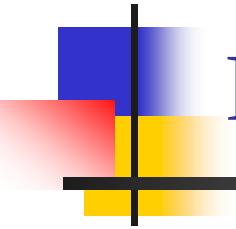
Corso di Crittografia

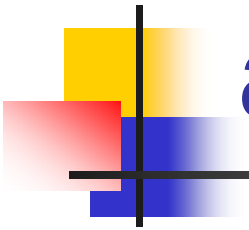
Prof. Dario Catalano



Cifrari Asimmetrici (Terza Parte):

RSA-OAEP e Cifrari basati sull'identita'

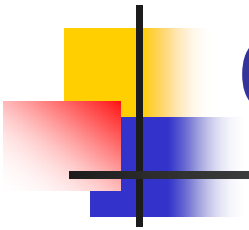
Cifrari sicuri contro attacchi attivi



- Fino ad oggi abbiamo visto cifrari sicuri contro avversari CPA, che “crollano” miseramente di fronte ad attacchi CCA.
- Oggi vedremo un esempio di cifrario sicuro contro attacchi di tipo attivo.
- Non dimostreremo che lo schema e’ sicuro
  - La dimostrazione e’ estremamente complessa

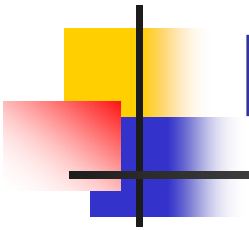
OAEP (Optimal Asymmetric Encryption Padding)

- Proposto inizialmente nel 1994.
  - Metodo per costruire un cifrario sicuro a partire da qualsiasi permutaz. Trapdoor (con qualche proprieta' addizionale)
- Nel 2001 e' stato scoperto un errore nella dimostrazione!
- Poco dopo la dimostrazione e' stata "riparata" ma solo per il caso RSA.



OAEP (cont.)

- RSA-OAEP e’ di fatto lo standard per cio’ che riguarda RSA encryption
  - PKCS#1 version 2, IEEE P1363
- La dimostrazione di sicurezza e’ (in un certo senso) euristica.
- RSA-OAEP e’ sicuro (assumendo la non invertibilita’ di RSA) nel cosiddetto *Random Oracle Model*.



Due domande

- Cosa rende cosi’ difficile costruire schemi sicuri contro attacchi attivi?
- Perche’ facciamo prove euristiche (nel Random Oracle Model) quando potremmo (e dovremmo) fare prove rigorose?

Difficolta’ di realizzare sicurezza CCA

Uno: Il simulatore deve essere in grado di decifrare senza essere in grado di risolvere il problema computazionale alla base della sicurezza del sistema.

Due: Dobbiamo riuscire ad impedire all’attaccante di creare crittotesti utili a partire da quello sul quale lo sfidiamo.

Un po’ di storia

- Il problema ha avuto diverse soluzioni teoriche (a partire dal 1991).
  - Tutte assolutamente improponibili in pratica
- Il primo schema pratico (che ammette una prova rigorosa) fu proposto nel 1998.
  - Cifrario Cramer-Shoup, basato su DDH

Il modello del Random Oracle

- Supponiamo di avere a disposizione una funzione hash che si comporta come una funzione casuale.
- Facciamo la dimostraz. in tale mondo immaginario.

Ipotesi del RO: Supponiamo che lo schema resti sicuro se, in pratica, usiamo una funzione hash.

Il modello RO (cont.)

Problema: Tale ipotesi e' falsa.

- Un numero notevole di controesempi sono stati proposti

Secondo Problema: Tutti questi controesempi sono del tutto artificiali.

Morale:

In teoria il RO non va bene, in pratica lo usano tutti.

