Corso di Laurea in Ingegneria Informatica

Corso di Reti di Calcolatori I

Antonio Pescapè (pescape@unina.it)

Sicurezza nella comunicazione in rete: tecniche crittografiche

I lucidi presentati al corso sono uno strumento didattico
che NON sostituisce i testi indicati nel programma del corso
I lucidi sono adattati dagli originali di J. Kurose e K. Ross e fanno riferimento al testo
Reti di calcolatori e Internet - Un approccio top-down (6a ed.)

Nota di copyright per le slide COMICS

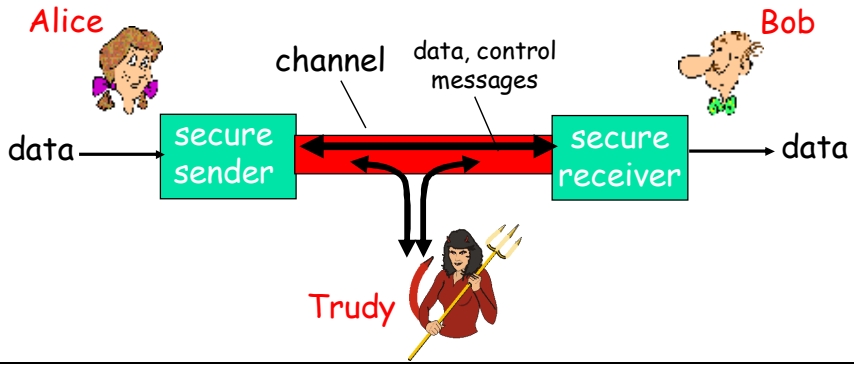
Nota di Copyright

Sicurezza della comunicazione in rete

- La sicurezza della comunicazione in rete coinvolge diversi aspetti
- Riservatezza: solo il mittente ed il destinatario "legittimo" dovrebbero essere in grado di comprendere il contenuto del messaggio
  - Le tecniche crittografiche servono innanzitutto a proteggere la confidenzialità della comunicazione
  - Il mittente effettua un'operazione di cifratura del messaggio (*encryption*) ed il ricevente un'operazione duale di decifratura (*decryption*)
- Integrità dei messaggi: mittente e destinatario di un messaggio desiderano essere certi che i messaggi scambiati non siano alterati da una terza parte senza che se ne possano accorgere
- Autenticazione: mittente e destinatario di un messaggio desiderano essere reciprocamente sicuri dell'identità della controparte
- Accessibilità e disponibilità dei servizi: i servizi offerti in rete devono essere protetti da eventuali attacchi (es. attacchi *Denial of Service*, DoS)

Amici e nemici: Alice, Bob, Trudy

- Alice e Bob sono le due parti che intendono comunicare in maniera sicura attraverso la rete
  - Potrebbero essere programmi client e server, o dispositivi (es. router) …
- Trudy è una terza parte che può ascoltare i messaggi scambiati da Alice e Bob ed eventualmente alterarli, cancellarli o crearne di falsi



Esempi di comportamenti malevoli

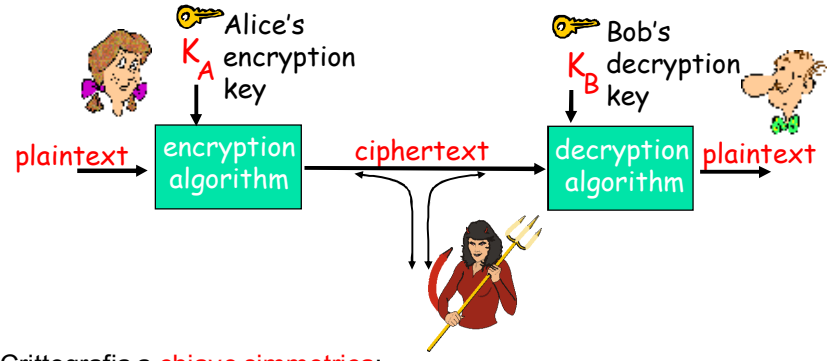
La terza parte Trudy può:

