LA SICUREZZA DELLA FIRMA DIGITALE
STATO E PROSPETTIVE

Armando Leotta

La firma digitale si basa su algoritmi, procedure e schemi ritenuti (computazionalmente) sicuri in un contesto molto diverso da quello in cui operano attualmente. Per una vera conoscenza dello stato dell'arte è opportuno comprendere quali siano le vulnerabilità, i contesti operativi in cui valutare le debolezze degli algoritmi utilizzati valutando i potenziali impatti nel medio termine e i rimedi immediati. Tante le soluzioni proposte a livello internazionale che anticipano soluzioni evolutive innovative come nuove funzioni *hash* e algoritmi basati su curve ellittiche.

3.7

1. INTRODUZIONE

I meccanismi di funzionamento della firma digitale, così come prescritta dal nostro ordinamento, poggiano essenzialmente sugli algoritmi crittografici a chiavi pubbliche detti anche a chiavi asimmetriche poiché utilizzano chiavi diverse per le operazioni di cifratura e decifratura.
Le basi di questo nuovo sistema furono poste nel 1976 da due studiosi, Whitfield Diffie e Martin E. Hellman, *New directions in cryptography*, che elaborarono un protocollo per lo scambio di una chiave segreta sopra un canale pubblico.
Esso segna il primo rivoluzionario progresso nella crittografia moderna.
Gli algoritmi asimmetrici (per esempio Diffie-Hellman, RSA, El Gamal) sono utilizzati in generale per le applicazioni in cui sono richieste la confidenzialità, l'autenticazione e la distribuzione delle chiavi.
In particolare la **firma digitale** fornisce:
- Autenticazione;
- Integrità;
- Non-Ripudio.

2. IMPIEGO DELLE CHIAVI CRITTOGRAFICHE

Le chiavi crittografiche possono essere utilizzate in modi diversi a seconda che lo scopo sia quello di cifrare il messaggio oppure apporvi una firma digitale. La differenza tra le due applicazioni risiede nel ruolo delle chiavi. Qualora si intenda spedire a qualcuno un messaggio cifrato, verrà applicata la chiave pubblica del destinatario all'intero messaggio. In tal modo il testo diviene illeggibile e può essere inviato. Il ricevente sarà in grado di leggere il messaggio solo dopo averlo decifrato apponendovi la propria chiave privata. Nel caso il messaggio arrivasse per errore ad un destinatario diverso, questo non sarebbe in grado di leggerlo perché non in possesso della chiave privata corretta cioè la corrispettiva chiave privata relativa alla chiave pubblica utilizzata nella fase di cifratura.
La firma digitale è pertanto una stringa che associa un messaggio a un originatore. L'algoritmo di generazione di firma digitale è un metodo per produrre una firma digitale. Lo schema di firma digitale consiste di un algo-

ritmo di generazione e di un algoritmo di verifica della firma digitale.
Nella figura 1 è illustrato il modello di funzionamento degli algoritmi asimmetrici.
Ciascun utente genera una coppia di chiavi (chiave pubblica, chiave privata). La chiave pubblica è resa pubblica, quella segreta viene custodita gelosamente. Se Bob desidera inviare un messaggio ad Alice, Bob cifra il messaggio utilizzando la chiave pubblica di Alice. Quando Alice riceve il messaggio, lo decifra utilizzando la sua chiave privata. L'autenticazione con gli algoritmi asimmetrici avviene secondo il modello raffigurato nella figura 2.
Quando si vuole apporre una firma digitale ad un messaggio, il mittente utilizza invece la propria chiave privata, che non deve essere necessariamente applicata a tutto il testo.
L'applicazione di un algoritmo di *encryption* produce [1] un *output* dimensionalmente paragonabile all'*input* trattato. Pertanto, nel caso di documenti corposi la cifratura a chiave pubblica può diventare lenta e particolarmente onerosa da un punto di vista computazionale.