RSA-OAEP - Preliminari

- k taglia del modulo RSA in uso.
- k_0, k_1 parametri ($k_0+k_1<k$)
- Spazio dei messaggi: $\{0,1\}^n$, $n=k-k_0-k_1$.
- Due funzioni hash (random oracles)

$$G:\{0,1\}^{k_0} \rightarrow \{0,1\}^{n+k_1}$$

$$H:\{0,1\}^{n+k_1} \rightarrow \{0,1\}^{k_0}$$

- L'algoritmo di generaz. della chiave e' quello di RSA

L'algoritmo di cifratura

Enc(N, m) // $m \in \{0,1\}^n$

$r \leftarrow_R \{0,1\}^{k_0}$;

$s \leftarrow G(r) \oplus (m||0^{k_1})$ // $s \in \{0,1\}^{n+k_1}$

$t \leftarrow H(s) \oplus r$; // $t \in \{0,1\}^{k_0}$;

$w \leftarrow s||t$; // $w \in \{0,1\}^k$

$y \leftarrow \mathtt{RSA}(w)$; // $y \in \{0,1\}^k$;

Return y

L'algoritmo di decifratura

Dec(N, y) // $c \in \{0,1\}^k$

$w \leftarrow \mathtt{RSA}^{-1}(y)$;

Sia $w = s||t$

// $s \in \{0,1\}^{n+k_1}$ $t \in \{0,1\}^{k_0}$;

$r \leftarrow H(s) \oplus t$;

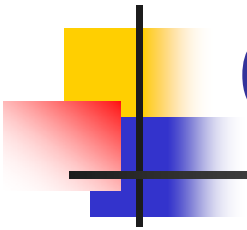
$z \leftarrow G(r) \oplus s$;

Sia $z = m||c$;

// $m \in \{0,1\}^n$ $c \in \{0,1\}^{k_1}$;

If $c == 0^{k_1}$ output m;

else output $\perp$.



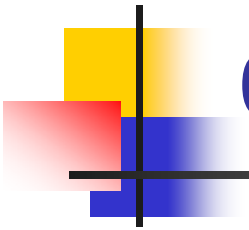
Commenti

- Nella prova di sicurezza il simulatore non conosce la fattorizzazione.
  - Potenza del Random Oracle!
- RSA-OAEP e’ praticamente altrettanto efficiente che standard RSA.
- Inoltre produce un crittotesto molto piccolo.

Identity Based Encryption

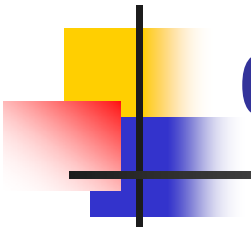
- Goal: crittosistema (asimmetrico) che permetta di utilizzare qualsiasi stringa come chiave pubblica.
- Cio' permetterebbe di semplificare la gestione di certificati.
- Es: Alice vuole mandare M a Bob bob@uni-bob.edu.
- Alice cifra M con la chiave bob@uni-bob.edu.
- Quando Bob riceve (il primo) messaggio cifrato, contatta l'opportuno ente autorizzato e ottiene la chiave privata corrispondente alla stringa bob@uni-bob.edu.

Cosa rende tanto speciale questa idea?



- Alice puo' inviare messaggi cifrati a Bob anche se questi non ha ancora generato la sua chiave.
- Chiavi a scadenza possono essere realizzate facilmente.
  - Alice cifra con bob@uni-bob.edu||Anno in corso
- IBE fornisce un metodo per creare facilmente chiavi temporanee.

Cosa rende tanto speciale questa idea? (cont.)



- Diventa (in certi contesti) piu' facile gestire il problema della revoca delle chiavi.
- Key Escrow e' risolto in modo automatico.
  - Ma sorge un probema di privacy delle chiavi.
- In realta' IBE ha tantissime altre applicazioni.

Lo schema Boneh-Franklin (2001) – I

- Utilizza mappe bilineri tra due gruppi G_1 e G_2 per gestire in modo efficiente la gestione di chiavi private locali.

Ulteriore Requisito: In G_1 una variante del problema Diffie-Hellman deve essere difficile.

- Esempi di tali mappe sono Weil e Tate pairing su curve ellittiche.

Lo schema Boneh-Franklin (2001) – II

- L’autorita’ T conosce una chiave segreta master, che permette di ricavare chiavi segrete locali.
- Senza la master key, e’ difficile fare altrettanto.
- La definizione di sicurezza per IBE e’ leggermente diversa da quella classica.