- intercettare i messaggi inviati da Alice a Bob (*eavesdropping*)
- inserire messaggi fasulli nel flusso della comunicazione da Alice a Bob
- inviare pacchetti con il campo source address fasullo (*spoofing*) in modo da fingere di essere Alice
- dirottare la comunicazione tra Alice e Bob, piazzandosi "in mezzo", ad es. fingendo con Alice di essere Bob e con Bob di essere Alice (*hijacking*)
- impedire al servizio offerto da Bob di essere utilizzabile (es. sovraccaricando le risorse di Bob) (*denial of service*)

Crittografia: concetti generali

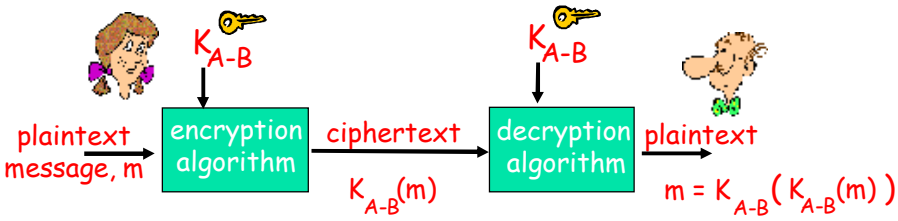
- Un *sistema crittografico* è un sistema in grado di cifrare e decifrare un messaggio attraverso l'uso di un *algoritmo* e di una *chiave* (una stringa alfanumerica)
- Il messaggio da cifrare è detto "testo in chiaro" (*plaintext*) mentre il risultato dell'algoritmo crittografico è detto "testo cifrato" (*ciphertext*)

Crittografia: terminologia



- Crittografia a chiave simmetrica:
  - mittente e destinatario usano la stessa chiave (segreto condiviso) per *encryption* e *decryption* ($K_A = K_B = K_{A-B}$)
- Crittografia a chiave pubblica:
  - la chiave per la *encryption* è *pubblica* (nota a tutti), la chiave per la *decryption* è *segreta* (privata)

Crittografia a chiave simmetrica



Crittografia a chiave simmetrica: Bob ed Alice entrambi conoscono la chiave crittografica (simmetrica) K_{A-B}

- Come fanno Bob ed Alice a mettersi d'accordo sul valore della chiave? Lo scambio deve avvenire in maniera sicura (es. in un incontro di persona…)

Crittografia a chiave simmetrica: un esempio

Cifrario per sostituzione: unità di testo del plaintext sono sostituite con corrispondenti sequenze di simboli nel testo cifrato secondo uno schema regolare

In particolare, **cifrario monoalfabetico**: una corrispondenza fissa tra ciascuna lettera dell'alfabeto in chiaro ed una lettera dell'alfabeto cifrato

```
plaintext:  abcdefghijklmnopqrstuvwxyz
ciphertext: mnbvcxzasdfghjklpoiuytrewq
```

Es.: `plaintext: bob, i love you. alice`

`ciphertext: nkn, s gktc wky. mgsbc`

Q: Quanto è difficile decifrare questo codice ?

- con attacco a forza bruta: $26! \approx 2^{88}$ tentativi
- con critto-analisi, abbastanza facile
  - pattern ricorrenti ed analisi statistica delle frequenze di occorrenza

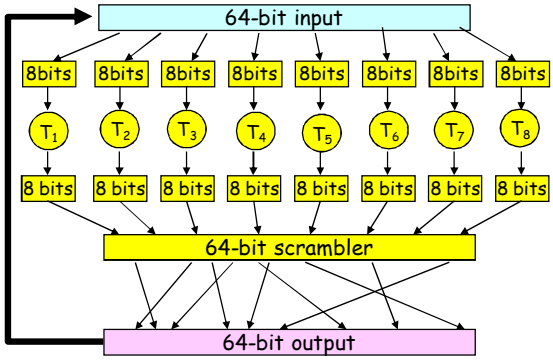
Cifrari a blocchi

- Un blocco di k bit del testo in chiaro è codificato con altri k bit nel testo in codice secondo uno schema fisso
- Es.: k=3

```
000 → 110   001 → 111   010 → 101   011 → 100
100 → 011   101 → 010   110 → 000   111 → 001
```

