Università degli studi di Perugia

Facoltà di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea Magistrale in Informatica

Sicurezza Informatica

Protocollo E-cash e RSA

Carlo Manasse

Giulio Baldantoni

Anno Accademico 2011-2012

Indice

1 Moneta elettronica **3**
1.1 Introduzione 3
1.2 Caratteristiche 4

2 Protocollo E-cash **6**
2.1 Introduzione 6
2.2 Funzionamento 7
2.3 Conclusioni 11

3 Algoritmo RSA **12**
3.1 Introduzione 12
3.2 Funzionamento 12
3.2.1 Generazione delle chiavi 12
3.2.2 Cifratura 13
3.2.3 Decifrazione 13
3.3 Correttezza dell'algoritmo 14
3.3.1 Teoremi e proprietà utili per la dimostrazione 14
3.3.2 Dimostrazione 14
3.4 Riservatezza, autenticità e non ripudiabilità 15
3.4.1 Firma elettronica 16
3.5 Blinded signatures in RSA 17

Capitolo 1

Moneta elettronica

1.1 Introduzione

Il denaro, ovvero la materia prima su cui si basa l'attività delle banche, sta man mano estinguendo. Gli assegni e le carte di credito ne hanno ridotto la quantità fisicamente circolante, ma la sua completa eliminazione è ancora lontana. D'altra parte, assegni e carte di credito, permettono spesso di invadere la privacy delle persone.
Oltre a queste ragioni è la sempre maggiore diffusione di Internet che ha permesso la creazione di un vero commercio elettronico (e-commerce), accompagnato da una maggiore sicurezza e rapidità con cui avvengono i pagamenti online.
Gli acquisti effettuati attraverso questo sistema prevedono ovviamente un pagamento elettronico, cioè un pagamento virtuale senza passaggio fisico di denaro. L'approccio che permette oggi di effettuare queste transazioni monetarie durante la navigazione in Internet è la Moneta Elettronica. Questo è un vero e proprio titolo di credito digitale firmato da una banca o da una organizzazione non bancaria di cui l'utente può usufruirne in maniera autonoma, ovviamente dove il servizio è offerto.
La moneta elettronica essendo digitale consta di bit e può essere trasmessa con facilità attraverso la rete. Come già menzionato sono vari i vantaggi di un tale strumento, quello principale è di garantire l'anonimato di colui che acquista, nonché l'immediatezza della transazione e la facilità di utilizzo, garantendo inoltre la non tracciabilità del compratore.
Infatti questi requisiti sono considerati utili, soprattutto per i pagamenti elevati, dal momento che il compratore in genere non vuole l'intervento di terzi nelle sue transazioni, inoltre questo sistema offre la non tracciabilità in modo che il compratore non venga identificato, quindi anche diversi pagamenti dello stesso non possono essere collegati tra loro.
Il requisito della non tracciabilità può essere mascherato verso l'esterno utilizzando opportune codifiche delle transazioni ed usando protocolli sicuri per la comunicazione in rete, però il problema fondamentale sembra consistere nella necessità per gli attori della transazione di essere correntisti della stessa istituzione bancaria. Inoltre c'è da non trascurare la necessità di rivolgersi alla stessa istituzione che ha emesso la moneta per convertire la stessa in denaro contante.

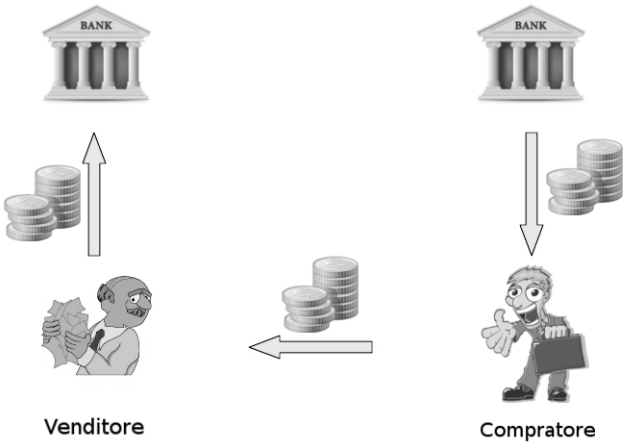


Figura 1.1:

Infatti tutto questo avviene mediante moderni sistemi che gestiscono elettronicamente la moneta, chiamati Sistemi Digital Cash.