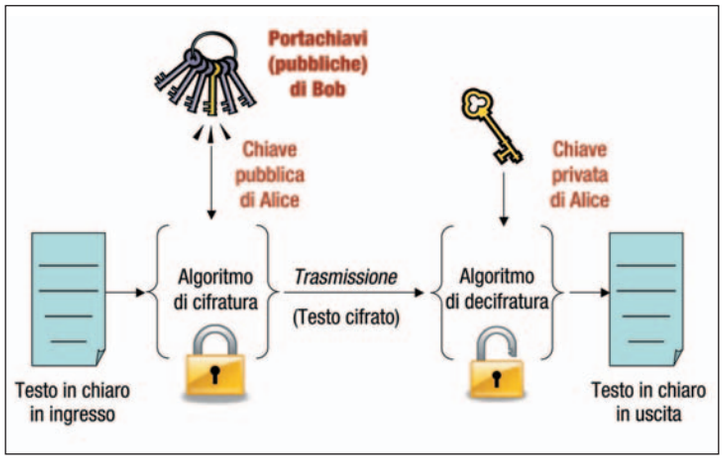

FIGURA 1
Cifratura

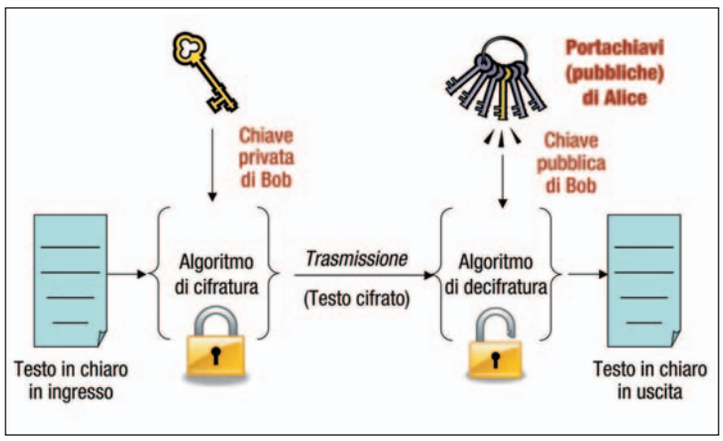

FIGURA 2
Autenticazione/verifica/decifratura

3. RUOLO DELLA FUNZIONE DI *HASH* NEL PROCESSO DI FIRMA DIGITALE

Nel nostro caso l'interesse è unicamente volto all'apposizione di una firma digitale [2] su un documento (e non alla cifratura dello stesso), stiamo pertanto ponendo attenzione agli aspetti di autenticazione e non di riservatezza.
A tal fine viene usata una particolare funzione matematica, chiamata funzione di *hash* che "comprime" il testo in una sorta di "riassunto", che viene anche definito "impronta digitale" (*finger printing* o *message digest*).
Il testo risultante è progettato in modo da minimizzare la probabilità che da testi diversi si possa ottenere il medesimo valore dell'impronta. La dimensione del riassunto è fissa e molto più piccola di quella del messaggio originale, pertanto la generazione della firma risulta molto più rapida.
Quest'ultima consiste quindi semplicemente nella cifratura per mezzo della chiave segreta dell'impronta digitale, generata tramite la funzione di *hash*. La firma è poi aggiunta in una posizione predefinita, normalmente alla fine del testo del documento. Solitamente, insieme con la firma vera e propria, è allegato al documento anche il valore dell'impronta digitale. Nella figura 3 viene schematizzato il processo.
Devono essere inoltre rilevabili attraverso la firma digitale gli elementi identificativi del soggetto titolare della firma, del soggetto che l'ha certificata e del registro sul quale essa è stata pubblicata. La verifica della firma digitale sarà poi fatta dal destinatario applicando la medesima funzione di *hash* usata nella fase di sottoscrizione in modo da ottenere il valore dell'impronta. Tale valore sarà poi confrontato con quello che si ottiene decodificando la firma digitale stessa applicandovi la chiave pubblica del mittente. La disponibilità del valore dell'impronta all'interno del messaggio ha solo la funzione di rendere più veloce la verifica.

Con tale procedura si possono quindi inviare documenti non cifrati e sottoscritti con firma digitale. Questi hanno dunque provenienza certa e colui che li ha inviati non potrà disconoscere di avere inviato il messaggio (non ripudio). È inoltre garantita l'integrità del documento in quanto ogni modifica anche minima apportata al documento firmato in modo digitale è immediatamente riconoscibile in quanto il processo di verifica non si conclude positivamente.
È anche possibile cifrare il contenuto del documento applicando in questo caso la chiave pubblica del destinatario al testo in modo che solo lui potrà decifrarlo utilizzando la corrispondente chiave segreta.