- Il messaggio (010)(110)(001)(111) è codificato in (101)(000)(111)(001)
- Quante corrispondenze si possono creare con k=3 bit ?
  N. permutazioni di 2^k=8 oggetti (triple di bit) = $2^k! = 8! = 40320$
- Per una maggiore sicurezza occorre aumentare k (es. k=64)
- Per valori di k grandi, difficoltà di implementazione: cifratura e decifratura richiedono una tabella di 2^k elementi in memoria
- Elevati valori di k ottenuti per composizione di valori più piccoli

Cifrari a blocchi

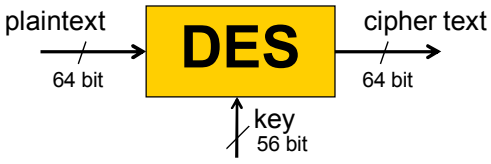


- dopo una iterazione, ogni singolo bit del testo in chiaro influenza otto bit del testo cifrato

- Se la funzione di rimescolamento è fissa, la chiave è costituita dalle 8 tabelle di permutazione
- Esempi di cifrari a blocchi: DES, 3DES, AES

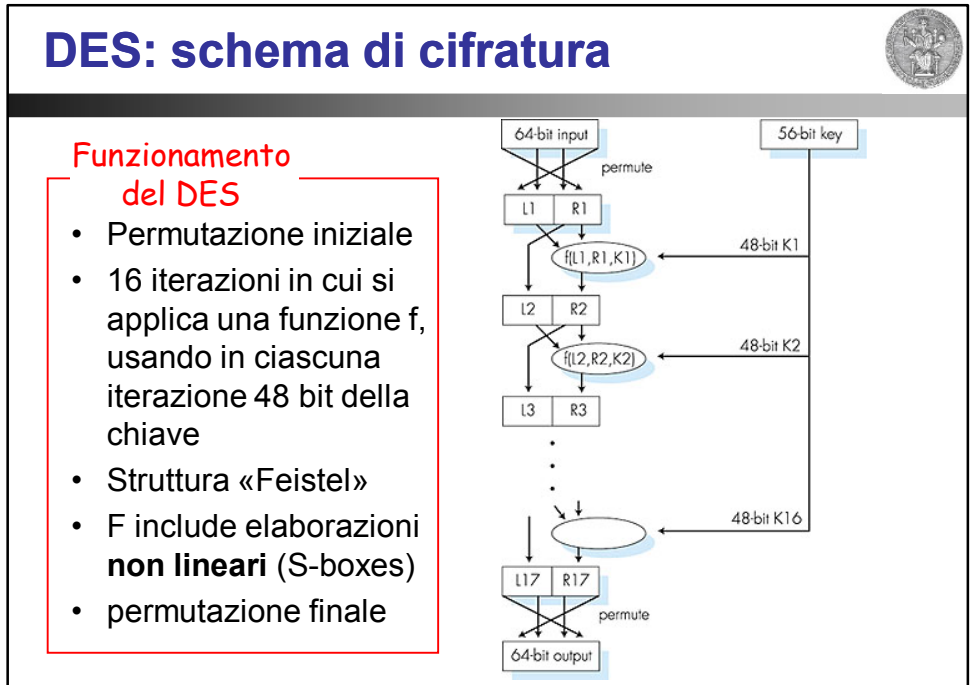
DES: caratteristiche generali

- DES codifica blocchi di 64 bit e usa una chiave di 56 bit
- Chiave da 56 bit + 8 bit di parità



- La chiave è memorizzata su 64 bit, con l'ottavo, il 16-esimo, ... , il 64-esimo bit calcolati come bit di parità per i 7 bit precedenti

