Sicurezza Reti e Calcolatori

Daniel Biasiotto

[2022-03-09 Wed 17:01]

CONTENTS

- Prof: Bergadano Francesco
- PDF Version

1 CIFRARI SIMMETRICI

Cifrari sono sempre esistiti, tra i cifrari pre-informatici piú famosi ci sono i cifrari simmetrici `character-oriented`:

- Cifrario di Cesare, cifrari monoalfabetici a 1 lettera
- Cifrario di Playfair, monoalfabetico a 2 lettere
- Cifrari monoalfabetici a N lettere
- Cifrario di Vigenére, polialfabetico

  I cifrari polialfabetici sostituiscono una lettera ogni volta in modo diverso, a seconda della sua posizione nel testo.

Questi cifrari si possono ancora suddividere in base alla tecnica utilizzata:

- a sostituzione
- a permutazione

Da quest'ultimo derivano i cifrari simmetrici `bit-oriented`:

- Cifrario di Vernam
- One-time Pad

In questi cifrari al posto dell'operazione di sostituzione alfabetica viene utilizzato $\oplus$[1]

I cifrari simmetrici moderni sono caratterizzati da:

- uso del calcolatore
- combinazione di permutazioni e sostituzioni
- uso di numerose fasi, *round*

Di questi ne esistono diversi:

- Macchine a Rotori
- Feistel Cipher
- DES
- AES

Una proprietá desiderabile in un encryption algorithm é chiamata *avalanche effect*

- un cambiamento marginale in un input (chiave o plaintext) dovrebbe produrre un grande cambiamento nel ciphertext

Queste tecniche sono utilizzate nel contesto della *bulk encryption*

1 Exclusive Or

1.1 Cifrari a blocchi

Utilizzando chiavi lunghe e testi arbitrariamente lunghi

1. cifrare a 2 fasi
   - suscettibile all'attacco *meet in the middle*
     - con *known plaintext*
     - conoscendo `<P1,C1> <P2,C2>`
       * servono estrambe per incrociare la ricerca, i match sono diversi per blocco
       * ci sono molte piu' chiavi che blocchi
     - *brute force* sulla prima fase di cifratura, su 2^{56} possibilita' su DES
2. cifrare a 3 fasi
   - `triple DES` o `3DES`
   - sicuro, chiave di $3 \cdot 56 = 168$
   - normalmente si utilizza `K1 = K3`
     - la forza sta nelle 3 fasi, non nelle 3 chiavi
   - si puo' utilizzare `3DES-EDE` con 3 chiavi uguali, che equivale a `DES`

Per *plaintext* lunghi si hanno diverse tecniche per creare un messaggio cifrato a partire dai blocchi:

- **Electonic Codebook**
  - molto semplice ed efficiente ma insicuro
  - divisione in blocchi esatti e criptarli tutti con la stessa chiave
    * parti di testo uguali avranno blocco *ciphertext* uguali
    * vulnerabilitá alla criptoanalisi statistica, utilizzabile solamente con testi corti
- **Cipher Block Chaining**
  - ogni blocco cifrato e mette in $\oplus$ con il successivo plaintext
    * si decifra con un $\oplus$ tra la decrittazione del blocco corrente C_i e il blocco precedente (cifrato) C_{i-1}
  - il primo blocco é in $\oplus$ con un *initialization vector* IV
    * solitamente publico
  - il piú usato, sicuro, semplice, efficiente
  - un errore di 1 bit rende indecifrabile il blocco successivo

- **Cipher FeedBack**
  - cifrario a flusso
  - simile al Cifrario di Vernam
  - inefficiente, viene scartato del lavoro
  - un errore di un bit essendoci feedback crea *effetto valanga*
- **Output Feedback**
  - molto simile al Cipher Feedback
  - il feedback é fatto utilizzando gli i bit di output del cifrario a blocchi
  - di fatto si divide in 2 fasi la procedura
    1. prima di conoscere il testo si produce la sequenza di i bit
    2. utilizzare questa informazione bufferizzata per cifrare in $\oplus$
  - simile al One-time Pad e al Cifrario di Vernam
    * solo simile in quanto il vettore di i e' solo pseudocasuale

1.2 Metodi dell'avversario

L'avversario puó decodificare i cifrari monoalfabetici a una lettera facilmente attraverso una Crittanalisi Statistica.

Questa analisi risulta molto piú difficile con un cifrario polialfabetico:

- in conoscenza di n é possibile fare la stessa analisi per lettere che distano n posizioni nel testo
  - per cui quindi vale la stessa sostituzione

Di conseguenza un testo cifrato di questo tipo risulta tanto piú facile da decifrare tanto é piú lungo, ancor di piú in presenza di parti di testo fisse.