Lo scenario IB

- Chi attacca il sistema potrebbe gia' avere a disposizione un certo numero di chiavi private.
- Il sistema deve restare sicuro anche relativamente a questo tipo di attaccante.

Formalmente

- Formalizziamo questo requisito fornendo un nuovo oracolo all'avversario.
- Extract(ID) → SK_{ID}
- A puo' fare un numero arbitrario (limitato) di extract queries.
- A chiede di essere sfidato su una ID di sua scelta (sulla quale non e' autorizzato a fare extract queries)
- Il resto e' come gia' discusso.

Definizione (ind-id-cpa)

- IBE=(Setup,KeyDer,Enc,Dec)

$Esp^{ind\text{-}id\text{-}cpa\text{-}1}$ (A)	$Esp^{ind\text{-}id\text{-}cpa\text{-}0}$ (A)
$(pk,msk) \leftarrow_R$ Setup	$(pk,msk) \leftarrow_R$ Setup
$b \leftarrow A^{Enc_{pk}(id^*,LR(.,.,1)),KeyDer_{msk}(.)}$	$b \leftarrow A^{Enc_{pk}(id^*,LR(.,.,0)),KeyDer_{msk}(.)}$
If A imbroglia Return 0	If A imbroglia Return 0
else return b	else return b

$Adv^{ind\text{-}id\text{-}cpa}(A)=Pr[Esp^{ind\text{-}id\text{-}cpa\text{-}1}(A)=1] - Pr[Esp^{ind\text{-}id\text{-}cpa\text{-}0}(A)=1]$

A imbroglia se interroga $KeyDer_{msk}(.)$ sull'identità id*.

Funzioni Bilineari

Siano G_1, G_2 e G_T gruppi con le seguenti proprieta'.

(1) G_1 e G_2 sono gruppi di ordine q (primo). G_1 e' generato da P e G_2 e' generato da P'.

(2) Esiste un isomorfismo $\rho : G_2 \rightarrow G_1$, $\rho(P') = P$.

(3) Esiste una mappa bilineare $e : G_1 \times G_2 \rightarrow G_T$.

Bilinearita': $\forall U \in G_1, V \in G_2, a, b \in Z$,
$$e(U^a, V^b) = e(U, V)^{ab}.$$
Non degenere: $e(P, P') \neq 1_{G_T}$
Efficienza: Le operazioni in G_1,G_2 e G_T, ρ e e sono efficientemente calcolabili.

Il cifrario Boneh-Franklin

Setup: T sceglie msk (uniformemente) a caso in [1,...,q].

La chiave pubblica e' $Pk=(P')^{msk}$.

Key Extraction: La chiave segreta dell'utente id e'

$P_{id}=H_1(id)$;

$usk=P_{id}^{msk}$

Il cifrario Boneh-Franklin

Enc$(m, \mathtt{id}, Pk)$

$r \leftarrow_R \{1, \ldots q\}$; $P_{\mathtt{id}} \leftarrow H_1(\mathtt{id})$

$k \leftarrow e(P_{\mathtt{id}}, Pk)^r$;

$c_1 \leftarrow (P')^r$; $c_2 \leftarrow m \oplus H_2(k)$

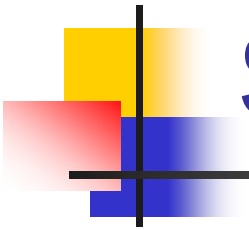
Return (c_1, c_2)

Dec$((c_1, c_2), \mathtt{id}, usk)$

$k \leftarrow e(usk, c_1)$; $m \leftarrow c_2 \oplus H_2(k)$

Return m

Sicurezza di BF



- Il cifrario BF e' sicuro (nel ROM) relativamente all'ipotesi Diffie-Hellmann bilineare decisionale.
- Esiste anche un secondo schema BF, sicuro (sempre nel ROM) relativamente ad attacchi attivi

Problema Diffie-Hellman Bilineare Decisionale

- Versione “bilineare” di DDH

Dati: P^a, P^b, $P^c \in G_1$, e t della forma

$$t=e(P,P)^{abc}$$

oppure

$$t \in_R G_T$$

Goal: Determinare di che “tipo” e’ t