DES: schema di cifratura



Funzionamento del DES

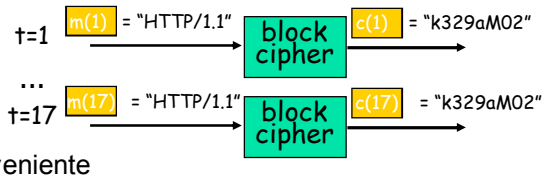
- Permutazione iniziale
- 16 iterazioni in cui si applica una funzione f, usando in ciascuna iterazione 48 bit della chiave
- Struttura «Feistel»
- F include elaborazioni **non lineari** (S-boxes)
- permutazione finale

56 bit non sono abbastanza: 3DES

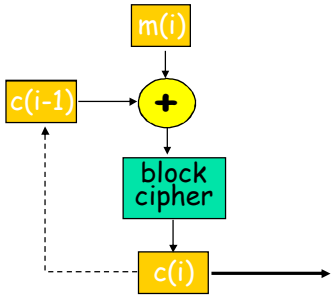
- DES con chiave a 56 bit ha mostrato i suoi limiti
  - 1997 DES Challenge: plaintext decifrato (brute force) in **4 mesi**
  - Nel 1998 Electronic Frontier Foundation con hardware ad hoc "Deep Crack" esplora key space in **56 ore**
  - Servizio online (con 48 FPGA Virtex)
    - a pagamento, brute-force in **~26 ore**
    - gratis su chosen-plaintext 1122334455667788 **25 sec** con 99.5% di successo (rainbow table)
  - nessun attacco "backdoor" noto
- per rendere DES più sicuro:
  - usare **tre chiavi** sequenzialmente (3-DES) su ogni blocco
  - usare cipher-block chaining

Cipher Block Chaining

- Debolezza dei cifrari a blocchi: blocchi in input uguali producono lo stesso testo cifrato
  - Si prestano a crittoanalisi



- Per ovviare a questo inconveniente si usa la tecnica detta cipher block chaining:
  c(i) si calcola con l'algoritmo di cifratura applicato al blocco ottenuto tramite XOR del testo in chiaro m(i) con il blocco di testo cifrato c(i-1) calcolato sull'input precedente
  - c(0) trasmesso in chiaro (vettore di inizializzazione)



Neanche 3DES è più sicuro

- La dimensione del blocco (64 bit) implica che, cifrati abbastanza dati con la stessa chiave, si verificheranno ripetizioni (**birthday attack**)
- Se il blocco è lungo m, la probabilità di ripetizioni è significativa dopo $2^{m/2}$ messaggi
- Ripetizioni in cyphertext -> rivelato XOR di plaintext
  - in caso di know-plaintext or chosen-plaintext viene rivelata parte della chiave

Cipher	Key-length		Cryptanalysis		Block size	
DES	56 bits	✗	2^{40}	✗	64 bits	✗
3DES	168 bits	✓	2^{112}	✓	64 bits	✗
Blowfish	32—448 bits	✓	None	✓	64 bits	✗
RC4	40—2048 bits	✓	2^{8}	✗	stream	✓
AES	128—256 bits	✓	Related-key	✓	128 bits	✓

[bhargavan2016] K. Bhargavan, G. Leurent , *"On the Practical (In-)Security of 64-bit Block Ciphers -- Collision Attacks on HTTP over TLS and OpenVPN"*, ACM CCS 2016