1.2 Caratteristiche

Le caratteristiche principali che un sistema Digital Cash deve possedere, sono :

- SICUREZZA: si cerca di trovare un modo per evitare la falsificazione della moneta elettronica che, essendo rappresentata da stringhe di bit, risulta facilmente duplicabile;
- ANONIMATO: è importante per chi utilizza moneta elettronica proteggere la propria privacy, in particolare evitare tracce sul percorso della moneta che facciano risalire a chi l'ha usata, mediante l'uso di opportuni protocolli;
- SCALABILITÁ: un sistema è scalabile se gestisce l'aumento degli utenti e delle risorse senza perdita di prestazioni. In genere l'uso di un server centrale, attraverso il quale avvengono le transazioni, limita la scalabilità del sistema, così pure come l'uso di meccanismi per individuare il riutilizzo della moneta (in quanto bisogna memorizzare per ogni utente molte informazioni relative alla moneta con un conseguente aumento della dimensione del database);
- ACCETTABILITÁ: è importante che la moneta elettronica emessa da una banca sia accettata dalle altre, quando si effettua un cambio di valuta, la loro riconciliazione deve avvenire in modo automatico;

- TRASFERIBILITÁ: è necessario che la moneta elettronica sia accettata da terzi, senza contattare la banca. In questo modo, viene valorizzata la proprietà di anonimato, anche se ciò complica il meccanismo che garantisce la sicurezza;
- INDIPENDENZA DELL'HARDWARE: per il riutilizzo delle monete durante le operazioni off-line, alcuni protocolli sfruttano dell'hardware particolare che protegge da possibili intrusioni/manomissioni;
- TIPOLOGIE DI PAGAMENTO: le modalità di pagamento sono diverse, come l'utilizzo di moneta elettronica opportunamente coniata, assegni elettronici, smart card e carte di credito;
- COSTI DI GESTIONE: i costi di gestione variano fondamentalmente in funzione del grado di sicurezza che viene offerto;Le differenti problematiche da affrontare per gestire correttamente le transazioni hanno portato alla creazione di diversi protocolli, più o meno efficaci a seconda del loro utilizzo.

Attualmente, sono pochi i sistemi che forniscono anonimato e non tracciabilità nei confronti di venditore e fornitore/acquirente; Ricordiamo tra i più conosciuti : E-cash e CAFE.

Analizziamo più in dettaglio nel prossimo capitolo il protocollo E-cash Electronic Cash Payment Protocol.

Capitolo 2

Protocollo E-cash

2.1 Introduzione

E-cash (Electronic Cash Payment Protocol) è un sistema di pagamenti sicuri per Internet elaborato da Digicash società fondata dal Dott. David Chaum del Center for Mathematics and Computer Science con sede ad Amsterdam, e reso operativo in collaborazione con la Mark Twain Bank di St. Louis.
Il sistema di pagamenti Ecash è stato introdotto nel 1993 dal Dott. David Chaum fondatore, nel 1990, della Digicash Corporation, azienda produttrice del software ECash.
L'innovazione è dovuta al fatto che non è un sistema di transazioni basato su carte di credito: Ecash è un interessante esempio di come può essere sfruttato il denaro virtuale, che viene fornito da diverse banche mondiali associate, attraverso l'utilizzo di monete elettroniche. Tali banche sono responsabili della certificazione e dell'autenticità delle monete virtuali di Ecash.
Il sistema è stato progettato e sviluppato allo scopo di consentire transazioni economiche veloci e sicure da un qualsiasi dispositivo attraverso la rete, permettendo di effettuare compravendite durante la navigazione.
Il software ECash è di tipo client-server e lavora con tutte le maggiori piattaforme: MS Windows, MAC e UNIX.
L'utente una volta procuratosi il software Ecash-client ed aperto un conto presso una delle banche aderenti all'iniziativa, scarica sul proprio dispositivo del denaro elettronico spendibile in tutti i negozi che accettano moneta elettronica con Ecash; tali monete, al momento di un acquisto, verranno trasferite al venditore sfruttando tecniche di crittografia a chiave pubblica e di firma digitale.

2.2 Funzionamento

Tre partecipanti sono coinvolti nel modello Ecash:

1. Clienti:
   - Hanno il software Ecash (cyberwallet) sui propri dispositivi.
   - Possono usare E-coins del loro portafoglio per fare acquisti dai commercianti.
   - Ritirano le monete dai loro conti presso le banche associate.
   - Memorizzano e gestiscono tutte le proprie transazioni.
2. Commercianti:
   - Accettano ed elaborano i pagamenti.
   - Interagiscono con una banca associata per eseguire la convalida e l'autenticazione.
   - Vendono oggetti e generano entrate.
3. Banche:
   - Istituzioni associate con E-cash dove i clienti e i commercianti hanno i conti.
   - Gestiscono i conti dei clienti e commercianti.

