\varphi(M)$ è equivalente a fattorizzare M. In fatti

$$p, q = \frac{M - \varphi(M) + 1 \pm \sqrt{(M - \varphi(M) + 1)^2 - 4M}}{2}$$

-

Perchè RSA è sicuro?

- È chiaro che se Carlo riesce a fattorizzare M,
  allora può anche calcolare $\varphi(M)$, d e quindi decifrare i messaggi
- Calcolare $\varphi(M)$ è equivalente a fattorizzare M. In fatti

$$p, q = \frac{M - \varphi(M) + 1 \pm \sqrt{(M - \varphi(M) + 1)^2 - 4M}}{2}$$

- **Ipotesi RSA.** L'unico modo per calcolare efficientemente

Perchè RSA è sicuro?

- È chiaro che se Carlo riesce a fattorizzare M,
  allora può anche calcolare $\varphi(M)$, d e quindi decifrare i messaggi
- Calcolare $\varphi(M)$ è equivalente a fattorizzare M. In fatti
  $$p, q = \frac{M - \varphi(M) + 1 \pm \sqrt{(M - \varphi(M) + 1)^2 - 4M}}{2}$$
- **Ipotesi RSA.** L'unico modo per calcolare efficientemente
  $$x^{1/e} \bmod M, \quad \forall x \in \mathbb{Z}/M\mathbb{Z}$$

Perchè RSA è sicuro?

☞ È chiaro che se Carlo riesce a fattorizzare M,
allora può anche calcolare $\varphi(M)$, d e quindi decifrare i messaggi

☞ Calcolare $\varphi(M)$ è equivalente a fattorizzare M. In fatti

$$p, q = \frac{M - \varphi(M) + 1 \pm \sqrt{(M - \varphi(M) + 1)^2 - 4M}}{2}$$

☞ **Ipotesi RSA.** L'unico modo per calcolare efficientemente

$$x^{1/e} \bmod M, \quad \forall x \in \mathbb{Z}/M\mathbb{Z}$$

(i.e. decifrare messaggi) è fattrizzare M

Perchè RSA è sicuro?

- ☞ È chiaro che se Carlo riesce a fattorizzare M, allora può anche calcolare $\varphi(M)$, d e quindi decifrare i messaggi
- ☞ Calcolare $\varphi(M)$ è equivalente a fattorizzare M. In fatti

$$p, q = \frac{M - \varphi(M) + 1 \pm \sqrt{(M - \varphi(M) + 1)^2 - 4M}}{2}$$

- ☞ **Ipotesi RSA.** L'unico modo per calcolare efficientemente

$$x^{1/e} \bmod M, \quad \forall x \in \mathbb{Z}/M\mathbb{Z}$$

(i.e. decifrare messaggi) è fattrizzare M

In altre parole

Perchè RSA è sicuro?

☞ È chiaro che se Carlo riesce a fattorizzare M,
allora può anche calcolare $\varphi(M)$, d e quindi decifrare i messaggi

☞ Calcolare $\varphi(M)$ è equivalente a fattorizzare M. In fatti

$$p, q = \frac{M - \varphi(M) + 1 \pm \sqrt{(M - \varphi(M) + 1)^2 - 4M}}{2}$$

☞ **Ipotesi RSA.** L'unico modo per calcolare efficientemente

$$x^{1/e} \bmod M, \quad \forall x \in \mathbb{Z}/M\mathbb{Z}$$

(i.e. decifrare messaggi) è fattrizzare M

In altre parole

I due problemi sono polinomialmente equivalenti