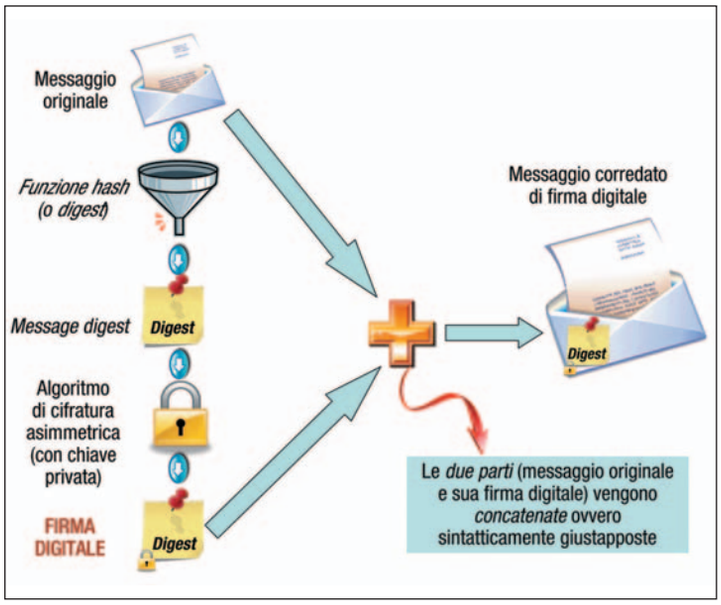

FIGURA 3
Processo di firma

4. HASH

Le funzioni *hash* svolgono un compito fondamentale per l'efficienza degli algoritmi di crittografia e di firma digitale in quanto trasformano una stringa in ingresso di qualsiasi dimensione in un *ouput* di dimensione fissata detto *digest* del messaggio.
Esse sono particolarmente efficienti da un punto di vista computazionale in quanto utilizzano funzioni molto semplici, prevalentemente lo XOR la cui proprietà è quella di "*appiattire*" e "*camuffare*" l'input su cui vengono effettuate le operazioni. Tutte le funzioni *hash* hanno in genere un tempo di esecuzione lineare rispetto all'input, qualsiasi modifica venga fatta al dato in ingresso causerà la produzione di un valore *hash* completamente diverso e sarà sempre un risultato deterministico di lunghezza finita.
Le funzioni *hash* rappresentano una classe di funzioni i cui comportamenti possono essere schematizzate come indicato nella figura 4. La figura rappresenta ad alto livello la struttura di una funzione *hash*.
L'input viene passato ad una funzione di compressione (simbolo a forma di imbuto) che ne riduce la dimensione. Quando la dimensione della computazione parziale è stata ridotta adeguatamente è possibile applicare una trasformazione finale che determinerà l'output finale.
In questo documento, ci concentreremo sulle funzioni *hash* [3] senza chiave.
Tali funzioni possono essere di due tipi:
- funzioni *hash one way* (OWHFs);
- funzioni *hash collision resistant* (CRHFs).

Esse hanno particolari peculiarità matematiche, ovvero sono non invertibili (le OWHF - *pre-image* e 2nd *pre-image resistant*) e sono resistenti alle collisioni (le CRHF). La figura 5 riassume la schematizzazione delle funzioni *hash*.
Affermare che una funzione matematica gode della proprietà di *pre-image resistance*,

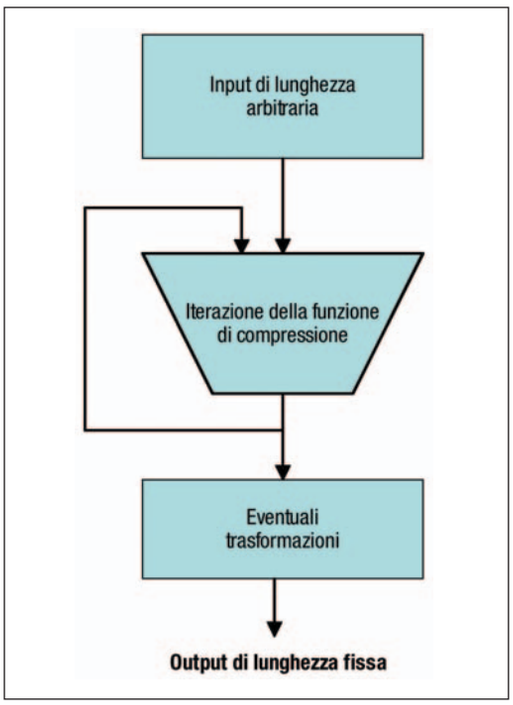

FIGURA 4
Funzione Hash

ovvero la caratteristica di non essere invertibile, significa che dato *y* (= target *hash*) e indicando con *h* la funzione *hash*, è computazionalmente impossibile determinare un valore *x* tale che *h* (*x*) = *y*.
Informalmente, questa caratteristica corrisponde alla difficoltà di trovare un secondo valore input che abbia lo stesso *hash* del primo (non noto).
Un'altra caratteristica importante è la 2nd *pre-image resistant*: dati *x* e *y* = *h* (*x*) e, indicando con *h* la funzione *hash,* è computazionalmente impossibile determinare un valore *x'* tale che *h* (*x'*) = *h* (*x*).
Informalmente, questa caratteristica corrisponde alla difficoltà di trovare un secondo valore input che abbia lo stesso *hash* del primo valore.
Le funzioni *Hash* resistenti alle collisioni (CRHFs) sono particolari funzioni *hash* che oltre alla proprietà di 2nd *pre-image resistance* hanno la proprietà di rendere computazionalmente difficile trovare due input distinti *x* e *x'* i quali generano lo stesso output [*h* (*x*) = *h* (*x'*)].
Essendo la lunghezza del *digest* (output della funzione di *hash*) sensibilmente minore della lunghezza del dato che lo produce (principio della piccionaia), le collisioni sono inevitabili ma le suddette caratteristiche matematiche rendono le collisioni difficili da creare e da gestire e conseguentemente le funzioni *hash* computazionalmente sicure.

5. COLLISIONI

Input diversi possono produrre il medesimo valore di *hash*? Se fossimo in grado di creare due dati in ingresso che generano entrambi lo stesso *digest* (Figura 6) saremmo *potenzialmente* nelle condizioni di cambiare il contenuto di un documento prima che esso venga consegnato alla funzione di *hash* senza che quest'ultima modifichi il *digest* prodotto.
Per mettere a fuoco l'importanza della proprietà *collision resistance* [4] è importante ricordare che nel procedimento di firma digitale quanto viene firmato per motivi di opportunità e di performance non è l'intero messaggio ma il *digest* dello stesso. Tale *digest* viene appunto creato tramite una funzione di *hash*.

Cosa succederebbe se, ipoteticamente, cambiassi un messaggio firmato digitalmente con un altro avente lo stesso *digest*? Nulla, i messaggi pur essendo diversi, avendo lo stesso *digest*, la firma apposta risulterebbe valida e il processo di verifica della stessa sarebbe superato con successo.
Studi matematici e di criptanalisi riportano gli obiettivi progettuali di sicurezza nonché l'onere computazionale necessario a scardinare una determinata proprietà della funzione di *hash* (Tabella 1).
Diremo in generale che un attacco di complessità 2^t richiede approssimativamente 2^t operazioni unitare di riferimento, l'esecuzione di una funzione di compressione, una cifratura ecc..