Nella seguente immagine ne viene mostrato il funzionamento di base, per semplificare consideriamo solo una banca:

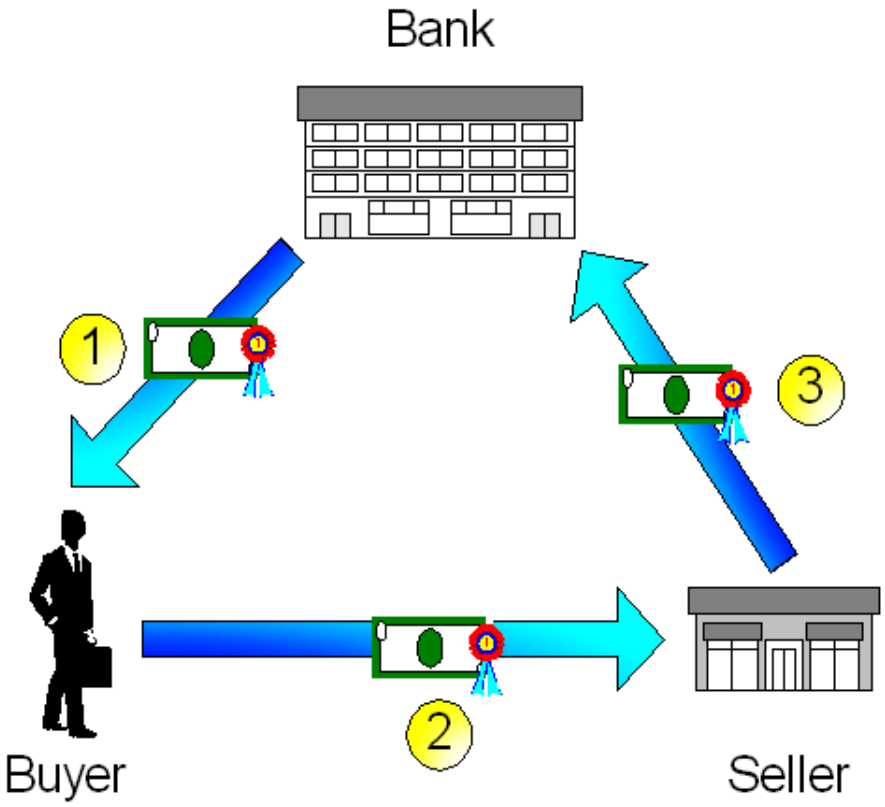


Figura 2.1:

1. L'acquirente riceve un biglietto firmato digitalmente dalla banca.
2. L'acquirente passa la sua nota a un venditore.
3. Il venditore deposita la sua nota nel suo conto in banca.

Procedura di pagamento con Ecash :

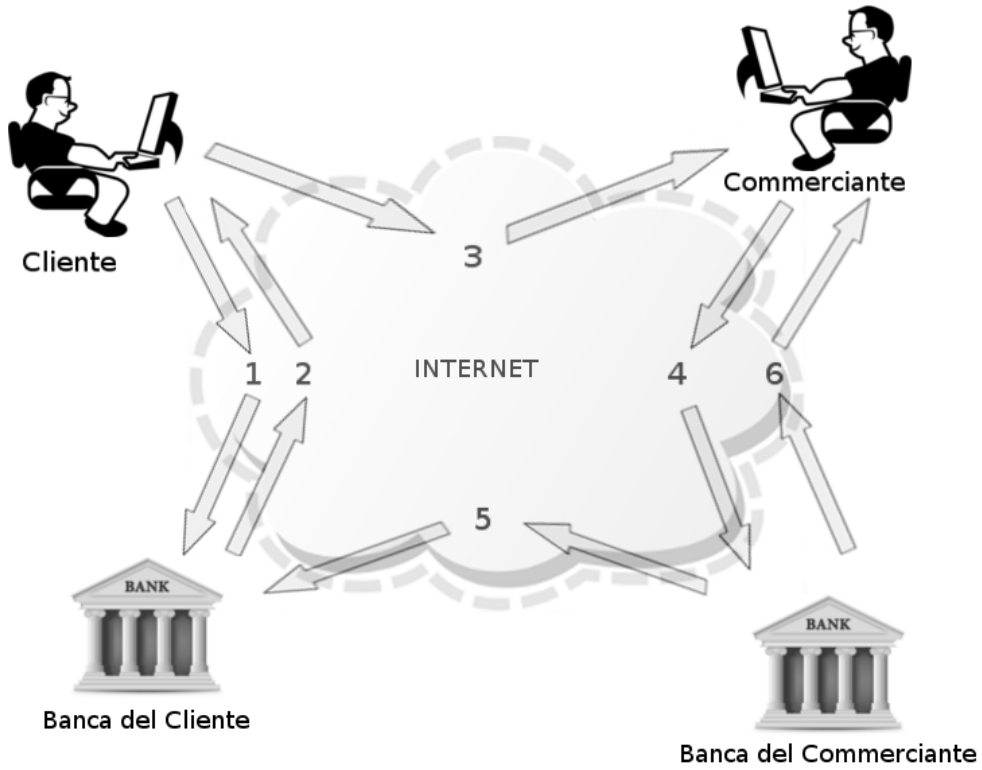


Figura 2.2:

1. Il Cliente invia la moneta codificata con il numero di serie per la Banca.
2. La Banca rimanda indietro al Cliente la moneta firmata.
3. Il Cliente decodifica il numero di serie ed usa la moneta per pagare.
4. Il Commerciante manda questa moneta alla propria Banca.
5. Le due Banche effettuano la verifica sulla moneta.
6. Il Commerciante riceve la conferma e l'accredito.

