Firma digitale e PEC: aspetti crittografici e sicurezza

Prof. Massimiliano Sala

Università degli Studi di Trento, Lab di Matematica Industriale e Crittografia

Trento, 17 Febbraio 2012

Introduzione agli aspetti crittografici

La sicurezza matematica: firma digitale e PEC

Essenzialmente la sicurezza della PEC risiede nella **robustezza** della **firma digitale**

Essendo spesso coinvolte transizioni on-line, bisogna considerare anche la network security $\longrightarrow$ sicurezza della comunicazione browser-server.

Le parti in gioco

Gli aspetti crittografici coinvolti nel nostro caso sono:

- Funzioni Hash
- Crittografia a chiave pubblica
- Crittografia a chiave privata

Le parti in gioco

Gli aspetti crittografici coinvolti nel nostro caso sono:

- Funzioni Hash
- Crittografia a chiave pubblica

} Firma digitale

- Crittografia a chiave privata

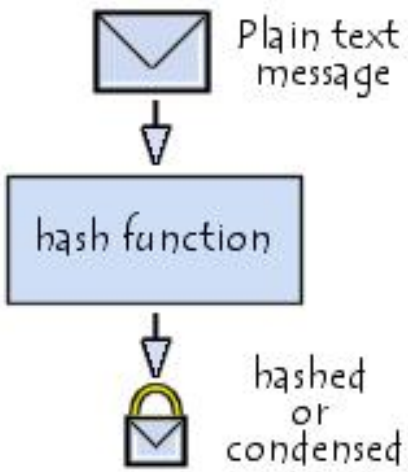
Le parti in gioco

Gli aspetti crittografici coinvolti nel nostro caso sono:

- Funzioni Hash
- Crittografia a chiave pubblica
- Crittografia a chiave privata

} Sicurezza browser-server

Funzioni HASH



Una **funzione hash** è una funzione che converte un vettore arbitrariamente lungo in un vettore di lunghezza fissata:

$$H : \mathbb{F}^{\infty} \rightarrow \mathbb{F}^{h}$$

dove $\mathbb{F} = \{ 0, 1 \}$.

Una collisione per H sono due file con lo stesso HASH.

Una buona funzione di HASH non dovrebbe permettere di trovare collisioni facilmente.

Crittografia a chiave privata

Esiste un'unica chiave privata K, sia per l'operazione di cifratura che per l'operazione di decifratura.

Possiamo pensare l'insieme dei messaggi da trasmettere come l'insieme delle stringhe binarie lunghe n.
Formalmente $\mathcal{M} = \mathbb{F}^n$, dove $\mathbb{F} = \{0, 1\}$.

Scelta una chiave K una funzione di crittazione è una mappa $f_k : \mathcal{M} \to \mathcal{M}$ (bigettiva) che trasforma plaintext in ciphertext, in modo che a ciascun plaintext corrisponda un solo ciphertext e viceversa.

Crittografia a chiave pubblica

A differenza del caso precedente non esiste solo una chiave privata, perchè siamo costretti ad usare un canale pubblico.

Diffie-Hellman

che consente a due entità di stabilire una chiave condivisa e segreta utilizzando un canale di comunicazione pubblico.

Cifratura asimmetrica

grazie alla quale si cifra e decifra usando una combinazione di chiave pubblica e privata (non condivisa).

Introduzione alla sicurezza crittografica

Come stimare robustezza sistema crittografico?

In ambito non militare la robustezza del sistema si basa sul miglior attacco che rompe il sistema.

Nel caso della chiave privata (cioè una sola chiave condivisa K), l'unico modo per rompere un cifrario ideale è provare tutte le chiavi (brute force), ovvero:

$$2^k,$$

nel caso in cui K ha k bit, ovvero $K \in \mathbb{F}^k$.

Come stimare robustezza sistema crittografico?