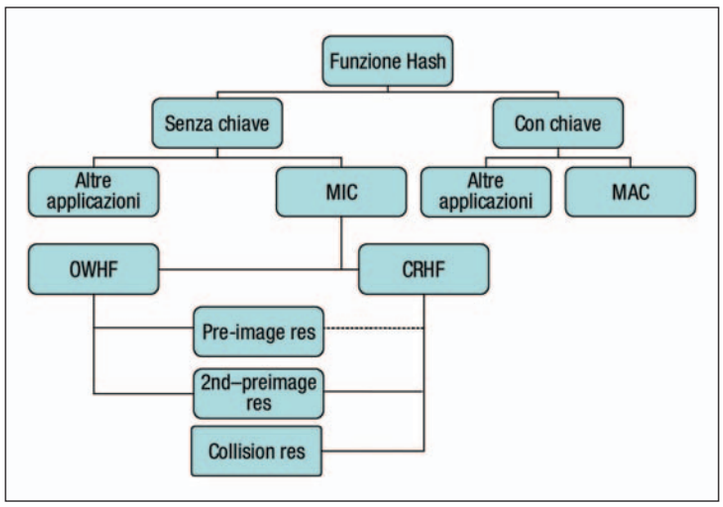

FIGURA 5
Le funzioni hash

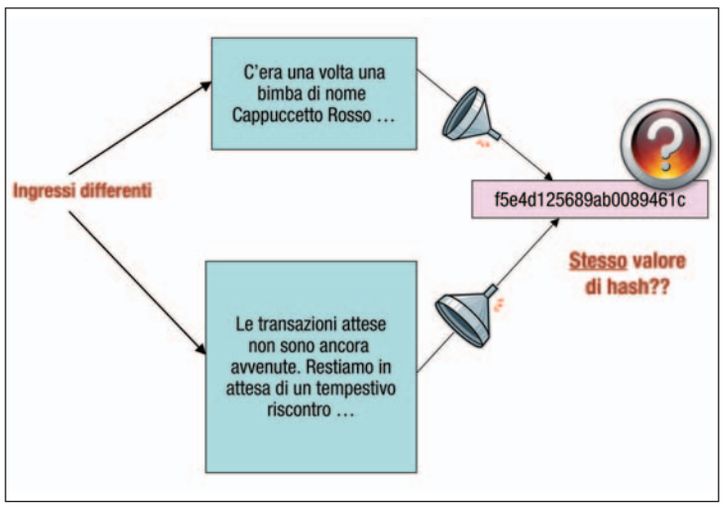

FIGURA 6
Collisioni

Tipo di Funzione Hash	Obiettivo del progettista	Robustezza ideale	Obiettivo dell'avversario
OWHF	Preimage resistance	2^n	Produrre preimage (immagine)
	2^{nd} preimage resistance	2^n	Trovare un secondo input, stessa immagine
CRHF	Collision resistance	$2^{n/2}$	Produrre collisioni

TABELLA 1

Robustezza delle funzioni hash

Per un avversario capace di scegliere i messaggi, un attacco secondo la tecnica del compleanno (*birthday attack*) permette di trovare collisioni in $2^{n/2}$ operazioni.

6. CRITICITÀ E CONTESTO

L'attuale processo di firma digitale prevede l'utilizzo dell'algoritmo di *hash* SHA-1 [5]. Come vedremo, è proprio la debolezza di questo algoritmo a minare la robustezza dell'intero processo di firma digitale.
Esso genera 160 bit di *digest* quindi avremo una collisione al costo di $2^{160/2} = 2^{80}$ operazioni.
Ricordando che, in linea generale, la sicurezza di un algoritmo crittografico è legata al numero di istruzioni necessarie a rompere il sistema, alla capacità computazionale dei sistemi di calcolo, nonché alle risorse disponibili, è determinante porsi un quesito: è ragionevole ritenere ancora sicuro il suddetto limite e quindi computazionalmente irraggiungibile?
Se sì, fino a quando?
Già nel 2005, sul blog di Bruce Schneier [**1, 2**], guru del settore, un team di ricercatori cinesi coordinati dalla Prof.ssa Wang della Shandong University [6, **3, 4**] ha presentato alcuni studi che pubblicavano come riuscire a produrre collisioni nello SHA-1 in 2^{63} operazioni, un fattore 2^{17} inferiore, ben oltre 131000 volte meno oneroso di quanto ritenuto sicuro.
Inoltre, il contesto odierno (*clustering, griding, multicore, parallel processing and distributed computing* ecc.) fa continuamente registrare un aumento sensibile delle performance di calcolo accompagnato da un altrettanto importante abbattimento dei costi sostenuti.
In particolare, nel campo crittografico, occorre segnalare l'impiego [7] delle *Graphical Processing Unit* (GPU), ovvero le unità di elaborazione grafica delle nuove schede grafiche anche nei normali home computer, per elaborazioni crittografiche attraverso il *porting* di opportune applicazioni riscritte con routine grafiche (DirectX, OpenGL).
AMD e Nvidia hanno rilasciato dei tool di sviluppo dedicati (SDK CUDA e AMCL).
Il risultato è un innalzamento consistente delle performance.
Il *Center For Visual Computing and Department of Computer Science* dell'Università Stony Brook di NewYork ha pubblicato uno studio nel quale emergono i benefici di utilizzare un cluster di moderne GPU per impegnativi calcoli scientifici paralleli [8].
I risultati, decisamente interessanti per performance e rapporto GFlops/costo, hanno alimentato ulteriori interessanti studi per accelerare calcoli computazionali di natura non necessariamente grafica.
Poche settimane scorse, la Elcomsoft [**5, 6**], società di software russa con sede a Mosca, ha registrato un brevetto negli Stati Uniti che protegge una tecnica di *password cracking* delle password di accesso ai computer utilizzando le caratteristiche di *parallel processing* delle più recenti GPU in commercio.
Tale tecnica ha fatto registrare un abbattimento di circa 25 volte le risorse temporali necessarie ad una moderna CPU.
Alla luce di questo contesto tecnologico, siamo ancora in piena sicurezza computazionale?
E ancora: se sì, fino a quando?
È possibile isolare la criticità (la funzione di *hash* SHA-1) dal resto del processo di firma digitale? Sarebbe sufficiente o andrebbe affrontato un nuovo schema di firma?

