Introduzione alla crittografia moderna

Gabriele D'Angelo (gda@cs.unibo.it)
Ludovico Gardenghi (garden@cs.unibo.it)

Master in Tecnologie del Software Libero e Open Source

Questa presentazione è in parte tratta da un lavoro di

Enrico Zimuel (www.zimuel.it)

Scaletta (1/2)

- Cenni storici minimali
- Ambiti di utilizzo della crittografia ed esempi d’uso
- Crittografia simmetrica
- Il principio di Kerckhoffs
- Alcuni famosi algoritmi crittografici
- Crittografia a chiave pubblica
- Firma digitale
- Sistemi crittografici ibridi (PGP, GPG)
- Attacchi basati su forza bruta

Scaletta (2/2)

- Frontiere della crittografia
- Zero-Knowledge Proof
- Crittografia probabilistica
- Crittografia e anonimato (Tor, Freenet)
- Steganografia
- Conclusioni
- Bibliografia

La crittografia moderna

- Le basi teoriche della crittografia moderna risalgono a circa 30 anni fa a partire dal 1969 con le prime ricerche di **James Ellis** del quartier generale governativo delle comunicazioni britanniche (GCHQ)
- Sviluppata ed affinata nel 1976 in America grazie al contributo di **Whitfield Diffie** e **Martin Hellman** con la nascita del termine crittografia a *chiave pubblica*
- Nasce nel 1977 il cifrario a chiave pubblica **RSA** da tre ricercatori del MIT (Massachusetts Institute of Technology), **Ronald Rivest, Adi Shamir e Leonard Adelman**
- Nel 1991 viene rilasciata la prima versione del software PGP (Pretty Good Privacy) di **Phil Zimmermann**, la crittografia diventa una realtà quotidiana

Foto di gruppo del 1977, da sinistra:
Adi Shamir, Ronald Rivest e Leonard Adelman

Gli ambiti di utilizzo della crittografia

- La crittografia viene utilizzata principalmente per implementare le seguenti operazioni: *autenticazione*, *riservatezza*, *integrità*, *anonimato*
- L'***autenticazione*** è l'operazione che consente di assicurare l'identità di un utente in un processo di comunicazione
- La ***riservatezza*** è l'operazione più nota della crittografia perché è quella che storicamente è nata per prima e che consiste nel proteggere le informazioni da occhi indiscreti
- L'***integrità*** è l'operazione che consente di certificare l'originalità di un messaggio o di un documento. In pratica si certifica che il messaggio non è stato modificato in nessun modo
- L'***anonimato*** è l'operazione che consente di non rendere rintracciabile una comunicazione, è una delle operazioni più complesse da realizzare

Alcuni esempi d'utilizzo della crittografia

- Nei **bancomat** come sistema di protezione delle comunicazioni tra POS (Point Of Sale, punto di vendita) e banca
- Nella telefonia mobile, ad esempio nel protocollo **GSM** tramite l'algoritmo A5/{1,2} o nel protocollo **UMTS**, per la protezione delle comunicazioni vocali
- Nelle comunicazioni satellitari per l'autenticazione e la protezione delle trasmissioni dati satellitari, ad esempio con lo standard **SECA2** impiegato dalla maggior parte delle Tv Digitali
- Su Internet per la protezione del commercio elettronico e delle comunicazioni riservate (protocollo **SSL**)
- Nelle applicazioni di firma dei documenti digitali (**firma digitale**)

* l'algoritmo A5/2 è la versione debole di A5/1 destinata alle nazioni non “affidabili”, inizialmente tenuti segreti. Sono state riscontrate molte debolezze in entrambi, A5/2 è banalmente vulnerabile

Le operazioni di cifratura e decifrazione

- Definiamo con **Msg** "l'insieme di tutti i messaggi" e con **Critto** "l'insieme di tutti i crittogrammi"
- ***Cifratura***: operazione con cui si trasforma un generico messaggio in chiaro **m** in un crittogramma **c** applicando una funzione **C**: Msg->Critto
- ***Decifrazione***: operazione che permette di ricavare il messaggio in chiaro **m** a partire dal crittogramma **c** applicando una funzione **D**: Critto -> Msg
- Matematicamente D(C(m))=m le funzioni C e D sono una inversa dell'altra e la funzione C deve essere iniettiva, ossia a messaggi diversi devono corrispondere crittogrammi diversi

Che cos'è un cifrario?

- Un cifrario è un sistema, di qualsiasi tipo, in grado di trasformare un testo in chiaro (messaggio) in un testo non intellegibile (testo cifrato o crittogramma)

- Nella pratica, per poter utilizzare un cifrario è necessario definire due operazioni: la cifratura del messaggio e la decifrazione del crittogramma

Un primo esempio di cifrario: il cifrario di Cesare

- Consideriamo l'alfabeto italiano, costruiamo un cifrario che sostituisce ad ogni lettera di questo alfabeto la lettera che si trova 3 posizioni in avanti

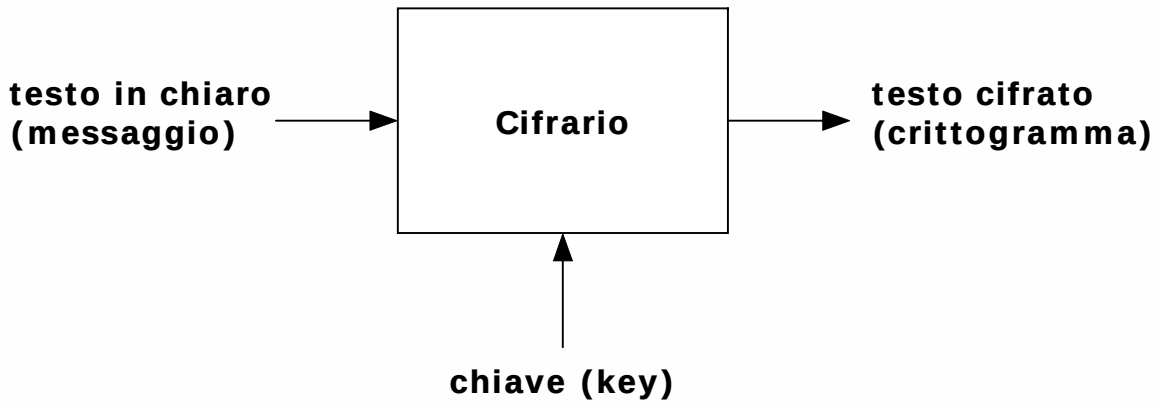
- Ad esempio il testo in chiaro “prova di trasmissione” viene cifrato nel crittogramma “surbd gn zudvpnvvnrqh”

Crittoanalisi del cifrario di Cesare

- Il cifrario di Cesare, come la maggior parte dei cifrari storici basati tu trasposizioni e traslazioni, può essere facilmente violato utilizzando tecniche statistiche (crittoanalisi statistica)
- Si analizzano le frequenze relative dei caratteri nel testo cifrato e le si confrontano con quelle di una lingua conosciuta, ad esempio l'italiano
- Le frequenze relative al testo cifrato "surbd gn zudvpnvvnrqh" risultano *s* (1/19), *u* (2/19), *r* (2/19), *b* (1/19), *d* (2/19), *g* (2/19), *n* (3/19), *z* (1/19), *v* (3/19), *p* (1/19), *h* (1/19).
- Si confrontano tali frequenze con quelle della lingua italiana: *a* (0,114), *e* (0,111), *i* (0,104), *o* (0,099), *t* (0,068), *r* (0,065),...
- Con queste informazioni ottengo una prima approssimazione del testo in chiaro "s**ro**b**a** g**i** z**ra**vp**i**vv**io**qh", procedo per tentativi ripetendo il procedimento