La sicurezza di un cifrario qualunque si rapporta a quella di un cifrario ideale.
Useremo k per confronto, ovvero nel il caso ideale k=k, nel caso generale $k \geq k$.

Ad esempio se un sistema ha un insieme di $2^{10} = 1024$ chiavi, ma si rompe usando solo 2^6 cifrature, allora si dice che:
$k = 10$
$k = 6$.

Chiavi deboli?

Per evitare formalismi diremo che:
una **chiave debole** K, in un sistema crittografico (sia pubblico che privato) è una chiave che "permette la rottura del sistema".

Riteniamo quindi fondamentale per la sicurezza valutare attentamente la robustezza delle chiavi usate (e quindi la loro generazione), per evitare l'uso involontario di chiavi deboli.

Chiavi random?

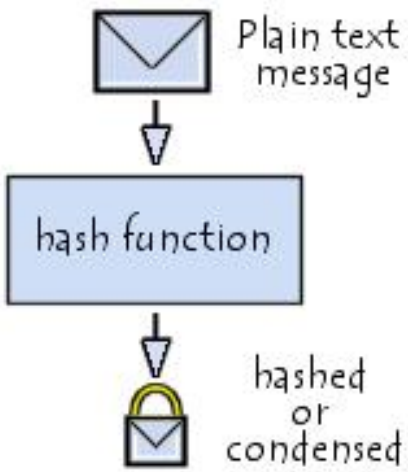
Sia nella crittografia pubblica, che in quella privata interviene la generazione random delle chiavi:

- PRIVATA: per generare la chiave vera e propria;
- PUBBLICA: per generare la parte privata della chiave o il segreto condiviso.

Una pessima qualità del random vanifica tutto lo sforzo per proteggere il sistema.

Funzioni Hash

SHA-1



- SHA-1: Security Hash Algorithm [FIPS 180-1]
- dato un messaggio in input di lunghezza massima 2^{64} bits, SHA-1 restituisce come output una stringa di 160 bit

input: ''m'illumino di immenso''

output:

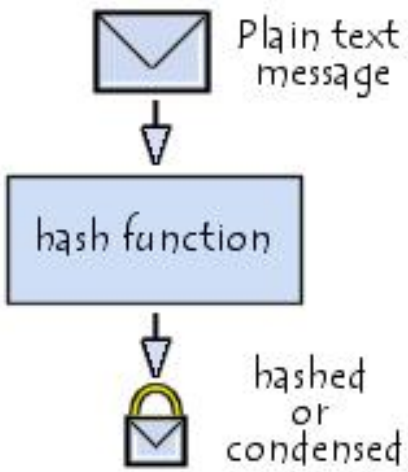
04DEC8C39C14B4E5AB284EE204C81D58F1A59936

input: ''Roma''

output:

DE5429D6F4FA2C86427A50757791DE88A0B75C85

SHA-1



- SHA-1: Security Hash Algorithm [FIPS 180-1]
- dato un messaggio in input di lunghezza massima 2^{64} bits, SHA-1 restituisce come output una stringa di 160 bit

input: ''mi illumino di immenso''

output:

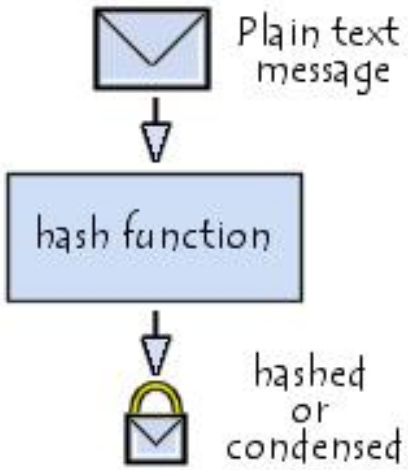
666BCFA1CC6D6580F316AF077B85B9DE34055A57

input: ''roma''

output:

A6B6EA31C49A8E944EFE9ECBC072A26903A1461A

SHA-2



- SHA-256: Security Hash Algorithm [FIPS 180-2]
- dato un messaggio in input di lunghezza massima 2^{64} bits, SHA-256 restituisce come output una stringa di 256 bit

SHA-256 è ritenuto nettamente più sicuro di SHA-1

Sicurezza della SHA

Complessità di trovare una collisione, ovvero due file con lo stesso HASH.