7. INIZIATIVE INTERNAZIONALI

Il *National Institute of Science and Technology* (NIST) americano ha già identificato degli standard per funzioni *hash* di lunghezza e resistenza maggiore: la cosiddetta famiglia SHA-2 (SHA-224, SHA-256, SHA-384 e SHA-512).
L'adozione di un algoritmo della famiglia SHA-2 rappresenta una rapida ed efficace soluzione nel medio e breve termine fino a che i risultati della *call for algorithm* [**7**] non porti i frutti

Utilizzo accettabile fino alla fine del 2009	Uso con certificati qualificati[1]: utilizzo accettabile fino alla fine del 2010	Utilizzo accettabile fino alla fine del 2010	Utilizzo accettabile fino alla fine del 2011
SHA-1	SHA-1	RIPEMD-160	SHA-224, SHA-256, SHA-384, SHA-512

[1] Esclusivamente per ila generazione e la verifica di certificati ma non per la generazione e verifica di altri dati qualificati firmati.

TABELLA 2
Funzioni hash e tempi di "sicurezza"

Fino alla fine del 2007	Fino alla fine del 2008	Fino alla fine del 2009	Fino alla fine del 2010	Fino alla fine del 2011
1024 (minimo)	1280 (minimo)	1563 (minimo)	1728 (minimo)	1976 (minimo)
2048 (raccomandato)	**2048** (raccomandato)	**2048** (raccomandato)	**2048** (raccomandato)	**2048** (raccomandato)

TABELLA 3
Criticità sulla lunghezza chiave RSA

attesi e cioè un nuovo standard condiviso da tutta la comunità internazionale.
Nel marzo del 2006, il governo tedesco ha pubblicato nella gazzetta ufficiale [**8**] quelli che considera tempi di sicurezza per gli algoritmi e le funzioni *hash* utilizzati nel procedimento di firma digitale (Tabella 2).
Oltre alla criticità circa l'utilizzo dello SHA-1 dopo il 2009, viene evidenziata (Tabella 3) la seria preoccupazione del governo tedesco per la lunghezza della chiave RSA [9].
Analoghe preoccupazioni sono sorte in seno al governo austriaco ed in Europa (progetto Nessie [10], [**9**, **10**]).
La *National Security Agency* (NSA) americana presenta la sua raccomandazione, Suite B [**11**] una lista di algoritmi crittografici ritenuti sicuri.
Ad analoghe conclusioni giungono in Europa (progetto NESSIE) e in Giappone (CRYPTREC *Evaluation Committee* [**12**, **13**, 11]).

8. SOLUZIONI IMMEDIATE E TRANSITORIE

"*Aumentare la lunghezza della chiave (per gli algoritmi crittografici, ad esempio RSA) e la dimensione del digest prodotto dalla funzione di hash*".
Queste misure, pur essendo concettualmente e matematicamente valide vanno valutate attentamente in quanto introducono un problema non indifferente di performance spostando solamente di qualche anno il problema.
Per le funzioni *hash* il problema è meno invasivo e l'adozione della famiglia SHA-2 [12, **14**] da 224 a 512 bit risulta un compromesso accettabile ed immediato.
Le medesime considerazioni valgono per l'algoritmo di *hash* WHIRLPOOL [**15**] a 512 bit.
Entrambi gli algoritmi indicati rientrano tra quelli considerati sicuri e adottabili nei diversi studi indipendenti precedentemente citati.

9. NUOVI SCENARI CRITTOGRAFICI

Gli algoritmi basati su curve ellittiche (ECC) [13, 14, 15, 16, 17, **16**] sono legati matematicamente al problema del logaritmo discreto (riquadro a p. 28) e rappresenta un approccio condiviso, robusto, efficace, efficiente e computazionalmente scalabile.
Di seguito, la tabella 4 (fonte NSA [**17**]) che mostra come a "sicurezza equivalente" l'adozione degli algoritmi basati su curve ellittiche riducano sensibilmente la dimensione della chiave rispetto ad algoritmi come RSA e Diffie-Hellman.

