Altre alternative a RSA interessanti e praticabili

Prof. Massimiliano Sala

MINICORSI 2011. Crittografia a chiave pubblica: oltre RSA

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

24 marzo 2011

1. Crittografia con il LOGARITMO DISCRETO

Che cosa è il Logaritmo Discreto

Il logaritmo discreto è in algebra astratta il corrispettivo del logaritmo $\log x$ ($e^{\log x} = x$).

In maniera più rigorosa:

Definizione

Sia G un gruppo ciclico con n elementi. Sia $g \in G$ tale che ogni elemento $b \in G$ può essere scritto nella forma $b = g^k$ per un certo intero k.

Chiamiamo k il logaritmo discreto di b, ovvero $\log_g b$.

Che cosa è il Logaritmo Discreto

Dato un gruppo G, calcolare il logaritmo discreto di un suo elemento rispetto ad un generatore b non è una cosa semplice.

Questo è il motivo per cui è usato in Crittografia.

Un metodo per calcolarlo, sarebbe la ricerca esaustiva, che però richiede un tempo di calcolo lineare rispetto a $|G|$ e quindi esponenziale rispetto a $\log_2 |G|$ (il numero di cifre di $|G|$).

Calcolare il Logaritmo Discreto

Alcuni metodi per il calcolo del logaritmo discreto su un gruppo qualunque:

- Baby-step Giant-step
- algoritmo ρ di Pollard
- algoritmo di Pohlig-Hellman
- ...

Tutti questi metodi hanno complessità esponenziale.

L' Algoritmo Diffie - Hellman (DH)

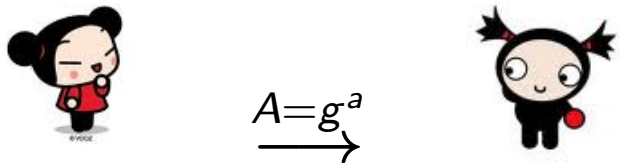
L'algoritmo è particolarmente adatto per scambiarsi una chiave segreta attraverso un canale non sicuro.

Alice  e Bob  vogliono negoziare una chiave segreta.

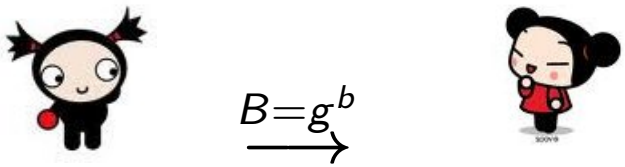
Innanzitutto Alice sceglie un primo N e un intero g, con $(N, g) = 1$.

(N, g) è la chiave pubblica k_{p_a}.

Alice sceglie un numero segreto a e trasmette g^a a Bob:



Bob sceglie un numero segreto b e trasmette g^b ad Alice:



A questo punto Alice calcola $B^a = g^{a \cdot b}$;

Bob calcola $A^b = g^{a \cdot b}$.

Quindi hanno ottenuto la stessa chiave $g^{a \cdot b}$.

L' Algoritmo Diffie - Hellman (DH)

Se si riesce a risolvere il problema del logaritmo discreto (DLP), è anche possibile attaccare questo crittosistema.
Infatti ad un attaccante basta calcolare il logaritmo di A o B per poi ricostruire la chiave segreta $g^{a \cdot b}$.

$$\text{Risolvere DLP} \longrightarrow \text{Attaccare DH}$$

Il viceversa non è noto:

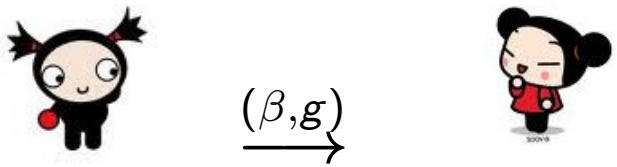
$$\text{Attaccare DH} \not\longrightarrow \text{Risolvere DLP ?}$$

non sono noti al giorno d'oggi algoritmi che risolvono DH senza calcolare il logaritmo discreto.

L' Algoritmo El Gamal

Questo è un altro crittosistema sempre basato sul logaritmo discreto.
Il messaggio m che Alice vuole trasmettere a Bob è un elemento di di G.

Bob genera la chiave pubblica, prendendo g e un numero intero a segreto, $0 < a < n$. Calcola $\beta = g^a$ e trasmette la chiave pubblica (β, g) ad Alice (G è pubblico).