SHA1	$k = 80$	$k = 58$
SHA256	$k = 128$	$k = 128$
SHA3	$k =?$	$k =?$

Crittografia a chiave privata

Principali cifrari

Su internet, i principali cifrari usati sono:

DES	$k = 56$	$k = 43$
3DES	$k = 168$	$k = 113$
AES256	$k = 256$	$k = 254$
RC4	$k = 256$	$k = 256$ (?)

AES256

Il miglior attacco (aiuto?) noto a AES256 (che recupera la chiave) ha un costo di 2^{254} crittazioni.
Supponendo che

- un core medio riesca a eseguire 2^{50} crittazioni al secondo,
- esistano 1000 core ogni abitante del pianeta, compresi neonati e le persone che vivono nella giungla amazzonica,
- al mondo ci siano 20 miliardi di persone,

AES256

Il miglior attacco (aiuto?) noto a AES256 (che recupera la chiave) ha un costo di 2^{254} crittazioni.
Supponendo che

- un core medio riesca a eseguire 2^{50} crittazioni al secondo,
- esistano 1000 core ogni abitante del pianeta, compresi neonati e le persone che vivono nella giungla amazzonica,
- al mondo ci siano 20 miliardi di persone,

Allora TUTTA la potenza del mondo unita dovrebbe lavorare senza sosta per 10^{31} MILIARDI di anni.

RC4

RC4 è un cifrario a flusso, quindi genera un flusso di byte.

- conviene usare 256 bit per la chiave;
- bisogna usarlo **con attenzione**, in particolare bisogna scartare i primi byte che genera;
- nota tecnica: esiste un problema con il re-keying process.

Crittografia a chiave pubblica

Algoritmi di crittografia asimmetrica

La sicurezza dell'algoritmo di cifratura asimmetrica utilizzato si basa sulla complessita' del problema matematico che sta alla base.

I principali algoritmi sono:

- DSA (Digital Signature Algorithm), basato sul **Discrete Logarithm Problem** (DLP)
- RSA, basato sull' **Integer Factorization Problem** (IFP)
- ECC
- ECDLP
- HFE

Descrizione di DSA

Descriviamo la struttura dell'algoritmo. Alice  vuole trasmettere il messaggio m a Bob  e insieme ad esso la sua firma digitale.

Occorre una coppia di chiavi asimmetriche.

- Alice deve creare una chiave pubblica $k_{p_a} = y$ ed una privata $k_{s_a} = x$ stando attenta a non divulgare x
- Alice trasmette y a Bob

Descrizione di DSA: generazione delle chiavi

Come crea Alice la chiave pubblica e privata?

- Alice sceglie due numeri primi p e q, tali che $p = qz + 1$ per un qualche numero intero z,
- Alice sceglie un numero h tale che $1 < h < p - 1$ e $g = h^z \mod p > 1$
- Alice genera un numero casuale x tale che $0 < x < q$.
- Alice calcola $y = g^x \mod p$
- solo Alice  conosce la chiave segreta x mentre y è pubblica
- I parametri (p, q, g) sono pubblici e possono essere condivisi da diversi utenti.

Descrizione di DSA: generazione della firma

Come avviene la firma del messaggio?

- Alice genera un numero casuale k tale che $0 < k < q$ e calcola $r = (g^k \mod p) \mod q$
- Alice calcola $s = (k^{-1}(H(m) + x * r)) \mod q$ dove $H(m)$ e' una funzione di hash SHA-d applicata al messaggio m
- Nel caso in cui $r = 0$ o $s = 0$ bisogna ricalcolare la firma
- Altrimenti Alice trasmette il messaggio m e la firma (r, s) a Bob

Descrizione di DSA: verifica della firma

Come avviene la verifica della firma?