Tutto il funzionamento del sistema si basa sulla presenza on-line di banche associate presso le quali gli utenti possono aprire conto correnti virtuali, ai quali viene fornita una chiave pubblica necessaria per la codifica della firma dell'acquirente che vuole effettuare un pagamento tramite la rete.
Gli utenti e i venditori aderenti non devono possedere hardware particolari, le banche invece dovranno disporre di un particolare hardware di codifica che ne assicuri la velocità e l'affidabilità delle operazioni.
La sicurezza e la riservatezza delle operazioni è garantita dall'uso di tecniche di firma digitale a chiave pubblica. L'utente preleva denaro virtuale dal proprio

conto con una password che protegge i dati su Ecash.
Ogni utente inoltre possiede un'unica coppia di chiavi che vengono generate automaticamente dal software al suo primo utilizzo, questo garantisce la sicurezza delle transazioni e dei messaggi inviati.
L'anonimato di colui che paga è anch'esso garantito, può però rendere nota la propria identità solo se decide di farlo. Invece non gode di anonimato colui che riceve il pagamento, infatti, durante la fase di compensazione, il beneficiario di una transazione è identificato dalla banca.
La riservatezza delle transazioni è garantita dalla tecnica della firma cieca : quando l'utente ha necessità di effettuare un pagamento deve avere il denaro virtuale sul proprio dispositivo, deve cioè aver fatto un prelievo presso la banca dove ha il conto;
questo prelievo non si configura tecnicamente come un semplice trasferimento di moneta virtuale tra la banca e il dispositivo dell'utente, infatti è il software Ecash che una volta calcolato quante monete sono necessarie per ottenere la somma richiesta, crea tali monete assegnando ad ognuna di esse, in modo causale, un numero di serie.
Poi spedisce queste monete alla banca, ognuna delle monete viene inserita in una particolare busta che rappresenta il fattore cecità; la banca codifica i numeri ciechi con la propria chiave segreta applicando la firma digitale attraverso la busta e addebita la somma prelevata sul conto dell'utente.
Le monete sono dunque restituite all'utente che potrà togliere il fattore cecità introdotto senza alterarne la firma della banca, cosi i numeri di serie rappresentano ora la moneta digitale il cui valore è garantito dalla stessa banca.
Quando in seguito l'utente effettuerà delle spese, la banca accetterà le monete perché le ha firmate anche se non sarà in grado di riconoscere chi ha effettuato il pagamento.

2.3 Conclusioni

PRO	CONTRO
Permette pagamenti su Internet Peer-to-peer Anonimo Adatto per micro-pagamenti Gratuito Non c'è bisogno di terzi Leggero	Non è adatto per grandi quantità Non è Internazionale

Poiché non è necessario alcun materiale speciale (come card reader), ogni cliente può agire sia come acquirente (l'invio di denaro) o come un venditore (ricezione denaro). E questo punto è davvero fondamentale, un sistema che mantiene i vantaggi di denaro convenzionale sarà ovviamente più facilmente accettato.
Anche se tutto quello che il sistema offre sembra essere molto promettente, il suo successo non si è evoluto come ci si aspettava.
Difatti nel 1998 la Digicash fallisce ed Ecash sparisce dalla scena. Alcuni imputano il fallimento al modo in cui i sistemi sono stati implementati : richiedendo troppi step durante la configurazione del software client-server, per un utente medio. Altri attribuiscono il fallimento alla mancata definizione di uno standard, data la molteplicità di implementazioni concorrenti. Altri ancora pensano che alcune istituzioni abbiano spinto per non far decollare Ecash, viste le problematiche introdotte dalla non tracciabilità dei pagamenti. Difatti difficilmente si possono controllare gli acquisti, fare profili dei consumatori ed analisi di marketing nonché scovare azioni fraudolente o reati in generale.

Capitolo 3

Algoritmo RSA

3.1 Introduzione

In letteratura esistono numerosi algoritmi di crittografia asimmetrica che permettono di effettuare comunicazioni sicure. Tra questi l'RSA è senz'altro uno dei più famosi e viene utilizzato frequentemente nelle telecomunicazioni, in software di uso comune quali ad esempio ssh e pgp, nonché in applicazioni militari. In particolare in questo contesto RSA rappresenta il garante della sicurezza e della riservatezza nel protocollo Ecash.
RSA deve il suo nome alle iniziali dei cognomi di tre ricercatori del MIT (Massachusetts Institute of Technology), Ronald Rivest, Adi Shamir e Leonard Adleman che lo hanno teorizzato nel 1976 e pubblicamente descritto nel 1978. L'RSA permette di cifrare e decifrare un messaggio basandosi sulla creazione di due chiavi distinte, una pubblica ed una privata, generate a partire dal prodotto n di due numeri primi. La forza dell'algoritmo, che ne ha decretato un così grande successo, risiede nel fatto che sebbene le due chiavi siano dipendenti, non è possibile risalire dall'una all'altra, se non fattorizzando n. La fattorizzazione in numeri primi è un problema non-polinomiale, e ciò significa che per numeri n molto grandi richiede tempi e risorse di calcolo enormi.