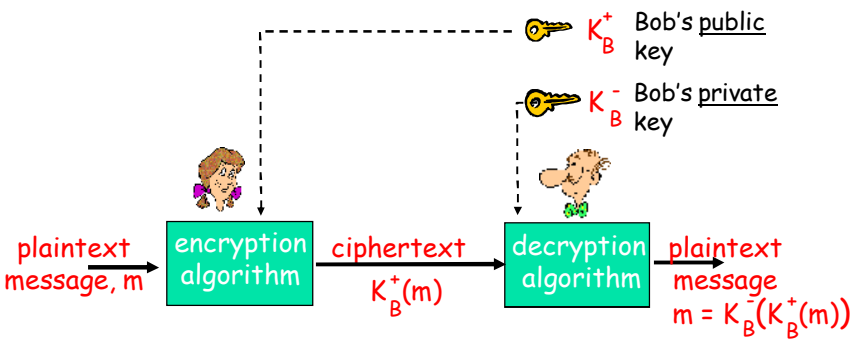
AES: Advanced Encryption Standard

- Standard NIST 2001, ha sostituito DES (1997)
  - Scelto da valutazione di algoritmi proposti in gara pubblica
  - Basato su Rijndael
    (inventato dai crittografi belgi Joan Daemen e Vincent Rijmen)
- Elabora i dati a blocchi da 128 bit
- Chiavi da 128, 192, o 256 bit
- Algoritmo di decifratura diverso da quello di cifratura (diversamente da DES)
- Un elaboratore in grado di decifrare DES con un attacco a forza bruta in 1 secondo, impiegherebbe 149 trilioni di anni a decifrare AES
- Attacco noto: «Related-key» è attualmente impraticabile https://eprint.iacr.org/2009/317

Crittografia a chiave pubblica

- Approccio radicalmente differente
  - Diffie-Hellman 1976, RSA78
- Mittente e destinatario non condividono una chiave segreta
- *Chiave pubblica* usata per la cifratura, nota a tutti
- *Chiave privata* usata per la decifratura, nota solo al destinatario

Public key cryptography



Algoritmi di cifratura a chiave pubblica

Requisiti:

1. Occorre trovare una coppia di chiavi $K_B^+(\cdot)$ and $K_B^-(\cdot)$ tali che

$$K_B^-(K_B^+(m)) = m$$

2. Nota la chiave pubblica K_B^+, dovrebbe essere impossibile calcolare la chiave privata K_B^-

Algoritmo RSA: Rivest, Shamir, Adleman

RSA: scelta delle chiavi (1/2)

1. Si scelgono due numeri primi grandi p e q
2. Si calcolano $n = p \cdot q$, $z = (p-1)(q-1)$
3. Si sceglie un numero e $(e<n)$ che non abbia fattori comuni con z (e, z "co-primi")
4. Si sceglie un numero d tale che $e \cdot d-1$ sia multiplo di z (in altre parole: $e \cdot d \bmod z = 1$)
5. La chiave pubblica è (n,e), la chiave privata è (n,d)

(n,e) = K_B^+ $\quad$ (n,d) = K_B^-

RSA: scelta delle chiavi (2/2)

1. Si scelgono due numeri primi grandi p e q — *modulo* RSA
2. Si calcolano $n = p \cdot q$, $z = (p-1)(q-1)$ — funzione *toziente* di n, o *ordine* del gruppo moltiplicativo Z^*_n
3. Si sceglie un numero e $(e<n)$ che non abbia fattori comuni con z (e, z "co-primi") — esponente di encryption
4. Si sceglie un numero d tale che $e \cdot d-1$ sia multiplo di z (in altre parole: $e \cdot d \bmod z = 1$) — esponente di decryption
5. La chiave pubblica è (n,e), la chiave privata è (n,d)

(n,e) = K_B^+ $\quad$ (n,d) = K_B^-

RSA: cifratura e decifratura

0. Dati (n,e) ed (n,d) come calcolati precedentemente …
1. La cifratura del testo in chiaro m *(stringa di bit)* si effettua calcolando

   $c = m^e \bmod n$ (resto della divisione di m^e per n)
2. La decifratura del testo cifrato c *(stringa di bit)* si effettua calcolando

   $m = c^d \bmod n$ (resto della divisione di c^d per n)

$$m = \underbrace{(m^e \bmod n)}_{c}{}^d \bmod n$$

RSA: esempio

Bob sceglie $p=5$, $q=7$ $\rightarrow$ $n=35$, $z=24$

$e=5$ (in modo che e, z siano relativamente primi)
$d=29$ (in modo che $e \cdot d-1$ sia divisibile per z)

Bob rende pubblica la coppia (n,e)=(35,5)
e mantiene segreto il valore d=29

	lettera	m	m^e	$c = m^e \bmod n$
cifratura:	l	12	248832	17

	c	c^d	$m = c^d \bmod n$	lettera
decifratura:	17	48196857210675091509141182522307169 7	12	l

RSA: perché $m = (m^e \bmod n)^d \bmod n$?