- Bob rifiuta la firma se non sono soddisfate le condizioni $0 < r < q$ e $0 < s < q$
- Bob calcola $w = s^{-1} \mod q$
- Bob calcola $u_1 = (H(m) * w) \mod q$
- Bob calcola $u_2 = (r * w) \mod q$
- Bob calcola $((g^{u_1} * y^{u_2}) \mod p) \mod q$
- La firma e' verificata se $v = r$




Descrizione di RSA

Possiamo così passare a descrivere il cifrario. Alice  vuole trasmettere il messaggio m a Bob  .

- Bob deve creare una chiave pubblica $k_{p_b} = (N, e)$ ed una privata $k_{s_b} = (N, d)$ stando attento a non divulgare d
- Bob trasmette (N, e) ad Alice

Descrizione di RSA *

Come crea Bob la chiave pubblica e privata?

- Bob sceglie due numeri primi p e q, molto grandi e li moltiplica $N = p \cdot q$
- Bob sceglie un numero e, coprimo con $\varphi(N)$ e più piccolo di $(p-1)\cdot(q-1)$
- Bob calcola d tale che sia l'inverso moltiplicativo di e mod $\varphi(N)$, cioé $(x^d)^e \equiv x \bmod N$, qualunque x.
- solo Bob  conosce la chiave segreta (N, d) mentre (N, e) è pubblica

Descrizione di RSA *

Come avviene la cifratura e decifratura?

- Alice calcola $c = m^e \mod N$
- Alice trasmette c a Bob
- Bob riceve c e lo decripta calcolando:

$$c^d \equiv m \mod N$$

Chiavi forti di RSA

Come si può ricavare la chiave privata da quella pubblica?

Apparentemente ci sono 3 alternative:

1. fattorizzare N
2. Calcolare $\varphi(N)$
3. ricavare il valore d direttamente dalla chiave pubblica (N, e)

Si può facilmente dimostrare che queste tre metodi sono equivalenti:

$$(1) \Leftrightarrow (2) \Leftrightarrow (3)$$

La resistenza di una chiave forte è $n \sim k^{1.7}$.

Cosa vuol dire in pratica?

Chiavi forti di RSA: un caso pratico

Usando l'algoritmo GNFS si riescono a fattorizzare numeri N dell'ordine di $2^{768} \sim 10^{231}$.

Spendendo un milione di euro per dotarsi di un cluster potente, si potrebbe rompere una chiave RSA-768 (cioè N è dell'ordine di 2^{768}) ogni 6 mesi (in media).

Lo sforzo per rompere una chiave RSA-1024 è circa mille forte quello per rompere una chiave di RSA-768, quindi chiavi RSA-1024 o superiori (purchè non siano chiavi deboli) sono al momento fuori dalla portata di un attacco pratico.

Considerazioni

RSA 1024 bit è più debole di AES 256 bit.