3.2 Funzionamento

L'algoritmo RSA si basa su tre passaggi fondamentali:

1. creazione delle chiavi
2. cifratura del messaggio
3. decifrazione del messaggio

3.2.1 Generazione delle chiavi

Come già accennato RSA lavora utilizzando una chiave pubblica, che può essere conosciuta da chiunque, utilizzata per cifrare il messaggio. I messaggi possono essere decifrati solamente attraverso l'utilizzo dell'altra chiave, quella privata, che deve deve essere segretamente custodita.

1. Il meccanismo di generazione delle due chiavi comincia a partire da due numeri primi p e q scelti in modo casuale e abbastanza grandi da garantire tempi di fattorizzazione molto alti. Date le attuali potenze dei computer, si utilizzano generalmente numeri con almeno 300 cifre;

2. Scelti p e q si calcola il loro prodotto $n = pq$. Questo sarà usato come modulo per entrambe le chiavi;

3. Si calcola il valore $\varphi(n) = (p-1)(q-1)$, cioè la funzione di Eulero (totiente) calcolata in n. Questa funzione è definita, per ogni $n \in \mathbb{N}$, come

$$|\{y \in \mathbb{N} \text{ tali che } y < n \text{ e } \ \mathrm{MCD}(n, y) = 1\}|,$$

   cioè il numero degli y minori di n che sono coprimi con n. Poiché p e q sono primi e quindi $\varphi(p) = p-1$ e $\varphi(q) = q-1$, si ha che $\varphi(pq) = (p-1)(q-1)$, in virtù del fatto che la funzione φ è moltiplicativa.

4. A questo punto si sceglie un numero intero e compreso tra 1 e $\varphi(n)$ tale che $\mathrm{MCD}(\varphi(n), e) = 1$, i.e $\varphi(n)$ ed e sono coprimi. Il valore e è detto esponente pubblico e la coppia (e, n) rappresenta la chiave pubblica del sistema.

5. Ottenuto e occorre determinare il valore d (detto esponente privato) che farà parte della chiave privata rappresentata dalla coppia (d, n). Questo si calcola come l'inverso moltiplicativo di e modulo $\varphi(n)$, cioè d è tale che $d \equiv e^{-1} \bmod \varphi(n)$ (equivalentemente $de \equiv 1 \bmod \varphi(n)$).

3.2.2 Cifratura

Ora che si hanno a disposizione le due chiavi è possibile cifrare un messaggio applicando le seguenti regole. Si consideri per semplicità uno scambio di messaggi tra due utenti A e B, in particolare B vuole mandare un messaggio cifrato ad A. Si può supporre che B sia a conoscenza della chiave pubblica (e, n) di A, o perché questa è scritta in un elenco o perché A gliel'ha mandata in un precedente messaggio (in chiaro).
Si supponga, sempre per semplicità, che il messaggio sia il valore intero m. Questo è possibile per il fatto che qualsiasi insieme di caratteri è facilmente rappresentabile con valori interi (generalmente si utilizza un procedimento detto padding scheme). Il testo cifrato c si calcola applicando la formula

$$c \equiv m^e \bmod n.$$

Il valore così ottenuto può essere inviato da B ad A.

3.2.3 Decifrazione

A, una volta ricevuto il messaggio c cifrato con la sua chiave pubblica, può decifrarlo per leggere il messaggio. Questo passaggio avviene con un procedimento del tutto simile alla cifratura. Il messaggio m di fatto viene riottenuto applicando la formula

$$m \equiv c^d \bmod n.$$

Solo A può decifrare m poiché la chiave (d, n) è mantenuta segretamente da A.

3.3 Correttezza dell'algoritmo

3.3.1 Teoremi e proprietà utili per la dimostrazione

La prova della correttezza di RSA si basa sul piccolo teorema di Fermat. Sia p un numero primo e a un intero non divisibile da p, il teorema afferma che

$$a^{(p-1)} \equiv 1 \bmod p.$$

La dimostrazione del teorema non sarà discussa in questa sede.
Inoltre si farà uso di un importante risultato del teorema cinese dei resti, il quale afferma che la relazione $a \equiv b \bmod pq$ risulta equivalente al sistema

$$\begin{cases} a \equiv b \bmod p \\ a \equiv b \bmod q \end{cases}$$

3.3.2 Dimostrazione

Ciò che si vuole mostrare è che dati due numeri primi p, q, due interi e, d calcolati come sopra ed m, valga la relazione $(m^e)^d \equiv m \bmod pq$. Per il teorema cinese dei resti ciò equivale a dire che valgono entrambe le relazioni $(m^e)^d \equiv m \bmod p$ e $(m^e)^d \equiv m \bmod q$.