2 CIFRARI ASIMMETRICI

Si utilizzano 2 chiavi, una per criptare e una per decriptare Le due chiavi non sono solo diverse nella forma, sono generate insieme e non é possibile ottenere una dall'altra La difficoltá per un avversario

non é piú informativa ma **computazionale** Questi cifrari non sostituiscono quelli tradizionali, simmetrici, in quanto piú impegnativo a livello computazionale, infatti i primi sono molto recenti (Diffie-Hellman Key Exchange).

- il protocollo piú utilizzato in questo ambito é RSA.
- sono spesso combinati con cifrari simmetrici e funzioni di hash
  - vedi Digital Envelope

É possibile classificare l'uso di questi sistemi in:

1. **Encryption/Decryption**
   - sender encrypts with recipient public key
2. **Digital Signature**
   - sender signs with its private key
3. **Key Exchange**
   - parts work together to exchange a common secret key

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

3 FUNZIONI DI HASH

Una funzione di Hash H accetta un blocco di dati M di lunghezza variabile e produce un valore di hash $h = H(M)$ di lunghezza fissa.

- una buona funzione di Hash ha la proprietá che applicata a un gran numero di input gli output siano ben distribuiti e apparentemente random
- un cambiamento a un qualsiasi bit o bits in M causa, probabilmente, un cambiamento nel codice hash generato

In crittografia si usa un particolare tipo di funzione di hash, che ha ulteriori proprietá:

- `one-way property`
  - *infeasible to find an object mapping to a pre-specified hash*
- `collision-free property`

  - *infeasible to find two objects mapping to the same hash*

Queste funzioni di hash sono utilizzate per:

- autenticare messaggi con i `message digest`
  - *sender* e *recipient* applicano entrambi la funzione e comparano i risultati
- `digital signature`
- `one-way password file`
- `intrusion detection`
- `virus detection`

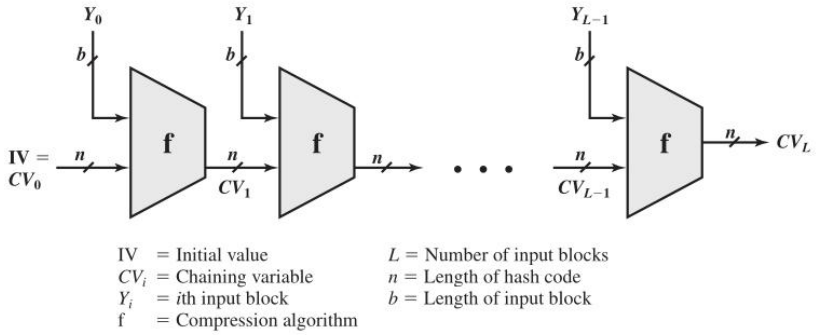

Figure 11.8 General Structure of Secure Hash Code

La funzione di hash piú utilizzata in tempi recenti é stato il Secure Hash Algorithm

Un *birthday attack* é effettuato generando collissioni:

- 2^m messaggi
- codici di c bit
- P(collision) > 0.5 per $m > \frac{c}{2}$
  - quindi per 64 bit bastano 2^{32} messaggi

Quindi un attaccante puó facilmente creare collisioni, ma il messaggio di cui il digest colliderá sará comunque incomprensibile, questo attacco é utile quando il ricevente si aspetta numeri o stringhe arbitrarie e non noterá nulla di strano nel messaggio ricevuto. Questi risultati impongono digest con almeno 256 bit.

4 AUTENTICAZIONE

NB Un messaggio cifrato non é necessariamente autentico, un messaggio autenticato puó essere leggibile. Spesso questi ultimi non vengono cifrati.

4.1 Simmetrica

- basata su **cifrari simmetrici**
- chiave condivisa

$\text{MAC}_K(M)$ - `Message Authentication Code`

1. DES-CBC - `MAC-CBC`
   - si usa l'ultimo blocco cifrato (o una parte) come `MAC`
2. *Keyed Hash Function* - `HMAC`
   - `MAC` generato applicando H a una combinazione di M e una chiave segreta
   - $\text{HMAC}_K(M) = H((K'' \oplus \text{opad}) || H((K'' \oplus \text{ipad}) || M'))$
     - K'': una chiave segreta K' con padding di o fino a j bit
       - se maggiore di j bit $K'' = H(K')$
     - ipad: 00110110 ripetuto j/8 volte
     - opad: 01011010 ripetuto j/8 volte
   - efficiente quanto H
     - molto piú efficiente che `MAC-CBC`