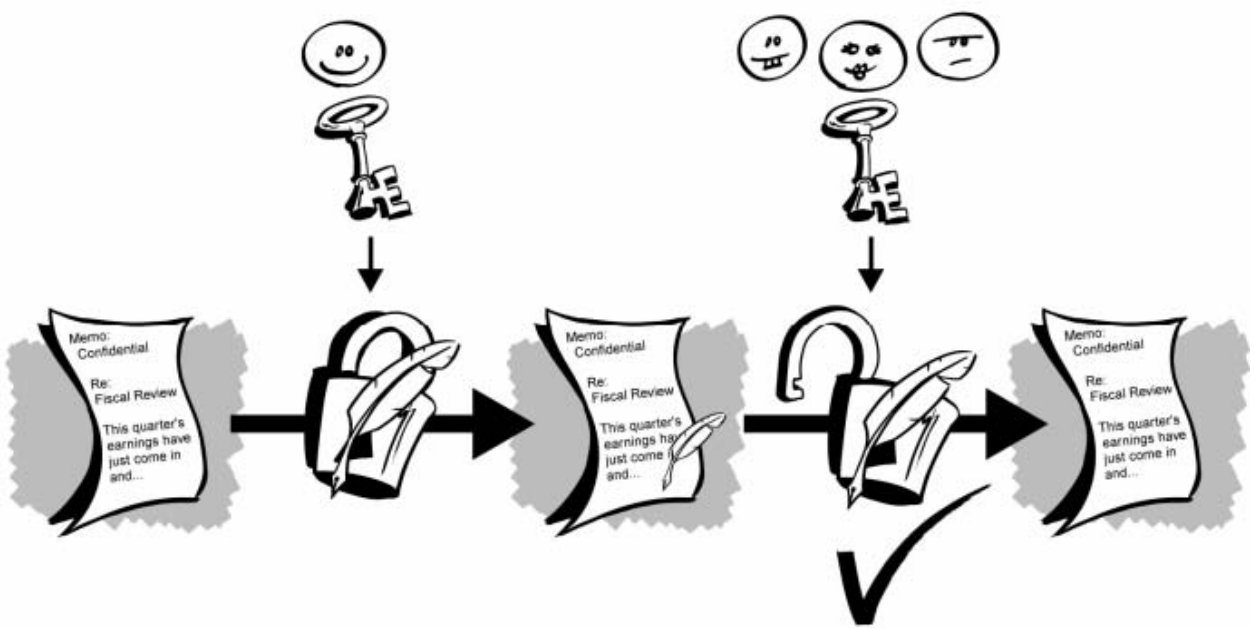
Attenzione: RSA lavora su un canale pubblico e quindi per forza deve compiere uno sforzo maggiore per ottenere lo stesso risultato.

Firma Digitale

Cos'è: un espediente per autenticare una qualunque sequenza di cifre binarie, indipendentemente dal significato.

È ottenuta tramite l'utilizzo di

Funzione Hash + Algoritmo di crittografia asimmetrica



Forgery

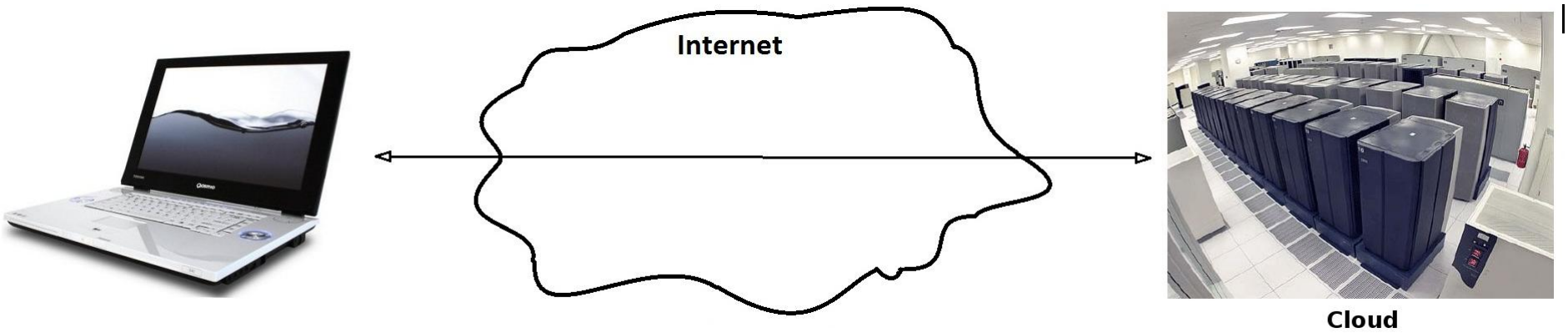
Due possibili strade percorribili:

- rompere la chiave privata
- ottenere un hash finto, ossia forzare una collisione.