10. CONCLUSIONI

È importante chiarire che non vi è una reale emergenza in atto.
Quello che occorre però evitare è ignorare i vari segnali d'allarme e l'elevato livello d'attenzione oramai presente in tutto il settore. Come detto, esistono diverse soluzioni e rimedi temporanei immediati e fortemente consigliati.

Il problema del logaritmo discreto

Il problema del logaritmo discreto è, insieme ad altri, come per esempio quello della fattorizzazione di interi, un problema matematico definito in termini di aritmetica modulare, estremamente importante nell'ambiente crittografico.
Fissato un numero primo p e un intero g tra 0 e $p-1$ si ottiene y dalla relazione:

$y = g^x \pmod p$

Il problema del logaritmo discreto è di determinare l'intero x dati g, y e p.
Per questo problema non è stato trovato nessun algoritmo efficiente in grado di risolverlo.
Taher ElGamal è stato il primo a proporre un sistema crittografico basato su questo problema. In particolare, propose due distinti sistemi: uno schema crittografico e uno schema di firme digitali (DSA [**23**] si basa sul lavoro di ElGamal).
Rompere uno di questi schemi significa risolvere il problema del logaritmo discreto.
Per avere una buona sicurezza il primo p deve essere lungo almeno 230 cifre decimali (760 bit).
Le prestazioni del sistema dipendono dalla velocità di esecuzione dell'esponenziazione modulare.
Il calcolo dominante in ogni trasformazione è:

$g^x \pmod p$

con g compreso tra 0 e $p-1$.
In conclusione la sicurezza di questi sistemi, si basa sul problema del logaritmo discreto modulo p, mentre l'efficienza dipende dalla velocità di esecuzione dell'esponenziazione modulare.

Lunghezza della chiave simmetrica (bit)	Lunghezza della chiave RSA e Diffie-Hellman (bit)	Lunghezza della chiave a curva ellittica (bit)
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512

TABELLA 4

Confronto a "sicurezza equivalente"

Sebbene coinvolga anche algoritmi non espressamente affrontati in questo contributo per ovvie ragioni di sintesi e contesto, risulta di estremo interesse la tabella 5, un quadro sinottico che mette a confronto le raccomandazioni frutto di studi indipendenti da parte degli studiosi di tutto il mondo.
Lo SHA-2 in prima istanza per le funzioni di *hash* e l'ECDSA [18] [**18**, **19**, **20**, **21**] (*Elliptic Curve Digital Signature Algorithm*), come schema di firma digitale, rappresentano soluzioni consolidate, sicure, robuste e innovative.
Si pensi al mercato della connettività mobile, dei produttori di chip: adottare ECC significa puntare sul moderno, sullo scalabile, sulle prestazioni con dimensioni decisamente più contenute, dunque risorse hardware inferiori ma con prestazioni superiori.
Nonostante i numerosi vantaggi delle curve ellittiche e la crescente diffusione tra gli utenti, molti nel mondo accademico e industriale ritengono un ostacolo al loro utilizzo alcuni fattori legati alla proprietà intellettuale che sottendono tali curve.
In effetti, vari aspetti della crittografia a curve ellittiche sono stati brevettati sia da individui che da società.
La Certicom Inc. per esempio, è una società canadese che detiene oltre 130 brevetti relativi alle curve ellittiche ed alla crittografia a chiave pubblica in generale.
La NSA ha stabilito un accordo [**22**] con la Certicom circa la proprietà intellettuale e l'utilizzo degli algoritmi brevettati.
Come si evince anche dal suddetto accordo (*agreement*), Certicom ha identificato 26 brevetti che riguardano l'utilizzo. La licenza accordata alla NSA include il diritto di sub-licenza di questi 26 brevetti a produttori incaricati di costruire dei dispositivi nel pieno rispetto della licenza e dei limiti di utilizzo accordati.
Certicom, da parte sua, ha mantenuto il diritto medesimo sia all'interno dell'ambito d'utilizzo che diversamente in base ad una negoziazione diretta con i produttori (*vendor*).
Questi aspetti e la diversa normativa in materia di brevetti su algoritmi (in molte nazioni non è consentito brevettare algoritmi) rappresentano fattori sicuramente non bloccanti ma che non facilitano l'introduzione massiva sul mercato.
Tutti gli esperti del settore sono concordi nel considerare il 2010 una *deadline* per imprimere una svolta ed un cambiamento strutturale nel mondo della crittografia asimmetrica.
È opportuno pertanto considerare seriamente, anche da un punto di vista di mercato produttivo e governativo, l'alternativa offerta dalle curve ellittiche per i vantaggi computazionali, di prestazione e di traffico offerti a parità di sicurezza e robustezza.
Inoltre, essendo le lunghezze delle chiavi sensibilmente minori rispetto alla prima generazione, le soluzioni ECC risultano essere anche più facilmente scalabili nel prossimo futuro.
In Italia, le nuove Regole Tecniche della Firma Digitale sono ad oggi in itinere.
Già nel novembre scorso, le corrispettive del-