La crittografia simmetrica

- Introduciamo un parametro chiamato **k** (key= chiave) all'interno delle funzioni di cifratura **C(m,k)** e decifrazione **D(c,k)**
- Si parla di crittografia simmetrica perché si utilizza la stessa chiave k per le operazioni di cifratura e decifrazione
- La robustezza del cifrario dipende, a differenza di prima, solo dalla segretezza della chiave **k**



La macchina Enigma

- Macchina crittografica elettro-meccanica, **portatile,** utilizzata durante la seconda guerra mondiale per cifrare e decifrare messaggi
- Basata su un insieme modificabile di rotori, posizionabili secondo diverse configurazioni
- Ad ogni messaggio si sarebbe dovuto alterare la configurazione iniziale
- ULTRA: progetto per la decifrazione delle comunicazioni tedesche da parte dell'intelligence britannica
- L'attacco ad Enigma ha riguardato diversi approcci:
  - Parziale forza bruta (bombe-machine, colossus)
  - Analisi delle debolezze
  - Cattura di una delle macchine e di cifrari
  - Errori pratici nell'utilizzo di Enigma

Il principio di Kerckhoffs

- Risulterà strano ma uno dei principi fondamentali della crittografia, utilizzato ancora nei moderni sistemi crittografici è stato individuato nel lontano 1883 dal linguista franco-olandese August Kerckhoffs nel suo celebre articolo "La cryptographie militaire" apparso nel Journal des sciences militaires
- Principio di Kerckhoffs: *"La sicurezza di un sistema crittografico è basata* ***esclusivamente*** *sulla conoscenza della chiave, in pratica si presuppone noto a priori l'algoritmo di cifratura e decifrazione"*
- Purtroppo alcuni sistemi crittografici proprietari moderni **non** rispettano questo essenziale principio di sicurezza

Piccoli incidenti di percorso

- **NSFOCUS Security Advisory (SA2000-05)**
  - **Topic:** Microsoft Windows 9x NETBIOS password verification vulnerability
  - http://www.nsfocus.com/english/homepage/research/0005.htm

- **Schneier on Security**
  - **Topic: Microsoft RC4 Flaw**
  - http://www.schneier.com/blog/archives/2005/01/microsoft_rc4_f.html

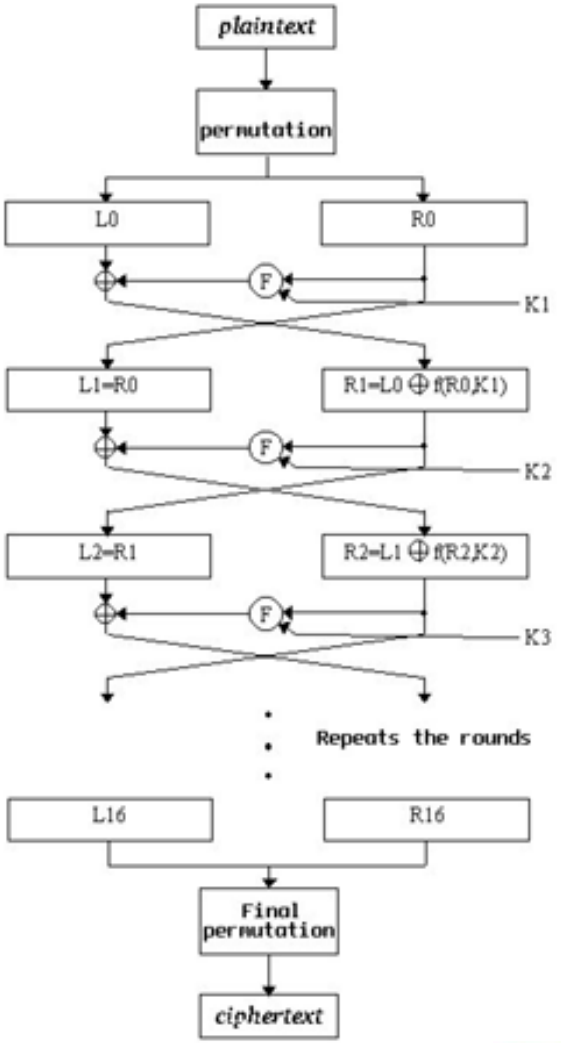
Il problema della trasmissione della chiave

- Volendo utilizzare un cifrario simmetrico per proteggere le informazioni tra due interlocutori come posso scambiare la chiave segreta? Devo utilizzare una "**canale sicuro**" di comunicazione

- **Il "canale sicuro" esiste nella realtà? E quanto è costoso?**
- Per una comunicazione sicura tra n utenti si dovranno scambiare in tutto (n-1)*n/2 chiavi, ad esempio con 100 utenti occorreranno 4950 chiavi, il tutto per ogni comunicazione!

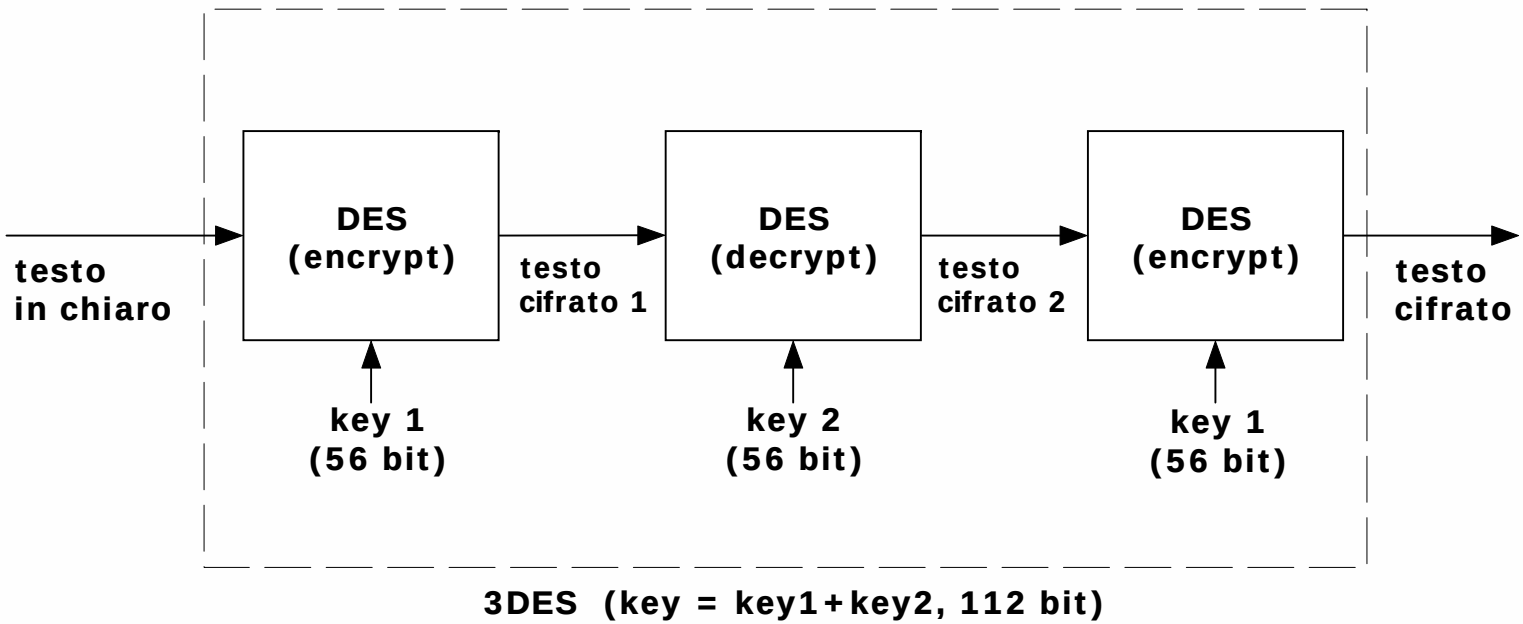
DES (Data Encryption Standard)