L' Algoritmo El Gamal

Alice sceglie un intero segreto k, con $0 < k < n$ e critta m nel seguente modo:

$$(g^k, m\beta^k) = (\gamma, \delta)$$

$\xrightarrow{(\gamma,\delta)}$

Per decifrare il messaggio, ora Bob calcola:

$$\gamma^{-a} \cdot \delta = g^{-ak} \cdot m\beta^k = g^{-ak} \cdot mg^{ak} = m$$

Crittosistemi basati sul Logaritmo Discreto

1. Crittografia con Curve Ellittiche (ECC)
2. Crittografia su Campi Finiti $\mathbb{F}_q$
3. Crittografia su Curve Iperellittiche

Crittografia con Curve Ellittiche (ECC)

Come abbiamo mostrato nella lezione precedente, è possibile usare i punti di una curva ellittica **E** su un campo finito $\mathbb{F}_q$.

Se $f(x,y) = y^2 + a_1xy + a_3y - x^3 - a_2x^2 - a_4x - a_6 = 0$, allora:

$$\mathbf{E} = \{(x,y) \in \mathbb{F}_q^2 |\ f(x,y) = 0\} \cup \mathcal{O}$$

$(\mathbf{E}, +)$ è un gruppo additivo, con un numero finito di elementi (il punto all'infinito $\mathcal{O}$ è lo *zero*).

Crittografia su Campi Finiti $\mathbb{F}_q$

Alternativamente possiamo utilizzare un qualsiasi campo finito $\mathbb{F}_q$, dove $q = p^m$, p primo e m intero positivo.

Se gli togliamo lo 0 otteniamo in gruppo finito $\mathbb{F}_q^*$, che è ciclico grazie al *Teorema dell'elemento primitivo*.

Crittografia su Campi Finiti $\mathbb{F}_q$

Per costruire un campo finito con p^m elementi, possiamo considerare l'insieme dei polinomi a coefficienti in $\mathbb{Z}_p = \{1,\ 2,\ 3,\ \ldots,\ p-1\}$ e prendere i resti della divisione per $f(x)$, polinomio irriducibile di grado m.

$$\mathbb{F}_q = \mathbb{Z}_p[x]/f(x) = \{h(x) \in \mathbb{Z}_p[x] \ \mid \ \deg h(x) < m\}$$

$(\mathbb{F}_q \setminus \{0\}, \cdot)$ è un gruppo moltiplicativo abeliano, finito e ciclico su cui possiamo sfruttare le difficoltà di calcolo del logaritmo discreto.

Crittografia su Campi Finiti $\mathbb{F}_q$

I più veloci algoritmi per calcolare il logaritmo discreto su un campo finito sono quelli di tipo *index calculus*, che sono subesponenziali probabilisticamente.

In particolare:

1. algoritmo di Coppersmith, se siamo su $\mathbb{F}_{2^m}$ con 2^m fino a 2^{613}
2. NFS (number fields sieve), se siamo su $\mathbb{F}_p$, con p primo minore di 2^{530}.
3. FFS (function fields sieve), se siamo su un generico campo finito con p^m elementi, con p piccolo e p^m dell'ordine minore di 2^{673}.

Curve Iperellittiche

Definizione (Curva Iperellittica)

Una Curva Iperellittica **I** *su un campo* $\mathbb{F}_q$ *di genere* g *è l'insieme dei punti su* $(\mathbb{F}_q)^2$ *che soddisfano:*

$$y^2 + h(x)y = f(x)$$

dove $h(x) \in \mathbb{F}_q[x]$ *è un polinomio di grado minore o uguale a* g, $f(x) \in \mathbb{F}_q[x]$ *monico e di grado* $2g + 1$.

Nota

Le Curve Ellittiche sono Curve Iperellittiche di genere $g = 1$

Curve Iperellittiche

Data $\mathbf{I}$, possiamo definire delle combinazioni finite di punti della curva, che chiameremo *divisori* D come:

$$D = \sum a_i P_i \text{ con } a_i \in \mathbb{Z},\ P_i \in \mathbf{I}$$

La somma degli a_i è il grado di D.

Jac($\mathbf{I}$) è l'insieme dei divisori D di $\mathbf{I}$ (a meno di una relazione di equivalenza).

Curve Iperellittiche

Jac($\mathbf{I}$) è un gruppo abeliano finito, che possiamo usare come gruppo per il problema del (DLP).

Non tutte le curve iperellittiche $\mathbf{I}$ vanno bene.

Per $g \geq 3$ ci sono metodi subesponenziali per il calcolo del logaritmo discreto.
Quindi a livello pratico si potrebbero usare curve iperellittiche di genere 2.

Curve Iperellittiche

Come nel caso delle curve ellittiche, è possibile migliorare l'efficienza e la sicurezza di un crittosistema scegliendo i parametri della curva in modo opportuno.

4. Crittografia con i Polinomi

Crittografia basata sui Polinomi

Supponiamo Alice  voglia trasmettere a Bob  il messaggio $M = (m_1,\ m_2,\ \ldots,\ m_n) \in \mathbb{F}_q{}^n$.