Dalle proprietà dei moduli la relazione

$$ed \equiv 1 \bmod \varphi(n)$$

può essere riscritta come

$$ed - 1 = h(p-1)(q-1)$$

per qualche intero h non negativo.

Per mostrare che valga $(m^e)^d \equiv m \bmod p$ si considerano due casi:

- $m \equiv 0 \bmod p$
  Nel primo caso m^{ed} è multiplo di p, quindi si ha $m^{ed} \equiv 0 \equiv m \bmod p$
- $m \not\equiv 0 \bmod p$
  Nel secondo caso si ha

  $$m^{ed} = m^{ed-1}m = m^{h(p-1)(q-1)}m = (m^{p-1})^{h(q-1)}m \equiv m \bmod p$$

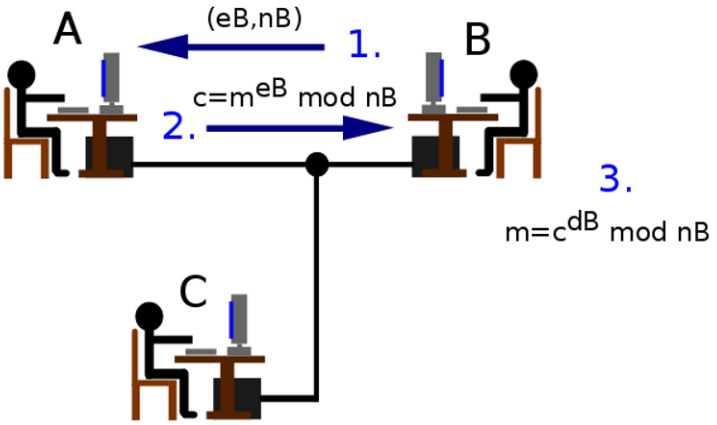
  In virtù del piccolo teorema di Fermat $m^{p-1} \equiv 1 \bmod p$ quindi

  $$(m^{p-1})^{h(q-1)}m \equiv 1^{h(q-1)}m \equiv m \bmod p$$

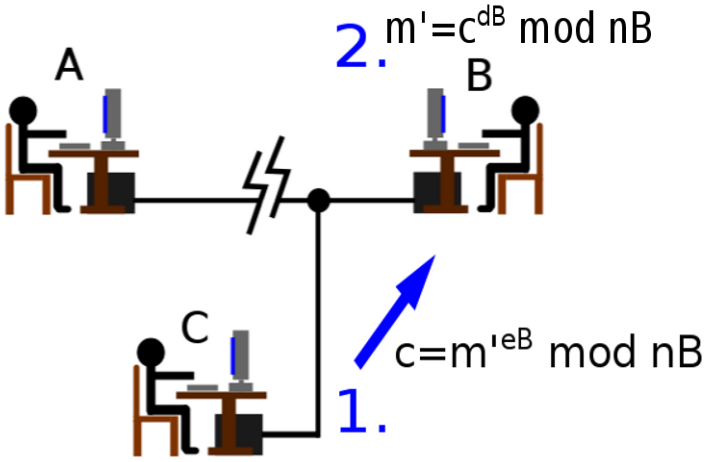
Verificare che valga $(m^e)^d \equiv m \bmod q$ è del tutto equivalente, trattando separatamente i due casi $m \equiv 0 \bmod q$ e $m \not\equiv 0 \bmod q$.
Dunque per ogni intero m si è dimostrato che $(m^e)^d \equiv m \bmod pq$ e cioè che decrittando un messaggio m cifrato con la chiave pubblica (e, n), tramite la chiave privata (d, n) si ottiene il messaggio m originale.

3.4 Riservatezza, autenticità e non ripudiabilità

Come si è visto fino ad ora, l'algoritmo RSA permette agli utilizzatori di effettuare comunicazioni sicure nel senso che la riservatezza è preservata. Si consideri il seguente schema in cui ci sono tre attori A,B e C. Si considerino inoltre le chiavi pubbliche e private (eA, nA),(dA, nA) e (eB, nB),(dB, nB) rispettivamente di A e di B.