- Sviluppato dall'IBM nel 1970 diventato standard nel 1976
- Utilizza chiavi di 56 bit, divide il testo in chiaro in blocchi di 64 bit, effettua delle permutazioni iniziali e finali ed un ciclo di 16 iterazioni di permutazioni e xor (Feistel network, tecniche di confusione e diffusione)
- Il 17 Luglio 1998, l'EFF (Electronic Frontier Foundation) costruisce un sistema dedicato in grado di violare il DES in meno di 3 giorni, tramite un attacco di tipo "brute-force"
- Morale della favola: non è una buona idea utilizzare sistemi di cifratura basati sul DES!
- Anche i protocolli crittografici "scadono" e con i protocolli "scadono" anche tutti i messaggi crittografati che vengono conservati!



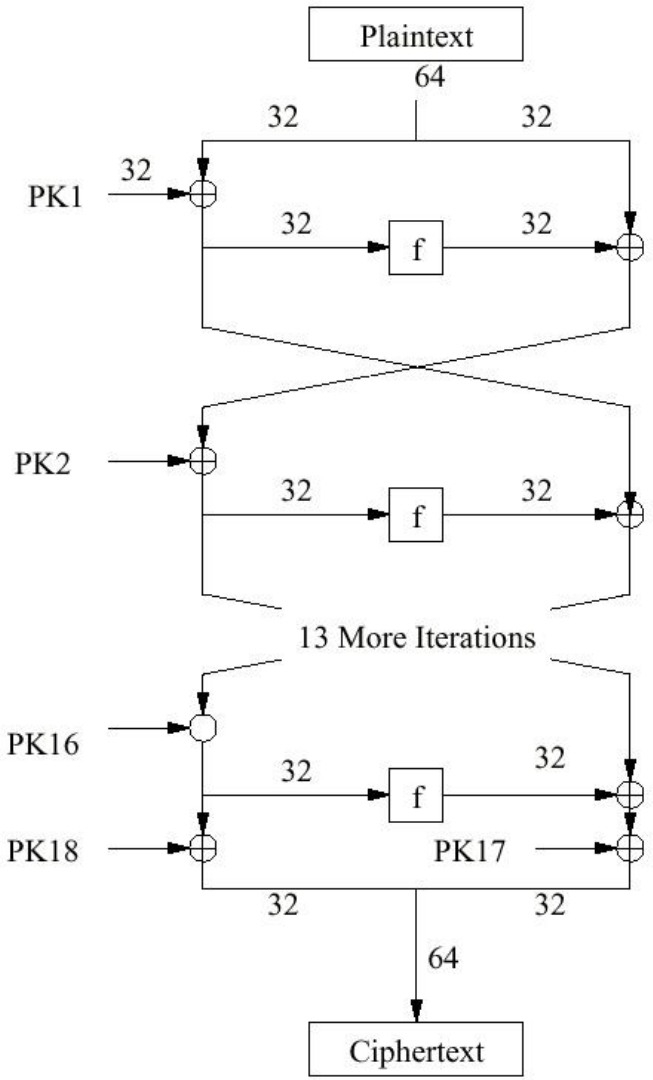
3DES

- Evoluzione del DES, è basato su un utilizzo del cifrario DES ripetuto, chiavi di 112 bit
- Esiste in varie versioni. Ad esempio si utilizza la tecnica della codifica-decodifica-codifica (EDE, Encrypt-Decrypt-Encrypt) utilizzando il cifrario DES



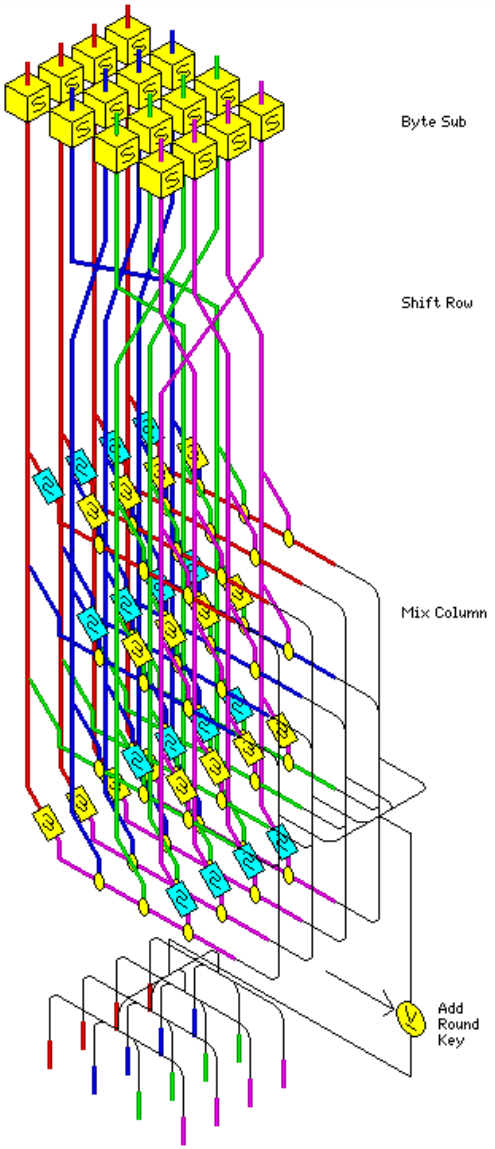
Blowfish

- Ideato nel 1993 da Bruce Schneier
- E' stato sviluppato come algoritmo di encryption: veloce, compatto, semplice da implementare e sicuro con chiavi di dimensioni variabili fino a 448 bit
- E' un cifrario a blocchi di 64 bit, basato sulle reti di Feistel
- Non si conoscono attacchi efficaci
- E' un algoritmo non patentato, utilizzato in molti sistemi open source (come ad esempio in OpenBSD)



Rijndael (AES)

- Sviluppato Joan Daemen e Vincent Rijmen
- Questo algoritmo ha vinto la selezione per l'Advanced Encryption Standard (**AES**) il 2 Ottobre 2000. Ufficialmente il Rijndael è diventato lo standard per la cifratura del XXI secolo
- Il cifrario utilizza chiavi di lunghezza variabile 128, 192, 256 bit (gli autori hanno dimostrato come è possibile variare le dimensioni delle chiavi con multipli di 32 bit). Lo schema del Rijndael è stato influenzato dall'algoritmo SQUARE