Un utile risultato di teoria dei numeri

Se p e q sono primi ed $n = p \cdot q$, allora:

$$x^y \bmod n = x^{y \bmod (p-1)(q-1)} \bmod n$$

$$(m^e \bmod n)^d \bmod n = m^{e \cdot d} \bmod n$$

$$= m^{e \cdot d \bmod (p-1)(q-1)} \bmod n$$

(grazie al risultato di sopra)

$$= m^1 \bmod n$$

(dal momento che $e \cdot d$ è stato scelto in modo che la sua divisione per $(p-1)(q-1)$ dia resto 1)

$$= m$$

RSA: un'altra proprietà notevole

$$K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$$

Si usa prima la chiave pubblica e poi la privata

Si usa prima la chiave privata e poi la pubblica

Il risultato è lo stesso → le due chiavi possono invertirsi i ruoli

RSA: discussione

- La robustezza di RSA dipende dal fatto che non sono noti **algoritmi veloci per la fattorizzazione** di numeri interi grandi
- Anche se $n = p \cdot q$ è noto, i due fattori p e q non sono velocemente determinabili, di conseguenza non è velocemente determinabile z, da cui si potrebbe facilmente risalire alla componente d della chiave privata, nota la parte e della chiave pubblica
- L'elevamento a potenza richiesto da RSA è computazionalmente oneroso
  - Algoritmi a chiave simmetrica come DES sono 100-1000 volte più veloci di RSA
  - Nelle comunicazioni su rete, RSA è spesso usato in combinazione con DES o 3DES

RSA: sicurezza (1/2)

- In [boneh1999] sono considerati attacchi derivanti da scelte improprie dei parametri o uso improprio dell'algoritmo
  - p-1 è prodotto di primi piccoli -> tempo di fattorizzazione basso
  - **modulo comune** (n condiviso da più utenti)
  - **blinding**: firma di messaggio contraffatto (eliminato da hash pre-firma)
  - **esponente privato d piccolo** -> consigliato n > 1024 bit (d~512bit)
  - **esponente pubblico e piccolo** (teorema di Coppersmith) -> consigliato e=65537
  - Attacchi sull'implementazione
    - **side-channel** in **tempo** o **potenza**: misura precisa -> ricava d
    - **Guasti transitori** "random faults" (e.g. sbalzi di tensione) -> fattore di n
    - **Bleichenbacher**: con random padding, errore di formato fa da oracolo per attacco chosen-ciphertext

[boneh1999] Boneh, Dan. "Twenty years of attacks on the RSA cryptosystem." Notices of the AMS 46.2 (1999): 203-213.

RSA: sicurezza (2/2)

- In [lenstra2012] analisi delle chiavi pubbliche RSA
  - Frequenti casi problematici nella scelta di n
  - Casi di scelte problematiche di e (probabile d piccolo)
- Raccomandazione European Union Agency for Network and Information Security [enisa]: lunghezza(n) > 15'360 bit
- Elliptic Curve Cryptography (ECC)
  - considerato più robusto di RSA (a parità di lunghezza delle chiavi)
  - Si basa su onerosità di calcolo del logaritmo intero
- Entrambi soccombono al Quantum Computing
  - oggetto di ricerca

[lenstrsa2012] Lenstra, Arjen, et al. "Ron was wrong, Whit is right". No. EPFL-REPORT-174943. IACR, 2012.

[enisa] Algorithms, key size and parameters report 2014, https://www.enisa.europa.eu/publications/algorithms-key-size-and-parameters-report-2014/at_download/fullReport