4.2 Firma elettronica

- basata su **cifrari asimmetrici**
- firma con la chiave *privata*, verifica con la chiave *pubblica* di chi firma

In questo caso:

1. RSA con `MD5/SHA-1`
   - $\text{SHA-1}(M)$: *digest*
   - $\text{RSA}(K^-(A), \text{digest})$
2. `DSA` con `SHA-1`

Per far funzionare questo meccanismo é necessario risolvere il problema della distribuzione delle chiavi pubbliche. Questo in quanto rimane possibile un **Man in the Middle attack**.

- una terza parte `C` puó ricevere $\langle \text{ID}, K^+(\text{ID}) \rangle$ e restituirne un certificato
- questo poi viene condivisto da altre terze parti o dagli stessi che lo hanno richiesto

- il certificato di chiave pubblica é un documento che attesta l'associazione univoca tra chiave pubblica e l'identitá del soggetto
- queste operazioni sono eseguite da un ente fidato, `Certification Authority` o `CA`
  - un attaccante pur sostituendo una chiave certificata *sniffata* non puó sostituirla con la propria, non ha accesso alla chiave privata della `CA` e non puó crearsi un certificato falso

Alla fine il messaggio autenticato avrá la forma: `M - FirmaElettronica - Certificato - Timestamp`

5 SNIFFING & SPOOFING

1. *sniffing*
   - non facile su rete geografica
   - possibile su `LAN`
     - sia su switch che non
     - non é possibile su *switch unicast*
     - solo su *broadcast*
2. *spoofing*
   - `ARP` spoofing/poisoning
   - `DHCP` associa automaticamente `IP` di router e `DNS`
   - `ARP` associa `MAC-IP`
     - *broadcast* per la richiesta del `MAC` associato a un `IP`
     - *unicast* per la risposta
     - l'avversario risponde con il proprio `MAC` ingannando il richiedente
   - possibile tecnica per:
     - `MAC`
       * scheda di rete in modalitá promiscua
       * `MAC` della scheda cambiato malevolmente
     - `IP`
       * non in `TCP` dove c'é il *3-way handshake*
     - `DNS`
       * instradamento degli utenti verso un `DNS` malevolo
       * `DNS` malevolo serve `IP` falsificati

    - URL
        * indirizzi falsi

Per evitare questi attacchi:

- non usare HUB ma switch
- non usare *broadcast*
- cifrare a livello applicativo e a livello di trasporto

6 DDOS

- raro
- difficile da evitare per i principi costituenti della rete
    - per applicazioni critiche é utile avere reti dedicate

Possibili attacchi:

1. syn flooding
    - primo messaggio dell'handshake TCP senza che questo sia poi portato a termine
2. ICMP echo request
    - distibuted, *zombie* e *reflectors*
    - *smurf attack*
        - echo request con payload consistente
            * possibilitá pensata per testing di rete, echo in broadcast
            * ora non piú possibile
3. relay SMTP
    - flooding tramite server mail
    - possibili configurazioni server per evitare questi attacchi

7 FIREWALL

- vulnerabilitá locali di una macchina possono permettere il controllo della rete intera
- un PC compromesso in LAN permette attacchi diretti alla rete locale

- il `Firewall` si interpone tra `LAN` e `WAN` come unico punto di accesso
  - servizi di
    - filtro (direzione, servizio, utente)
    - log (traffico, utenti)
    - allarme
  - incluso nel *router*, `screening router`
    - scarta i pacchetti sospetti
    - non notifica
  - `dual homed gateway`
    - tra `LAN` e *router*
    - il router si occupa di routing
    - spesso comunque tutte le funzioni sono concentrate in un unico dispositivo
    - dispositivi specializzati: *firewall appliance*
  - `screened host gateway`
    - fisicamente i pacchetti non sono forzati attraverso il `FW`
    - si forza il passaggio a livello logico `IP`

Spesso in sicurezza, e anche per questi dispositivi, si parla di *High Availability*

- piú `FW` possono servire in parallelo per garantire la funzionalitá in caso di guasti
- Internet → Router → Switch → FW | FW → Switch → `LAN`

Una `DMZ` é una cosiddetta

- *DeMilitarized Zone*
- server che devono poter comunicare con l'esterno senza interferenze dall'`FW`