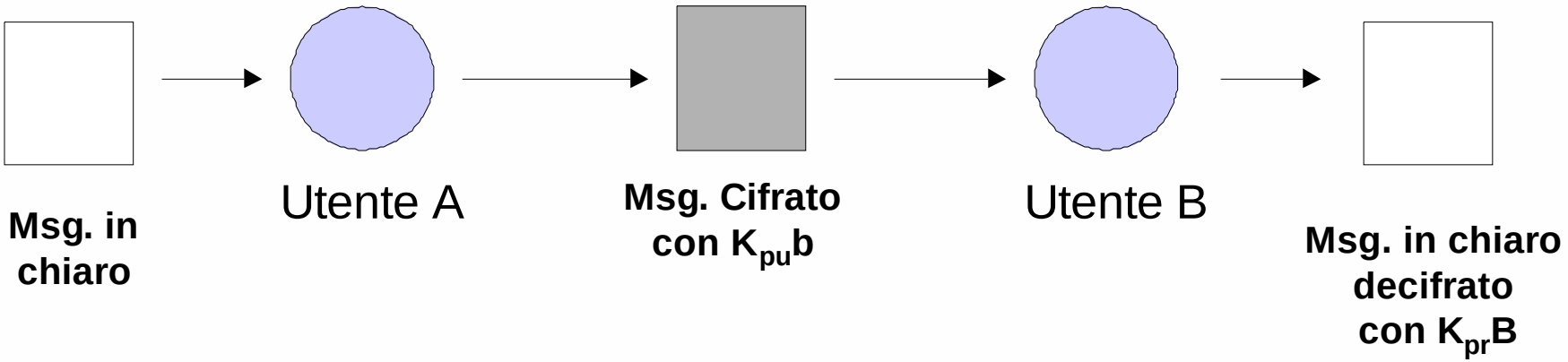


La crittografia a chiave pubblica

- Utilizza una coppia di chiavi per le operazioni di cifratura (*encryption*) e decifrazione (*decryption*)
- Una chiave detta pubblica (**public key**) viene utilizzata per le operazioni di encryption
- L'altra chiave, detta privata (**private key**), viene utilizzata per le operazioni di decryption
- A differenza dei cifrari simmetrici **non** è più presente il tradizionale problema della **trasmissione sicura delle chiavi**. Ma è presente un nuovo problema: la **diffusione della chiave pubblica**.
- Sono normalmente considerati come intrinsecamente sicuri poiché utilizzano tecniche di tipo matematico basate sulla teoria dei numeri, sulla teoria delle curve ellittiche, etc. (Salvo che non si dimostri che questi problemi sono "*facili*" o gli ovvi ed estremamente comuni errori di implementazione/gestione/utilizzo)

La crittografia a chiave pubblica

- Esempio di encryption (trasmissione sicura):

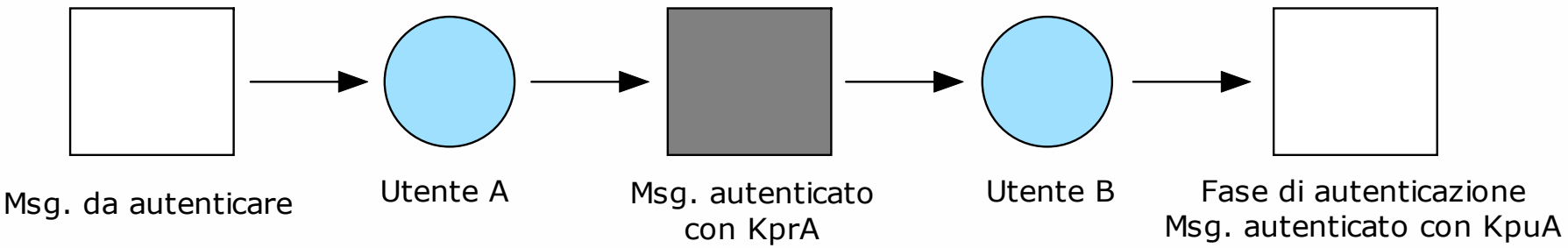


$K_{pu}B$ = chiave pubblica dell'utente B

$K_{pr}B$ = chiave privata dell'utente B

La crittografia a chiave pubblica

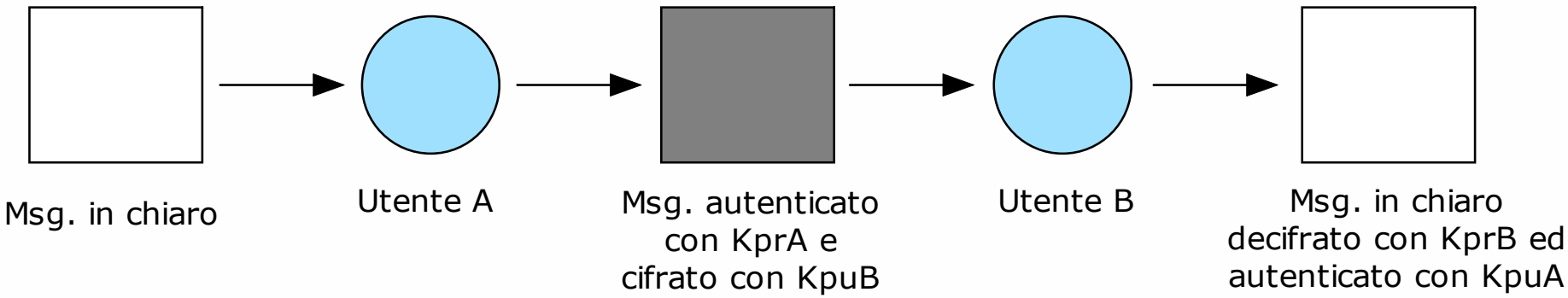
- Esempio di autenticazione:



KprA = chiave privata dell'utente A
KpuA = chiave pubblica dell'utente A

La crittografia a chiave pubblica

- Esempio di encryption ed autenticazione:



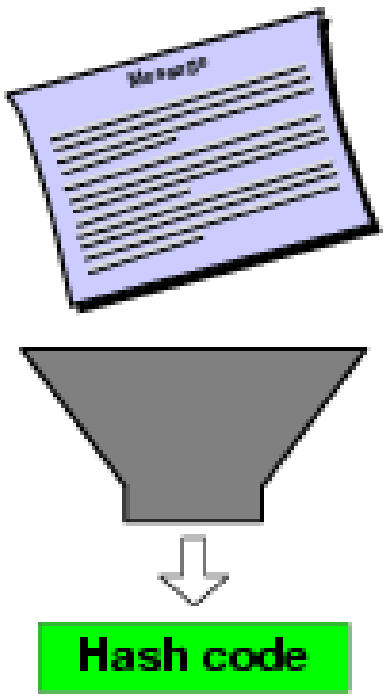
KprA = chiave privata dell'utente A
KpuA = chiave pubblica dell'utente A
KprB = chiave privata dell'utente B
KpuB = chiave pubblica dell'utente B