C naturalmente non può leggere il messaggio m che A ha spedito a B. Ma come fa B a sapere chi effettivamente ha generato m? Si consideri il seguente scenario:

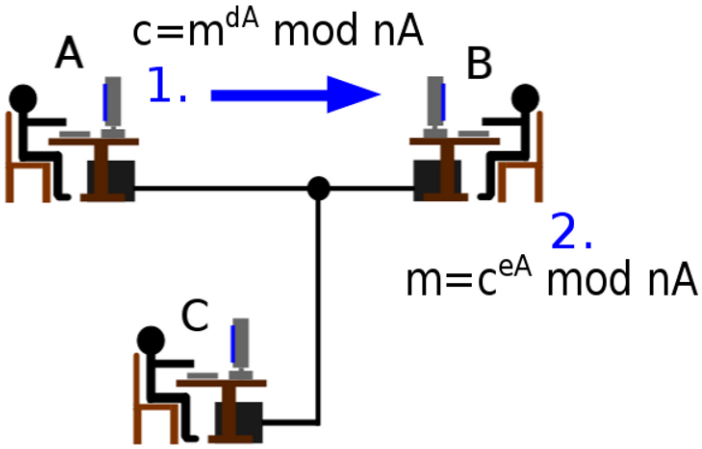


C intercetta m e si contrappone nella comunicazione tra A e B. C genera un messaggio m' che cifra con la chiave pubblica di B, e lo spedisce a B fingendo che m' sia stato generato da A. In questo caso B non ha gli strumenti per verificare

l'autenticità dei messaggi.
RSA permette facilmente di superare questi limiti.

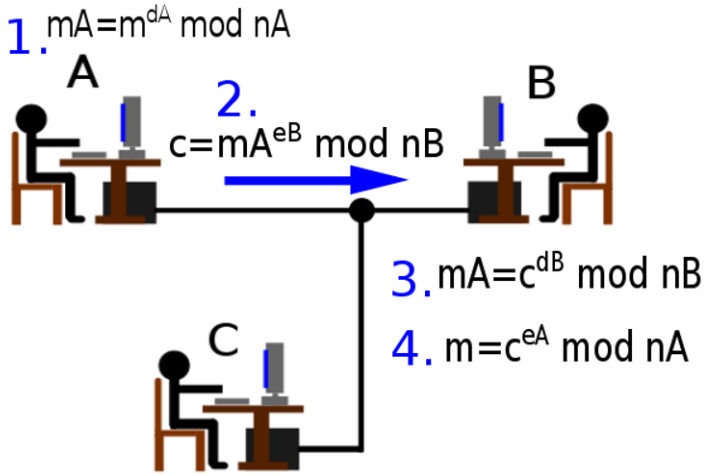
3.4.1 Firma elettronica

Autenticare un messaggio in RSA risulta molto simile alla cifratura. Considerando l'esempio precedente, A prima di spedire il messaggio m lo cifra usando la sua chiave privata. Da qui A ottiene il testo cifrato $c = m^{dA} \bmod nA$ che spedirà a B. Quando B riceverà c sarà sicuro che quest'ultimo è stato spedito da A in quanto solo la sua chiave pubblica (eA, nA) può decrittarlo, e dunque solo A può averlo generato. Inoltre A non può negare di aver generato c in quanto solo lui dovrebbe conoscere la sua chiave privata (non ripudiabilità). Questo meccanismo permette dunque di firmare digitalmente un messaggio garantendo che chi lo riceverà potrà verificarne l'autenticità. Nel seguito è illustrato uno scenario in cui si utilizza la tecnica della firma digitale.



Si noti che la dimostrazione della correttezza di RSA è valida anche nel caso sia utilizzato come strumento per la firma digitale. In questo caso il messaggio viene prima cifrato con la chiave privata e poi decrittato con la chiave pubblica. Ciò equivale a dire che $(m^d)^e \equiv m \bmod pq$.

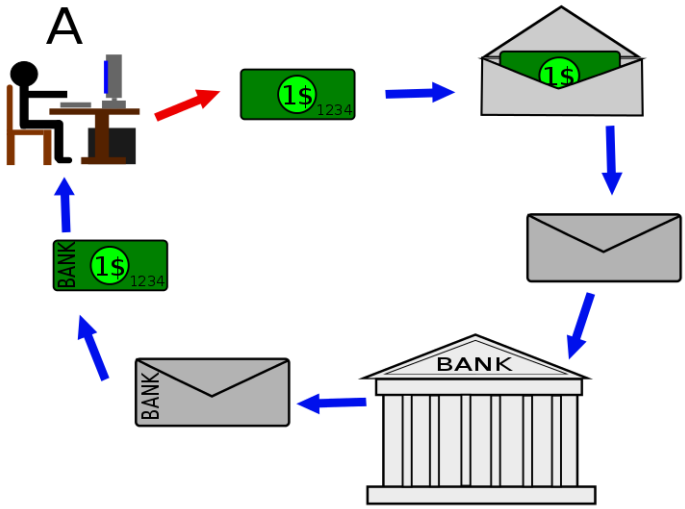
Combinando i due meccanismi si può garantire sia autenticità sia riservatezza.