Network security

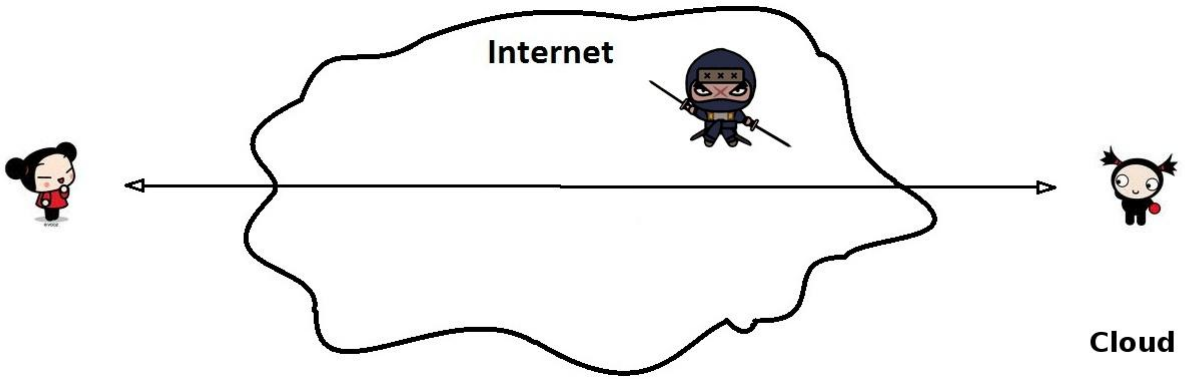
SSL/TLS

É il protocollo usato a livello di comunicazione tra browser e server per la trasmissione sicura dei dati.



SSL/TLS

É il protocollo usato a livello di comunicazione tra browser e server per la trasmissione sicura dei dati.



Fasi SSL/TLS

1. A e B si mettono d'accordo sulla crittografia da usare (Cipher Suite)
2. A si crea per conto suo dei bit segreti random
3. B fa lo stesso
4. A e B scambiano una chiave segreta K (segreto condiviso) tramite RSA o DH
5. A e B usano K con AES, 3DES o DES per crittare la conversazione

Pericoli nascosti

Randomicità

E' essenziale che le chiavi siano generate in modo che all'attaccante sembrino random.

Questo perchè?

Il rischio più grande è se l'attaccante riesce a capire come rigenerarle e quindi ricrearle.

Creazione randomicità

Esistono principalmente tre modi per ottenere una chiave random:

- utilizzare sorgenti random naturali (es. campionamento del campo elettrico e della voce, rilevamento fenomeni quantistici)
- utilizzare algoritmi matematici pseudo-random (PRF)
- utilizzare una commistione di sorgenti random e pseudo-random

Quanto costa il random?

È importante riflettere su due aspetti:

- quanto costa creare random
- quanto costa condividerlo

Quanti random?

	CONDIVISI	NON CONDIVISI
TANTI BIT RANDOM	impossibile (o costosissimo)	• segreti in DH • segreti in RSA • segreti collegati al piccolo segreto condiviso
POCHI BIT RANDOM	• K chiave simmetrica (AES, DES, . . .) • seed di PRF	

PRF: seed$\rightarrow$ 101000111 . . . . . .

Chiavi Deboli

Spesso si pensa che più una chiave è lunga, più questa è sicura.
In realtà non è vero.
Le chiavi deboli sono molto più pericolose, perchè la loro debolezza soggiace nella struttura algebrica della chiave stessa.

A titolo esemplicativo, ci concentriamo su RSA.