		NIST FIPS e SP (considerando un uso fino alla fine del 2030)	Lista di cifrari raccomandati dal CRYPTREC per l'e-government	Cifrari raccomandati (NESSIE)	SO/IEC 18033
Cifrari asimmetrici	Firma	❑ RSA come specificato in ANS X9.31 (2048) ❑ DSA (2048) ❑ ECDSA (224)	❑ DSA (1024) ❑ ECDSA (160) ❑ RSASSA-PKCS1-v1_5 (1024) ❑ RSA-PSS (1024)	❑ RSA-PSS (1536) ❑ ECDSA (160) ❑ SFLASH	(Specificato nell'ISO/IEC 9796-2 e 14888-3)
	Confidenzialità	Nessuna racc.	❑ RSA-OAEP (1024) ❑ RSAES-PKCS1-v1_5 (1024)	❑ PSEC-KEM (160) ❑ RSA-KEM (1536) ❑ ACE-KEM	❑ RSA-KEM ❑ RSA_OAEP ❑ HIME(R) ❑ ACE-KEM ❑ PSEC-KEM ❑ ECIES-KEM
	Scambio chiavi	❑ DH ❑ MKV	❑ DH (1024) ❑ ECDH (160) ❑ PSEC-KEM (160)	Nessuna racc.	(Specificato nell'ISO/IEC 11770-3)
Cifrari simmetrici	Cifrari a blocchi da 64 bit	Triple-Des (a tre chiavi)	❑ CIPHERUNICORN-E ❑ HIEROCRYPT-L1 ❑ MISTY1 ❑ TRIPLE-DES (a tre chiavi)	MISTY1	❑ CAST-128 ❑ MISTY1 ❑ TRIPLE-DES (racc. a tre chiavi)
	Cifrari a blocchi da 128 bit	AES	❑ AES ❑ CAMELLIA ❑ CIPHERUNICORN-A ❑ HIEROCRYPT-3 ❑ SC2000	❑ AES ❑ CAMELLIA	❑ AES ❑ CAMELLIA ❑ SEED
	Cifrari a flusso	Nessuna racc.	❑ MUGI ❑ MULTI-S01 ❑ RC4 (128 bit)	Nessuna racc.	❑ MUGI ❑ SNOW 2.0
Funzioni HASH		❑ SHA-224 ❑ SHA-256 ❑ SHA-384 ❑ SHA-512	❑ RIPEMD-160 ❑ SHA-1 ❑ SHA-256 ❑ SHA-384 ❑ SHA-512	❑ SHA-256 ❑ SHA-384 ❑ SHA-512 ❑ WHIRLPOOL	(Specificato nell'ISO/IEC 10118)

Nota: Le lunghezze delle chiavi in tabella sono espresse in bit ed indicano i valori minimi raccomandati

TABELLA 5

Sintesi e raccomandazioni

la Carta d'Identità Elettronica DM 8/11/2007 nell'allegato B formalizzava l'abbandono dell'RSA a 1024 bit (minimo 2048 o 3072 bit) e dello SHA-1 in favore dello SHA-2.

Inoltre, per la prima volta, viene effettuata la formale apertura verso l'ECDSA con curve raccomandate da 224 e 283 bit.

È stato appena pubblicato sulla *Gazzetta Uf-*

ficiale n. 129 del 6 giugno il DPCM del 30 marzo "Regole tecniche in materia di generazione, apposizione e verifica delle firme digitali e validazione temporale dei documenti informatici". Non appena entrerà in vigore (180 giorni dalla pubblicazione) abrogherà il DPCM del 13 gennaio 2004 recante le "regole tecniche per la formazione, la trasmissione, la conservazione, la duplicazione, la riproduzione e la validazione, anche temporale, dei documenti informatici", pubblicato sulla *Gazzetta Ufficiale* 27 aprile 2004, n. 98.
Estremamente importante l'art. 2 comma 3 che conferma al CNIPA il pieno controllo delle regole tecniche:

"*Gli algoritmi di generazione e verifica delle firme, le caratteristiche delle chiavi utilizzate, le funzioni di hash, i formati e le caratteristiche dei certificati qualificati, le caratteristiche delle firme digitali e delle marche temporali, il formato dell'elenco di cui all'articolo 39 del presente decreto, sono definiti, anche ai fini del riconoscimento e della verifica del documento informatico, con deliberazioni del CNIPA e pubblicati sul sito internet dello stesso Centro nazionale*".

Sarà suo compito pertanto armonizzare le eventuali modifiche del CAD (*Codice dell'Amministrazione Digitale*) e proiettarci in Europa dove la sicurezza (per esempio, il formale impiego di algoritmi basati su curve ellittiche e l'impiego della famiglia SHA-2 per le funzioni di *hash*) e l'interoperabilità dei formati (per esempio, l'uso della sottoscrizione in linguaggio XML, i formati della busta crittografica e di firma come XML e PDF e altri conformi a specifiche pubbliche *Publicly Available Specification* -PAS-) rappresentano aspetti cruciali dell'*e-government* internazionale dei prossimi anni.

Bibliografia

[1] Handbook of Applied Cryptography, Menezes, Ooschot, Vanstone.

[2] Manca G.: Il futuro della firma digitale. *Mondo Digitale*, 2002,

[3] Stinson D.: *Cryptography: theory and practice*. Chapman & Hall.