Nell'ultimo esempio A firma m con la sua chiave privata (dA, nA) ottenendo $mA \equiv m^{dA} \bmod nA$ ed in seguito lo cifra con la chiave pubblica (eB, nB) di B , cioè $c \equiv mA^{eB} \bmod nB$. C a questo punto non può ne leggere ne modificare il messaggio. Quando B riceve il messaggio lo decritta con la sua chiave privata (dB, nB) riottenendo mA, in quanto $c^{dB} \equiv (mA^{eB})^{dB} \equiv mA \bmod nB$ ed infine estrae m e ne verifica l'autenticità con la chiave pubblica (eA, nA) di A tramite l'operazione $mA^{eA} \equiv (m^{dA})^{eA} \equiv m \bmod nA$

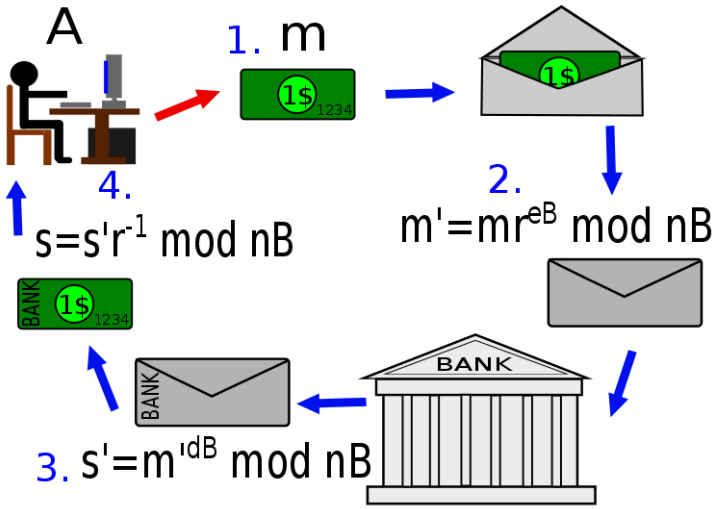
3.5 Blinded signatures in RSA

L'innovazione che E-cash ha portato nel mondo dei pagamenti elettronici, è senz'altro il fatto che le transazioni non sono tracciabili, nel senso che è garantita l'anonimità per chi spende le monete elettroniche. Ma come è possibile che una banca possa certificare le monete rilasciate ad un certo utente e al contempo garantire che da queste non sia possibile risalire a chi le ha spese? La risposta che Chaum ha proposto ed implementato nel protocollo E-cash utilizza la cosiddetta firma cieca (blinded signature). Si consideri il seguente scenario in cui l'utente A vuole acquistare da una certa banca B una moneta elettronica rappresentata dal messaggio m. La moneta è costituita da numerosi campi che ne indicano alcune proprietà come ad esempio l'id della banca che la rilascia, le informazioni sulla moneta, l'expiration date, il numero seriale ecc... Per semplicità in questo contesto, m conterrà solamente il numero seriale cioè il numero che identifica in modo univoco una banconota elettronica. Proprio su questo numero difatti si agisce per raggiungere lo scopo fissato in quanto ciò che si vuole è che la banca non possa associare il numero identificativo della moneta con il cliente che la acquista.
Un modello intuitivo è rappresentato nello schema successivo.



Ciò che si osserva è che A manda la moneta a B (dove B ora è la banca) dentro una busta, e B la certifica (la firma) senza aprire la busta, ovvero senza conoscerne il contenuto. Più in dettaglio ciò che avviene è che A genera m e prima di spedirlo a B lo moltiplica per un fattore r, detto di cecità, che rappresenta la busta. B ricevuto il messaggio siffatto lo firma cifrandolo con la sua chiave privata, e lo rispedisce ad A. A questo punto A può eliminare il fattore r per ottenere il seriale certificato. In termini matematici questo discorso si traduce nei seguenti steps:

- A genera un numero casuale r compreso tra 1 e n che sia coprimo rispetto ad nB.
- A aggiunge ad m il fattore di cecità tramite il passaggio $m' \equiv mr^{eB} \mod nB$. Poiché r, e di conseguenza anche r^{eB}, è random, m' non porta alcuna informazione riguardo m.
- Nel momento in cui B riceve m' lo firma con la sua chiave privata ottenendo $s' \equiv m'^{dB} \mod nB$. Come detto in questo passaggio B non può conoscere m. B infine manda s' ad A.
- Ricevuto s', A può rimuovere il fattore cecità tramite il passaggio $s \equiv s'r^{-1} \equiv m^{dB} \mod nB$ per ottenere la moneta firmata dalla banca B.



Pochi passaggi dimostrano la correttezza del procedimento:
$s \equiv s'r^{-1} \equiv m'^{dB}r^{-1} \equiv (mr^{eB})^{dB}r^{-1} \equiv m^{dB}r^{eBdB}r^{-1} \equiv m^{dB}rr^{-1} \equiv m^{dB} \bmod nB$.
Dove si è usata la sostituzione $r^{eBdB} \equiv r \bmod nB$.

Bibliografia

[1] http://cui.unige.ch/~deriazm/apps/ecash/

[2] http://telemat.die.unifi.it/book/Security/ecash.htm

[3] http://en.wikipedia.org/wiki/RSA_(algorithm)

[4] http://en.wikipedia.org/wiki/Blind_signature