Chiavi Deboli di RSA

É facile rompere il sistema se ad esempio:

- se m ed e sono così piccoli che $m^e < N$; allora risulta facile trovare la radice e-esima di c, poichè $c = m^e$
- se $p - q < N^{\frac{1}{4}}$, ci sono algoritmi veloci per trovare p e q
- se $p - 1$ o $q - 1$ hanno solo fattori piccoli, N si fattorizza velocemente con l'algoritmo di Pollard $p - 1$
- se $d < \frac{1}{3} N^{\frac{1}{4}}$ (attacco di Wiener)

Purtroppo molte implementazioni dell'algoritmo RSA non controllano l'eventualità di avere una chiave debole.

Possiamo fidarci delle chiavi generate?

Chi genera le chiavi?

Dispositivi chiamati HSM (Hardware Security Module) sono solitamente chiamati a generare le chiavi.

Chi controlla chi genera le chiavi?

Per gli HSM esistono molti certificati (= troppi?):

- FIPS 140-2 level 2, 3
- Common Criteria EAL4+
- procedura accreditamento OCSI
- MEPS
- IdenTrust
- e-passport BAC and EAC
- Entrust
- APCA
- PCI HSM

Quanti deboli le deboli?

Le seguenti chiavi RSA 1024:

- $N = 313256375876377212440015318323108678520006173008034636063302469362957128066154999976915271523113164049627718408844530703484989822647023510143641161477819303748834981751956165887681798612125473753385171353058142794346935884074455798705326089160080254660362883833231925389841878822795695169771993196329575554015$
  $e = 29$
- $N = 703606728829425222386754525735867532353561794249517072279524442134219740346812313911122376577798291308343173684600484528948470625632095674346511670057314481944944614776388555094301742500113459518088266875640891456768243692575911682249129488446432250644872863589855553619827615234588027031314797877915$
  $e = 37$

Le ho date come sfida ai team di studenti impegnati nelle CryptoWars e le hanno rotte in

Quanti deboli le deboli?

Le seguenti chiavi RSA 1024:

- $N = 313256375876377212440015318323108678520006173008034636063302469362957128066154999976915271523113164049627718408844530703484989822647023510143641161477819303748834981751956165887681798612125473753385171353058142794346935884074455798705326089160080254660362883833231925389841878822795695169771993196329575554015$
  $e = 29$
- $N = 703606728829425222386754525735867532353561794249517072279524442134219740346812313911122376577798291308343173684600484528948470625632095674346511670057314481944944614776388555094301742500113459518088266875640891456768243692575911682249129488446432250644872863589855553619827615234588027031314797877915$
  $e = 37$

Le ho date come sfida ai team di studenti impegnati nelle CryptoWars e le hanno rotte in meno di un secondo.

Hanno rotto anche una chiave di curve ellittiche da 300 bit in meno di un minuto!

Ma tutto questo è solo un gioco?

Ma tutto questo è solo un gioco?

Presentiamo i risultati di un recentissimo lavoro scientifico appena uscito (14/02/2012), che porta la firma di Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter.

Ma tutto questo è solo un gioco?

Presentiamo i risultati di un recentissimo lavoro scientifico appena uscito (14/02/2012), che porta la firma di Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter.

- 270.000 siti che condividono la chiave pubblica con almeno un altro sito;

Ma tutto questo è solo un gioco?

Presentiamo i risultati di un recentissimo lavoro scientifico appena uscito (14/02/2012), che porta la firma di Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter.

- 270.000 siti che condividono la chiave pubblica con almeno un altro sito;
- 71.052 chiavi sono usate da tanti siti diversi (stessa chiave anche per migliaia di siti);

Ma tutto questo è solo un gioco?

Presentiamo i risultati di un recentissimo lavoro scientifico appena uscito (14/02/2012), che porta la firma di Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter.

- 270.000 siti che condividono la chiave pubblica con almeno un altro sito;
- 71.052 chiavi sono usate da tanti siti diversi (stessa chiave anche per migliaia di siti);
- 12.720 chiavi RSA1024 sono vulnerabili e si rompono in meno di un millisecondo.

Grazie dell'attenzione.