La firma digitale e le funzioni hash sicure

- Nasce come applicazione dei sistemi a chiave pubblica
- Viene utilizzata per autenticare la paternità di un documento informatico e la sua integrità
- Si utilizza un cifrario a chiave pubblica e si “cifra” un documento (file) con la propria chiave segreta. Chiunque può verificare la paternità del documento utilizzando la chiave pubblica dell'utente che ha firmato il documento
- Problema: per l'autenticazione di un documento di grandi dimensioni con un algoritmo a chiave pubblica occorre molto tempo
- Soluzione: posso autenticare solo un “riassunto” del documento tramite l'utilizzo di una funzione hash sicura

Le funzioni hash sicure

- Vengono utilizzate per generare un sorta di "riassunto" di un documento informatico (file)
- Una funzione hash accetta in ingresso un messaggio di lunghezza variabile M e produce in uscita un digest di messaggio H(M) di lunghezza fissa
- Questo digest (impronta digitale, targa, riassunto) è strettamente legato al messaggio M, ogni messaggio M genera un H(M) univoco
- Anche considerando due messaggi M ed M' differenti solo per un carattere le loro funzioni hash H(M) e H(M') saranno diverse



Requisiti di una funzione hash sicura H(x):

- H può essere applicata a un blocco di dati di qualsiasi dimensione;
- H produce in uscita un risultato di lunghezza fissa (ad esempio 160 bit);
- Per qualunque codice h il calcolo di x tale che H(x)=h deve avere una complessità computazionale improponibile;
- Per qualunque blocco di dati x il calcolo di y≠x tale che H(x)=H(y) deve avere una complessità computazionale improponibile
- Ai fini pratici H(x) deve essere relativamente semplice da calcolare

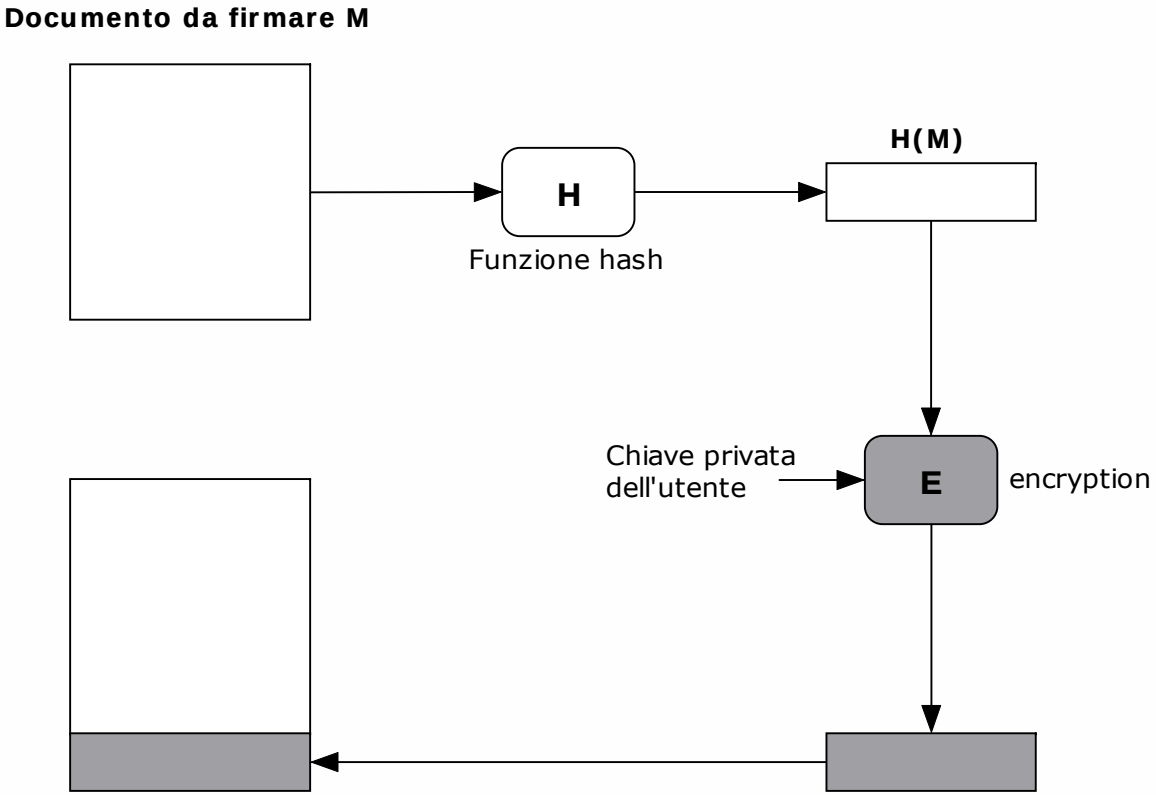
Esempio di funzione hash:

- Tutte le funzioni hash operano sulla base del seguente principio: i dati in ingresso sono considerati come una sequenza di blocchi di n bit, essi vengono elaborati un blocco alla volta iterativamente per produrre una funzione hash di n bit
- Una delle più semplici funzioni hash è quella che esegue lo XOR bit a bit di ciascun blocco, ossia:

$$C_i = b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{im}$$

- Dove C_i rappresenta l'i-esimo bit del codice hash, m il numero di blocchi di n bit, b_{ij} l'i-esimo bit all'interno del j-esimo blocco e l'operatore $\oplus$ l'operazione di XOR
- La probabilità che un errore nei dati produca lo stesso valore hash è 2^{-n}, con n=128 bit abbiamo $2^{-128} \approx 2{,}9387 * 10^{-39}$

Esempio di firma digitale di un documento:

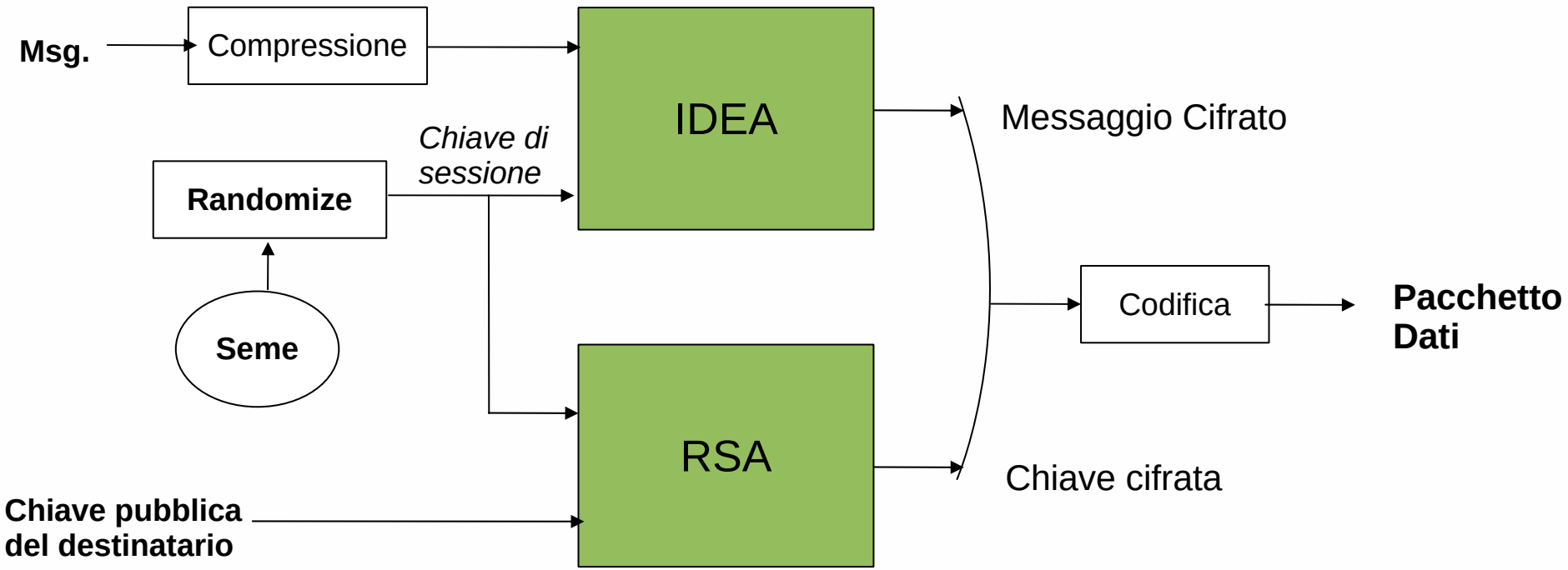


Documento firmato:
Il ricevente può verificare
la firma utilizzando la
chiave pubblica dell'utente firmatario
e riapplicando la funzione hash

Esempio di un sistema crittografico ibrido: PGP

- **PGP (Pretty Good Privacy)** è un software creato da Phil Zimmermann nel 1991
- E' un software per la privacy personale: protezione delle email, dei file, firma digitale
- Utilizza gli algoritmi di crittografia a chiave pubblica RSA, Diffie-Hellman, DSA e gli algoritmi simmetrici IDEA, CAST, 3-DES
- E' basato su di un sistema di crittografia "ibrido" nel senso che utilizza crittografia simmetrica per le operazioni di encryption sui dati generando delle chiavi di sessione pseudo-casuali cifrate con un algoritmo a chiave pubblica
- Attualmente PGP viene distribuito dalla PGP Corporation come software commerciale

Il funzionamento del PGP: esempio di cifratura



GPG: The GNU Privacy Guard

- GPG è l'implementazione free dello standard OpenPGP (RFC 2440)
- La versione 1.0.0 è stata rilasciata nel settembre 1999. Ora siamo rispettivamente alla versione 1.4.7 e 2.0.6
- Rappresenta un ottimo sostituto di PGP e mantiene compatibilità con varie versioni (5, 6, 7)
- Non usa alcun algoritmo brevettato
- Supporta: ElGamal, DSA, RSA, AES, 3DES, Blowfish, Twofish, CAST5, MD5, SHA-1, RIPE-MD-160 and TIGER
- **Supporto integrato per i keyserver**
- Disponibile per molti flavor di Unix/Linux, OS X, Microsoft Windows ecc.

Perchè la crittografia è in grado di garantire la sicurezza?

- Perché è basata sull'impossibilità di risolvere, allo stato attuale, dei problemi matematici in tempi “ragionevoli”
- In altre parole la sicurezza della crittografia è basata sulla difficoltà di risoluzione di alcuni problemi, ad esempio come il problema della fattorizzazione di grandi numeri
- Siano p e q due numeri primi, scelti a caso, di dimensioni elevate (dell'ordine di 10^{20}) e sia $n=pq$ il prodotto di questi primi. Conoscendo solo n è molto difficile scomporlo nei suoi fattori primi, ossia calcolare p e q
- Non esiste, allo stato attuale, un algoritmo di risoluzione del problema della fattorizzazione in tempi “ragionevoli” (al più polinomiali). E se esistesse l’impatto sull’informatica moderna sarebbe devastante
- La sicurezza del cifrario **RSA** è basata proprio su questo assunto

Gare di fattorizzazione su Internet

- La società RSA Security offriva 20.000 $ a chiunque riuscisse a fattorizzare il seguente numero di 193 cifre decimali:

  n=3107418240490043721350750035888567930037346022842727545720161948823206440518081504556346829671723286782437916272838033415471073108501919548529007337724822783525742386454014691736602477652346609

- p=?, q=?

- Ma non è finita…
  - http://www.rsasecurity.com/rsalabs/node.asp?id=2093

Attacchi massivi alla crittografia: distributed.net

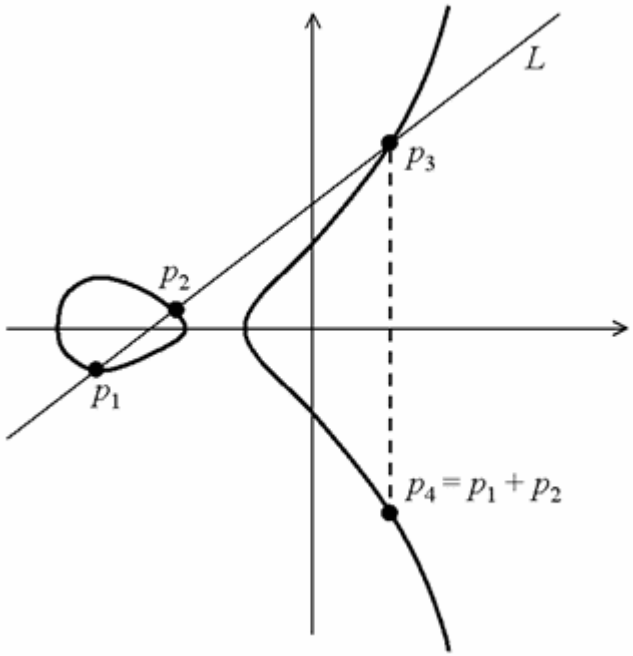
- Gli attacchi brute-force o semi-brute-force richiedono potenze di calcolo impossibili da ottenere su una macchina singola. Qual'è il limite della legge di Moore?

- L'alternativa è quella di aggregare assieme quante più risorse possibili, attraverso un numero massivo di client collegati tra di loro attraverso una LAN, WAN o eventualmente Internet

- I risultati sono sorprendenti ed utili sia per quanto riguarda la crittografia sia per lo studio di alcuni tipi di sistemi distribuiti (seti@home ecc.)

Le frontiere matematiche della crittografia

- Fondamentalmente la base della crittografia è la teoria dei numeri e la costruzione di funzioni "difficilmente" invertibili
- La tendenza attuale è quella di utilizzare insiemi costruiti su strutture algebriche particolari come le curve ellittiche
- Una curva ellittica è l'insieme dei punti (x,y), su di un piano cartesiano, che soddisfano le condizioni di un'equazione cubica, ad esempio $y^2=x^3+ax+b$, con l'aggiunta di un elemento denotato *O* e chiamato *punto infinito*



La rivoluzione della crittografia quantistica

- Mentre nella crittografia classica si utilizzano tecniche matematiche per garantire la privacy delle comunicazioni, nella crittografia quantistica sono le leggi della fisica a proteggere l'informazione
- La crittografia quantistica è basata sulle leggi della meccanica quantistica ossia lo studio della fisica a livello microscopico delle particelle elementari della materia
- Una delle leggi fondamentali della meccanica quantistica, il principio di **indeterminazione di Heisenberg**, ci dice che ogni misura effettuata su un sistema quantistico perturba il sistema stesso. Un altro principio utilizzato è quello dell'**entanglement quantistico**
- La crittografia quantistica sfrutta questa proprietà per garantire una comunicazione sicura. Nessuno è in grado di intercettare un messaggio senza modificarne il contenuto!