- Bob si costruisce un sistema di l polinomi in n variabili a coefficienti in $\mathbb{F}_q$ che sa risolvere:

$$\mathcal{A}: \begin{cases} f_1(x_1,\ x_2,\ \ldots,\ x_n) = 0 \\ f_2(x_1,\ x_2,\ \ldots,\ x_n) = 0 \\ \ldots \\ f_l(x_1,\ x_2,\ \ldots,\ x_n) = 0 \end{cases}$$

Questo costituisce la chiave pubblica.

Crittografia basata sui Polinomi

- Alice valuta M sul sistema $\mathcal{A}$
- Alice manda a Bob
  $(f_1(m_1,\ m_2,\ \ldots,\ m_n),\ \ldots,\ f_l(m_1,\ m_2,\ \ldots,\ m_n))$
- Bob sa risolvere $\mathcal{A}$ e calcola il messaggio iniziale

Crittografia basata sui Polinomi

La sicurezza di questo crittosistema dipende dal modo in cui il sistema è stato costruito, e dalla difficoltà, per chi non lo ha costruito, di risolverlo.
Un modo di costruirlo è quello di usare un sistema di equazioni con grado almeno 2.

$$\mathcal{A}' : \begin{cases} f_1(x_1,\ x_2,\ \ldots,\ x_n) = 0 \\ f_2(x_1,\ x_2,\ \ldots,\ x_n) = 0 \\ \ldots \\ f_l(x_1,\ x_2,\ \ldots,\ x_n) = 0 \end{cases}$$

Il sistema $\mathcal{A}'$ può essere visto come una funzione da ${\mathbb{F}_q}^n \longrightarrow {\mathbb{F}_q}^l$.

Crittografia basata sui Polinomi

Bob  sceglie due trasformazioni lineari affini $\mathcal{S}$ e $\mathcal{T}$:

$$\mathcal{S} : \mathbb{F}_q{}^n \longrightarrow \mathbb{F}_q{}^n \qquad \mathcal{T} : \mathbb{F}_q{}^l \longrightarrow \mathbb{F}_q{}^l$$

- chiave pubblica ($\mathcal{A} = \mathcal{S} \circ \mathcal{A}' \circ \mathcal{T}$)
- chiave privata ($\mathcal{S},\ \mathcal{A}',\ \mathcal{T}$)

Crittografia basata sui Polinomi

In fase di Cifratura:

$\xrightarrow{\mathcal{A}(M)}$

Per decifrare Bob deve conoscere $\mathcal{A}^{-1}$ ma:

$$M = \mathcal{A}'^{-1} \circ \mathcal{A}'(M) = \mathcal{A}'^{-1} \circ \mathcal{S}^{-1} \circ \mathcal{A}(M) \circ \mathcal{T}^{-1}$$

Hidden Field Equations (HFE)

Sia $X = (x_1,\ x_2,\ \ldots,\ x_n) \in (\mathbb{F}_q)^n$. $\mathcal{A}'$, detta mappa centrale, è del tipo:

$$\mathcal{A}'(X) = \sum_{q^{\phi_{i,j}} + q^{\theta_{i,j}} \leq d} \beta_{i,j} X^{q^{\phi_{i,j}} + q^{\theta_{i,j}}} + \sum_{q^{\xi_k} \leq d} \alpha_k X^{q^{\xi_k}}$$

Bob è in grado di invertire questa mappa in maniera veloce.

Però per un attaccante non è semplice trovare le trasformazioni $\mathcal{S}$ e $\mathcal{T}$ per i sistemi polinomiali $\mathcal{A}$ e $\mathcal{A}'$.

Tame Transformation Method (TTM)

Bob  sceglie 2 automorfismi triangolari:

$$\phi_1 : (\mathbb{F}_q)^n \longrightarrow (\mathbb{F}_q)^{n+r}$$

$$\phi_2 : (\mathbb{F}_q)^{n+r} \longrightarrow (\mathbb{F}_q)^{n+r}$$

e due traformazioni affini:

$$\mathcal{T} : (\mathbb{F}_q)^n \longrightarrow (\mathbb{F}_q)^{n+r}$$

$$\mathcal{S} : (\mathbb{F}_q)^{n+r} \longrightarrow (\mathbb{F}_q)^{n+r}$$

e con questi crea la chiave pubblica e quella privata.

Tame Transformation Method (TTM)

$$\phi_1 = \begin{cases} y_1 = x_1 \\ y_2 = x_2 + f_2(x_1) \\ \dots \\ y_n = x_n + f_n(x_1, \ \dots, \ x_{n-1}) \end{cases}$$

$$\phi_2 = \begin{cases} y_1 = x_1 + P(x_{n+1}, \ \dots, \ x_{n+r}) \\ y_2 = x_2 + Q(x_{n+1}, \ \dots, \ x_{n+r}) \\ y_3 = x_3 \\ \dots \\ y_{n+r} = x_{n+r} \end{cases}$$

dove P e Q sono polinomi con alcune particolari proprietà.