[4] *Introduction to computer security*. Bishop 2005

[5] *Secure Hash Standard*. SHA-1, FIPS 180-1, NIST.

[6] Wang X.: *Cryptanalysis for Hash Functions and Some Potential Dangers*. RSA Conference, 2006, Cryptographer's Track, 2006 - Xiaoyun Wang, Yiqun Yin, Hongbo Yu, Finding Collisions in the Full SHA-1, Crypto'05 - Collision Search Attacks on SHA1, 2005.

[7] Debra L. Cook, John Ioannidis, Angelos D. Keromytis, Jake Luck: *CryptoGraphics: Secret Key Cryptography Using Graphics Cards*.

[8] *GPU Cluster for High Performance Computing*. ACM/IEEE Supercomputing Conference 2004, November 06-12, Pittsburgh, PA.

[9] Rivest R., Shamir A., Adleman L.: A method for obtaining digital signatures and public key cryptosystems. *Communications of the ACM*, Vol. 21, n. 2, 1978.

[10] *NESSIE è un progetto della Information Society Technologies (IST) Programme della Commissione Europea*. (Key Action II, Action Line II.4.1).

[11] *Year 2010 Issues on Cryptographic Algorithms*. Masashi Une and Masayuki Kanda, March 2007.

[12] NIST: *FIPS Publication 180-2: Secure Hash Standard (SHS-1)*. August 2002.

[13] Miller V.: *Uses of Elliptic Curves in Cryptography*. Advances in Cryptology, CRYPTO '85, Proceedings, Lecture Notes in Computer Science 218, Springer-Verlag, 1986, p. 417-426.

[14] Koblitz N.: Elliptic Curve Cryptography. *Mathematics of Computation*, Vol, 48, 1987, p. 203-209.

[15] Menezes A., Vanstone S.A.: Elliptic curve cryptosystems and their implementation. *Journal of Cryptology*, Vol. 6, 1993, p. 209-224.

[16] Gathani Amit N.: *Implementation of Elliptic Curve Cryptography in Embedded System*. 2001.

[17] *Advances in Elliptic Curve Cryptography, Series*. London Mathematical Society Lecture Note Series (No. 317), Edited by Ian F. Blake, University of Toronto, Gadiel Seroussi, Hewlett-Packard Laboratories, Palo Alto, California, Nigel P. Smart, University of Bristol.

[18] ISO 14888 ANSI X9.62 Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA), X9.63 Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography.

Webgrafia

[1] http://www.schneier.com/essay-074.html

[2] http://www.schneier.com/blog/archives/2005/02/sha1_broken.html

[3] http://news.zdnet.com/2100-1009_22-5598536.html

[4] http://www.infosec.sdu.edu.cn/people/wangxiaoyun.htm

[5] http://www.elcomsoft.com

[6] http://www.elcomsoft.com/EDPR/gpu_en.pdf GPGPU

[7] http://www.nist.gov/public_affairs/techbeat/tb2007_1108.htm#sha

[8] http://www.bundesnetzagentur.de/media/archive/5952.pdf

[9] https://www.cosic.esat.kuleuven.be/nessie/

[10] https://www.cosic.esat.kuleuven.be/nessie/deliverables/decision-final.pdf

[11] http://www.nsa.gov/ia/industry/crypto_suite_b.cfm

[12] http://www.ipa.go.jp/security/enc/CRYPTREC/index-e.html CRYPTREC Giappone

[13] http://www.imes.boj.or.jp/english/publication/mes/2007/me25-1-6.pdf

[14] http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf

[15] http://paginas.terra.com.br/informatica/paulobarreto/WhirlpoolPage.html

[16] The Basics of ECC. http://www.certicom.com

[17] http://www.nsa.gov/ia/industry/crypto_elliptic_curve.cfm

[18] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=1439

[19] X9 committee http://www.x9.org

[20] CERTICOM Patent letter : http://publicaa.ansi.org/-sites/apdl/Patent%20Letters/PL337.pdf

[21] IEEE P1363 2 http://www.ieee.org/

[22] NSA & CERTICOM Agreement http://www.certicom.com/download/aid-501/FAQThe%20NSA%20ECC%20License%20Agreement.pdf

[23] http://www.itl.nist.gov/fipspubs/fip186.htm Digital Signature Standard.

Armando Leotta è laureato in Scienze dell'Informazione presso La Sapienza di Roma. Nel 2008 nello stesso Ateneo e Dipartimento ha conseguito il Master di II livello in Gestione della Sicurezza informatica per l'impresa e la Pubblica Amministrazione, concluso con un tirocinio trimestrale presso il CNIPA. Certificato IRCA-RICEC Auditor / Lead Auditor per Sistemi di Gestione per la Sicurezza delle Informazioni (SGSI) a norma ISO/IEC 27001:2005 & BS 7799-2:2002, Certificato OMG UML Professional e OCEB (OMG Certified Expert in BPM), APM Group ITIL v3 e QRP Prince2, dal 2003 si occupa di Gestione della sicurezza e Management dei processi afferenti la sicurezza informativa nella Ragioneria Generale dello Stato. Nel 2009 conclude con AICA un percorso formativo che lo conduce alla certificazione EUCIP Professional/Elective IS Auditor.
E-mail: armando.leotta@gmail.com, info@armandoleotta.it