La rivoluzione della crittografia quantistica

- La crittografia quantistica si utilizza convenzionalmente per scambiare la chiave di cifratura di due interlocutori e non il messaggio vero e proprio
- Successivamente con la chiave di cifratura ed un algoritmo di tipo simmetrico è possibile cifrare le comunicazioni
- Lo scambio dell'intero messaggio su un canale quantistico non protegge in sé l'informazione ma consente solo di stabilire se non ci sono intrusi in ascolto
- Per questo è conveniente generare a caso una chiave di cifratura inviarla su di un canale di comunicazione quantistico e determinare se è stata o meno intercettata. Nel caso in cui la chiave è stata intercettata si ripete l'operazione con una nuova chiave di cifratura fino a quando la comunicazione non risulterà sicura

Zero-Knowledge Proof

- Metodo interattivo attraverso il quale una parte prova all'altra che un'affermazione è vera, senza divulgare nulla più che la veridicità dell'affermazione
- Ad esempio: conosco la password e lo dimostro, ma senza mai divulgare qual'è la password. E quindi proteggendomi da un eventuale sniffer e in parte da un server compromesso

Crittografia probabilistica

- Utilizzo della casualità nei meccanismi crittografici
- Particolarmente utile nel caso di crittografia a chiave pubblica: messaggio con ugual contenuto spedito a più destinatari
- Un'implementazione banale: aggiungere dell'informazione casuale durante la crittografia, poi scartata in decrifratura

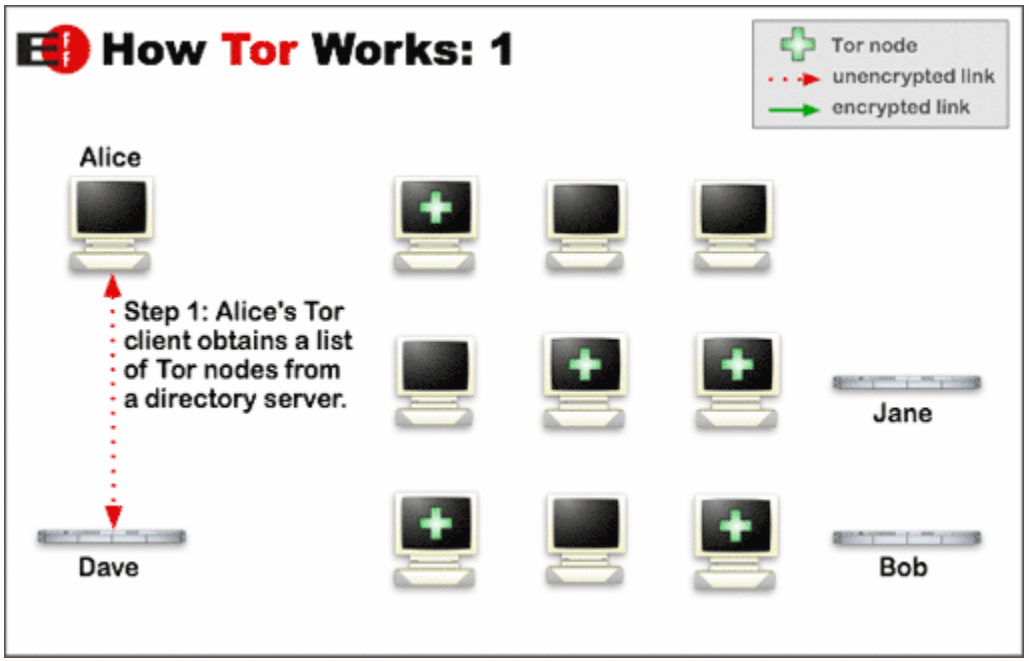
Crittografia e anonimato

- Una serie di applicazioni della crittografia ha lo scopo di garantire non solo la *riservatezza* delle comunicazioni ma anche di rendere difficile a terze persone **risalire al *mittente*, al *destinatario* o a entrambi** gli estremi coinvolti in uno scambio di informazioni
- I motivi possono essere vari e più o meno "nobili": comunicazioni in paesi con leggi restrittive della libertà di espressione; sfiducia nei gestori di reti e servizi; scambio illegittimo di dati; attacchi verso host remoti
- All'anonimato concorrono sia la struttura della rete e del protocollo sia l'uso di crittografia
- Esistono strumenti che offrono diversi livelli di anonimato; ciascuno di essi raggiunge un **compromesso** fra sicurezza e prestazioni, semplicità d'uso, affidabilità

Tor: un'applicazione dell'*onion routing*

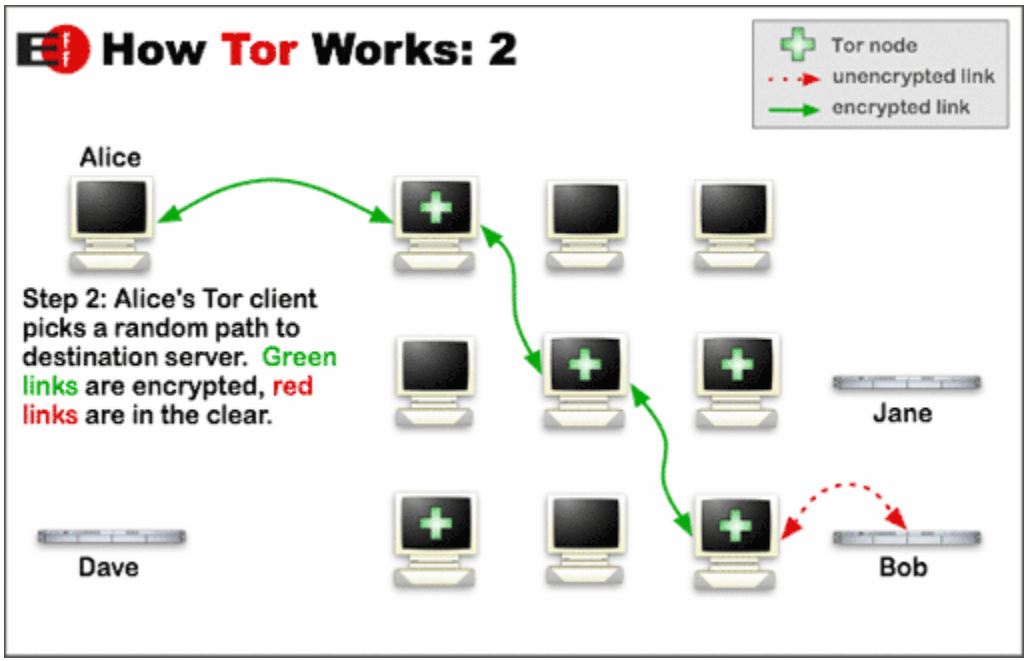
- Tor vuole fornire 3 tipi di protezione: nascondere la posizione del client; impedire analisi locale del traffico (ad esempio da parte del provider); offrire sicurezza anche se uno o più nodi della rete di overlay vengono “compromessi”
- Si basa sul concetto di tunnel virtuale. I tunnel sono a tempo e vengono instradati attraverso un insieme di nodi intermedi
- I pacchetti vengono cifrati “a cipolla”: ciascun nodo rimuove lo strato di crittografia più esterno e ottiene solamente le informazioni necessarie a inoltrare il pacchetto al nodo successivo
- Solitamente l'identità del server non viene nascosta, anche se è possibile farlo
- Le garanzie di anonimato sono buone ma inadeguate per situazioni “estreme”