Tame Transformation Method (TTM)

La chiave publica che Bob manda ad Alice  è:

$$\mathcal{A} = \mathcal{S} \circ \phi_1 \circ \phi_2 \circ \mathcal{T}$$

Notiamo che Bob sa risolvere il sistema dato da $\mathcal{S} \circ \phi_1 \circ \phi_2 \circ \mathcal{T}$ in quando è la composizione di automorfismi triangolari e affini.

$\mathcal{A}$ è un automorfismo *tame* (cioè composizione di automorfismi triangolari e di trasformazioni affini).

Attacchi a Crittosistemi basati su Polinomi

Per attaccare un crittosistema costruito su polinomi possiamo:

- ricostruire direttamente la chiave segreta $(\mathcal{T},\ \mathcal{A}',\ \mathcal{S})$
- utilizzare le Basi di Groebner per risolvere il sistema $\mathcal{A}$

Risolvere un sistema di n equazioni in n variabili di grado 2 su un piccolo campo è molto spesso impraticabile: già con $n \sim 100$ non ci sono metodi migliori della ricerca esaustiva.

Attacchi a Crittosistemi basati su Polinomi

Per HFE non sono noti attacchi pericolosi, invece per l'algoritmo TTM esistono attacchi praticabili.

L'attacco all'algoritmo TTM consiste nel ridurre il problema al calcolo del MinRank.

Date m matrici $n \times n$ su $\mathbb{F}_q$: $\mathcal{M}_1,\ \mathcal{M}_2,\ \ldots,\ \mathcal{M}_m$, il problema del MinRank(r) consiste nel trovare una combinazione lineare α tale che:

$$Rank(\sum_{i=1}^{m} \alpha_i \mathcal{M}_i) \leq r$$

Uno dei migliori metodi per il calcolo del MinRank è il *Kernel Attack*.

Attacchi a Crittosistemi basati su Polinomi

E' stata rotta una *challenge* di TTM (TTM 2.1) con $r = 2$ in 2^{52}, che ha impiegato solo 3 minuti.
Rimangono comunque aperte altre *challenge*.

Rimane però il fatto che TTM usa speciali polinomi e non c'è una costruzione generale convincente.
La loro particolare struttura algebrica potrebbe essere una fonte di debolezza.

3. MCELIECE-NIEDERREITER

L'algoritmo McEliece-Niederreiter

Sia $\mathcal{C} \subset (\mathbb{F}_2)^n$ un codice binario in grado di correggere fino a t errori.
Questo codice è definito da:

$$\mathcal{C} = \{v \in (\mathbb{F}_2)^n | \mathcal{H}v = 0\}$$

con $\mathcal{H}$ matrice $k \times n$ di rango k.

Se il codice è algebrico, esiste un algoritmo $\mathcal{A}_{\mathcal{H}}$ veloce che quando riceviamo $y = c + e$ con $\omega(e) \leq t$, ci corregge gli errori partendo da $s = \mathcal{H}y$:

$$s \quad \xrightarrow{\mathcal{A}_{\mathcal{H}}} \quad (c,\ e)$$

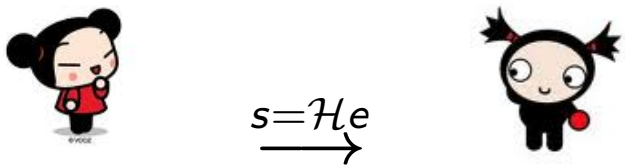
Cifratura e Decifratura con McEliece-Niederreiter

Bob sceglie un codice algebrico $\mathcal{C}$ e tramite questo definisce una chiave pubblica e privata:

- chiave pubblica $\mathcal{H}$
- chiave privata $\mathcal{A}$

I possibili plaintext che Alice può trasmettere sono i vettori in $(\mathbb{F}_2)^n$ con peso minore o uguale a t.

Se il messaggio è e:



Cifratura e Decifratura con McEliece-Niederreiter

Per decifrare, Bob riceve s e attraverso l'algoritmo $\mathcal{A}_{\mathcal{H}}$ si ricava e:

$$s \xrightarrow{\mathcal{A}_{\mathcal{H}}} e$$

Sicurezza con McEliece-Niederreiter

Ci sono codici algebrici $\mathcal{C}$ per cui solo data la matrice $\mathcal{H}$ risulta difficile dedurre $\mathcal{A}_{\mathcal{H}}$.

Solo con questa informazione, ricostruire la chiave segreta ha un costo esponenziale.