7.1 Package Filter

- livello 3 e parzialmente 4
  - `IP` e `TCP/UDP`
- protegge in base alla direzione
  - interfaccia in/out
  - `IP` mittente e destinatario

  - porta sorgente e destinazione
- la *frammentazione IP* puó essere usata per passare attraverso un `FW`
  - piccoli frammenti 24-28 Byte, senza header `TCP`
- da bloccare il *source routing*
  - permette al mittente di decidere l'instradamento
  - permette `IP` spoofing con `TCP` su `WAN`
- `ACL` - Access Control List
  - omonimo con sistema *Windows*, diversi contesti e usi
  - lista di regole di accesso

7.2 Sofware Firewall

- livello 5
  - applicativo e di trasporto `TCP/UDP`
- piú semplice attraverso un `proxy-FW`
  - va configurato un *proxy* per ogni servizio da attivare
  - non é trasparente
  - piú lento
  - sicuro, sofisticato
- mascheramento degli indirizzi tramite `NAT`
  - megli il `NAPT`
    - unico indirizzo pubblico
    - indirizzi tradotti assieme alle porte
  - puó anche effettuare *load balancing*
    - round robin, evita attacchi di carico
- `WAF` - Web Application `FW`
  - *reverse proxy*
  - esamina il payload applicativo
  - solo se sicura apre la connessione al nostro server web e inoltra

8 VPN

Standard: IPsec

- permette collegamento a rete privata virtualmente
  - lavorare da remoto con la stessa sicurezza che si ha all'interno della LAN
- traffico *virtualmente interno* passa su internet e va protetto

8.1 IPsec

IP level security

- livello 3
- RFC 1825
- layer che si va a inserire sopra quello IP
  - header annidato all'interno dell'header IP
  - PDU cifrata/autenticata assieme a info per decifrazione
  - l'header IP non viene modificato
    - i router non si accorgono del cambiamento
- protezione da modifica e intercettazioni
- cifratura ai capi della comunicazione tra le LAN
- ovviamento non protegge da vulnerabilitá interne

Due modalitá di funzionamento:

1. transport
2. tunnel

E tecniche

1. AH
2. ESP

Queste tecniche sono annidabili

- prima applicando AH e poi ESP

8.1.1 Transport

- software VPN sui calcolatori comunicanti
- protegge da spoofing/sniffing si rete locale
- non é trasparente, necessaria configurazione
- unico metodo per una postazione mobile
  - sono possibili soluzioni miste

8.1.2 Tunnel

- cifratura/auth da parte di un agente esterno *terminatore*
  - spesso incluso nel router e FW
  - i pacchetti escono dal *tunnel* decriptati
- non protegge da spoofing/sniffing su rete locale
- nasconde gli indirizzi
  - sono solamento noti gli IP dei *terminatori*
- trasparente
- veloce, efficiente

8.1.3 Authentication Header

AH

- garantisce integritá
- posizionato tra header IP e PDU
- formato
  - Next Header
    - 8B
    - protocollo superiore
  - Length
    - 8B
  - Reserved
    - 16B
  - SPI
    - 32B
    - Security Parameter Index

        * parametri (entrambi indici di una tabella interna condivisa)
            · tipo di algoritmo
            · chiave simmetrica
    – Data
        * $N×$32B
        * dati di autenticazione `MAC`
        * questo `MAC` coper da `header IP` in poi
            · ignora campi variabili `TTP` e `checksum` impostandoli a 0

8.1.4 Encapsulating Security Payload

`ESP`

- posizionato dopo `header IP` e incapsula il `PDU` cifrato
- formato in modalitá *Transport*
    – `SPI`
        * non cifrato
    – `PDU`, Next Header, autenticazione
        * cifrati
- formato in modalitá *Tunnel*
    – `SPI`
        * non cifrato
    – `header IP` incapsulato
        * cifrato
        * header originale nascosto dal terminatore `VPN`
        * funzione di offuscamento del traffico
    – `PDU`, `NH`, auth
        * cifrati

8.1.5 Anti-Replay

- individua ripetizione pacchetti
    – non é possibile escludere che non creino problemi a livello applicativo
- pacchetti `IPsec` numerati con un *sequence number* 16bit
- tecnica a *sliding window* con *W* bit

- implementazione con un *bit vector*
- N ultimo *sn* ricevuto
- finestra da $N - W$ a $N + 1$
  * *sn* ricevuto a sinistra della finestra, non posso decidere
  * *sn* ricevuto a destra, sicuramente nuovo
  * *sn* all'interno il vettore indica se é stato ricevuto o no

9 WEB SECURITY

10 BLOCKCHAIN