Funzionamento del protocollo Tor



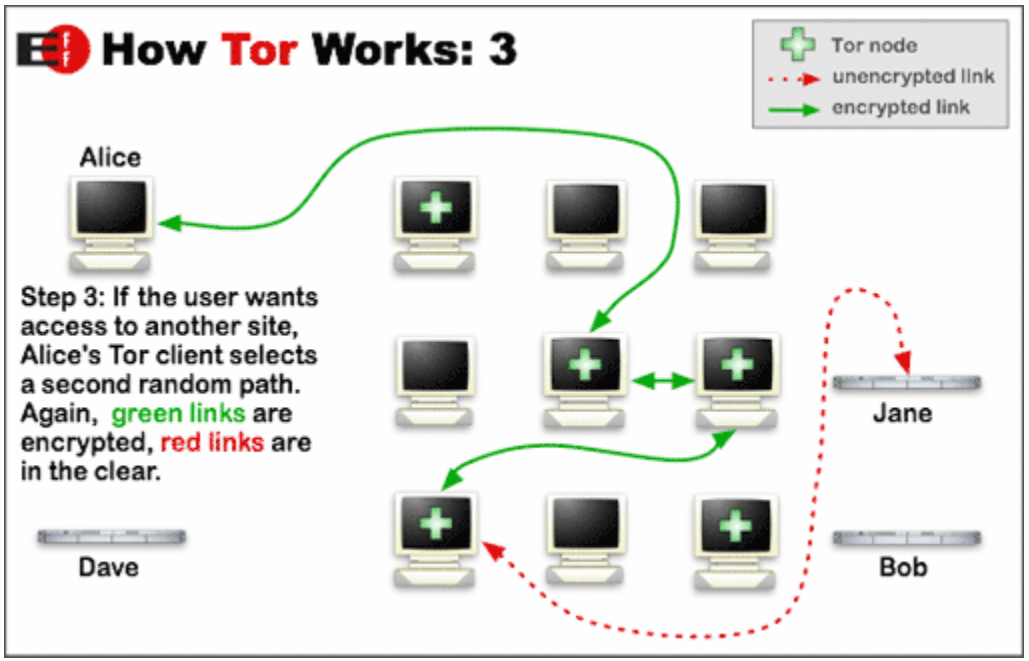
- Dopo aver interrogato un directory server viene costruito un tunnel virtuale che verrà mantenuto non più di 10 minuti. Ogni "hop" del tunnel utilizza una coppia diversa di chiavi crittografiche

Funzionamento del protocollo Tor



- Il tunnel viene costruito scegliendo casualmente dei nodi tra quelli disponibili. Una volta costruito il tunnel possono essere veicolate comunicazioni TCP

Funzionamento del protocollo Tor



- La compromissione di un nodo lungo il tunnel e lo sniffing non alterano sensibilmente la sicurezza del protocollo. Ogni comunicazione con destinatario diverso richiede la costruzione di un nuovo tunnel virtuale

Freenet: una rete contro la censura

- Freenet consente lo scambio di informazioni indicizzate in base a una chiave; non è garantita la persistenza delle informazioni "poco richieste" ma allo stesso tempo è impossibile rimuovere volontariamente (cioè censurare) un dato file dalla rete o modificarlo
- Le informazioni si propagano fra i nodi, ciascuno dei quali mette a disposizione spazio usato come "cache" per i dati di tutta la rete; le richieste vengono propagate fino a un nodo che contiene il dato richiesto
- I file sono conservati in forma cifrata sui nodi; è legalmente (ma non impossibile) complesso accusare chi gestisce un nodo per la responsabilità della detenzione di dati illegali
- Anche le comunicazioni sono cifrate e non viene tenuta traccia del mittente né del destinatario di un pacchetto

Crittografia e anonimato: pro e contro

- Una comunicazione cifrata che garantisce l'anonimato può essere usata da chiunque anche per scopi illeciti (i soliti esempi: pedopornografia, terrorismo, traffici illeciti ecc.)
- Può venire la tentazione di vietare l'uso legale della crittografia per poter controllare questi fenomeni
- L'applicazione pratica di una restrizione di questo tipo è molto complessa: è sempre possibile, per esempio, ricorrere a tecniche di *steganografia* (delle quali si parlerà fra poco)
- Inoltre, non è una conseguenza ovvia il fatto che chi vuole usare crittografia per scopi *illegali* smetta di farlo soltanto in quanto *illegale*

Steganografia

- Non si tratta di una tecnica crittografica ma può essere complementare all'uso della crittografia
- Idea di base: *nascondere* dei dati (di qualunque tipo) all'interno di altri (solitamente dall'aria “innocua”)
- Esempio classico: nascondere un file di testo all'interno di un'immagine, sfruttando la scarsa sensibilità dell'occhio umano per codificare informazione sotto forma di leggere variazioni di colore
- Da sola non è particolarmente efficace (è come un algoritmo crittografico che vìoli il principio di Kerckhoffs)
- È invece utile per nascondere dati cifrati e rendere il traffico “meno sospetto” agli occhi di un osservatore esterno

Alcune conclusioni molto generali

- Applied Cryptography By Bruce Schneier (Second edition)

AFTERWORD By Matt Blaze. "Top Ten Threats to Security in Real Systems":

1. *The sorry state of software*
2. *Ineffective protection against denial-of-service attacks*
3. *No place to store secrets*
4. *Poor random number generation*
5. *Weak passphrases*
6. *Mismatched trust*
7. *Poorly understood protocol and service interactions*
8. *Unrealistic threat and risks assessment*
9. *Interfaces that make security expensive and special*
10. *No broad-based demand for security*

Bibliografia

- *Introduzione alla crittografia*. A. Languasco, A. Zaccagnini. Hoepli Informatica. ISBN 88-023-3392-9

- *Crittografia. Tecniche di protezione dei dati riservati*. A. Sgarro. Muzzio Biblioteca. ISBN 88-7021-640-3

- http://www.philzimmermann.com
- http://www.rsasecurity.com
- http://www.schneier.com

- http://www.gnupg.org/
- http://www.freenetproject.org/
- http://tor.eff.org/

- http://www.wikipedia.org
- http://en.wikipedia.org/wiki/Cryptography
- http://en.wikipedia.org/wiki/Topics_in_cryptography

Ringraziamenti e licenza d'uso

- Parte di queste slide sono "liberamente tratte" da un precedente lavoro di Enrico Zimuel (http://www.zimuel.it), rilasciate secondo licenza CopyFree. *Si ringrazia vivamente!*

- Queste slide sono rilasciate con licenza Creative Commons Attribution-ShareAlike, *fatti salvi i singoli componenti rilasciati sotto diversa licenza*