SICUREZZA INFORMATICA

Quando si parla di sicurezza dobbiamo innanzitutto parlare dei possibili attacchi ad un sistema.
Gli attacchi si dividono in due categorie:
- ATTACCHI PASSIVI che riguardano l'intercettazione o il monitoring di trasmissioni.
- ATTACCHI ATTIVI che riguardano una reale modifica del flusso di dati.

ATTACCHI PASSIVI

Essi riguardano l'intercettazione e il monitoraggio di trasmissioni di dati.
L'obiettivo principale degli attacchi passivi è riuscire ad ottenere le informazioni circa le trasmissioni di dati.
Vi sono due tipologie di attacchi passivi:
1. **Intercettazione del contenuto dei messaggi** che comprende conoscere effettivamente il contenuto dei messaggi (come ascoltare una conversazione telefonica,leggere una mail ecc.)
Il pericolo sta nel fatto che in alcuni casi possono essere trasmesse informazioni delicate
2. **Analisi del traffico** che in qualche modo comprende il non riuscire a leggere il contenuto di un messaggio perchè ad esempio potrebbe essere cifrato ma riuscire lo stesso ad estrarre informazioni come la lunghezza dei messaggi scambiati,l'identità degli host coinvolti nella comunicazione,o genericamente la natura della comunicazione.
Gli attacchi passivi sono difficili da rilevare (è impossibile sapere se qualcuno sta ascoltando una conversazione telefonica) in quanto essi non alterano i dati,però sono più semplici da prevenire ad esempio possiamo cifrare i dati scambiati facendo si che l'attaccante non possa essere a conoscenza del contenuto del messaggio.

ATTACCHI ATTIVI

Essi riguardano la modifica del flusso dei dati o addirittura la creazione di un flusso totalmente diverso da quello originale.
Vi sono 4 tipi di attacchi attivi:
1. **Mascheramento** esso consiste in una entità(chiamata attaccante) che finge di essere un'altra entità, di solito tale attacco prevede anche altri tipi di attacchi attivi,cioè un attaccante si maschera per poter effettuare un altro attacco attivo senza essere riconosciuto.
2. **Ripetizione** che consiste nella cattura passiva dei dati e nella riproduzione di essi in maniera non autorizzata
3. **Modifica dei messaggi** che consiste nell'alterazione dei messaggi(quindi nel riordine o ritardo)
4. **Denial of Service**(DoS) che impedisce il normale utilizzo o la gestione di un sistema di comunicazione
A differenza degli attacchi passivi che sono facili da prevenire ma difficili da rilevare,gli attacchi attivi dato che riguardano essenzialmente la modifica reale dei dati sono semplici da rilevare ma difficili da prevenire.

PAROLE CHIAVI

Le parole chiavi di un sistema di sicurezza sono:

- **Autenticazione**: un sistema di sicurezza deve essere in grado di assicurare l'autenticità del mittente/destinatario in una comunicazione, quindi le entità coinvolte devono essere autentiche(cioè devono essere effettivamente chi dicono di essere).

- **Controllo degli accessi**: un sistema di sicurezza deve essere in grado di impedire un accesso non autorizzato ad una risorsa(quindi si intende che il sistema deve essere in grado di decidere chi può accedere ad una risorsa,in che condizioni e che operazioni può effettuare su tale risorsa).

- **Segretezza dei dati**: qualora vi sia un accesso non autorizzato il dato in qualche modo deve essere protetto

- **Integrità dei dati**: il sistema di sicurezza deve garantire che i dati ricevuti siano effettivamente quelli che erano stati inviati(senza modifiche,ripetizioni o eliminazioni)

- **Non ripudiabilità**: il sistema di sicurezza deve essere in grado di evitare che un mittente o un destinatario siano in grado di negare di aver partecipato ad una comunicazione (alla quale avevano partecipato).

MODELLO DI SICUREZZA PER LE RETI

Alla base di un sistema di sicurezza vi devono essere necessariamente due componenti:

1. **Trasformazione di sicurezza** del messaggio che può comprendere la cifratura del messaggio quindi la modifica/trasformazione del messaggio secondo un accordo tra mittente e destinatario oppure l'aggiunta di codice al messaggio che permette al destinatario di verificare l'identità del mittente.

2. **Un'informazione segreta** associata al messaggio che ad esempio potrebbe essere la chiave segreta associata ad un sistema di cifratura che permette lato sorgente di andare a cifrare il messaggio e lato destinazione di andare a decifrarlo. Ovviamente tale informazione segreta non deve essere conosciuta dall'attaccante.

CRITTOGRAFIA CLASSICA

La cifratura viene suddivisa in due modelli:
1. **Modello di cifratura simmetrico**(chiave unica) 2.**Modello di cifratura asimmetrico**(chiave pubblica)

Un qualsiasi modello di cifratura per essere minimamente sicuro deve soddisfare entrambi i seguenti due **requisiti**:
1. **L'algoritmo di cifratura/decifratura è pubblico**, quindi la sicurezza non deve basarsi sull'algoritmo ma sulla chiave segreta. In particolare l'attaccante pur conoscendo l'algoritmo di cifratura/decifratura e qualche testo cifrato non deve essere in grado di ottenere il relativo testo in chiaro o la relativa chiave segreta

2. **La chiave segreta non deve essere scoperta** quindi sia mittente che destinatario devono assicurare la segretezza di tale chiave.

MODELLO DI CIFRATURA SIMMETRICO

Esso prevede 5 elementi:
- **Testo in chiaro**: rappresenta il messaggio originale che deve essere spedito
- **Algoritmo di cifratura:** è un algoritmo che va ad effettuare una trasformazione del testo in chiaro in testo cifrato.
- **Chiave segreta:** è un testo che rappresenta l'informazione segreta che mittente e destinatario devono scambiarsi. Tale informazione verrà data in input insieme al testo in chiaro all'algoritmo di cifratura.
- **Testo cifrato** : rappresenta l'output restituito dall'algoritmo di cifratura che ha preso in input un testo in chiaro e una chiave segreta. Nota bene: ogni testo cifrato è associato a un testo in chiaro e a una chiave segreta.
- **Algoritmo di decifratura** che prende in input un testo cifrato e una chiave segreta e da in output il relativo testo in chiaro.

ATTACCO AD UN SISTEMA CRITTOGRAFICO

Gli attacchi ad un sistema di cifratura non consistono solamente nell'individuare il testo in chiaro associato ad un testo cifrato,piuttosto si cerca di trovare la chiave segreta utilizzata da mittente e destinatario per lo scambio di messaggi.
Gli attacchi ad un sistema crittografico possono essere suddivisi in due tipologie:
- **CrittoAnalisi**: si cerca di sfruttare le caratteristiche dell'algoritmo di cifratura (ad esempio conoscendo qualche coppia testo cifrato/testo in chiaro) per riuscire a dedurre la chiave segreta
- **Attacco di forza bruta**: vengono tentate tutte le possibili combinazioni di una chiave finchè non si trova la chiave corretta (quindi si ha una traduzione del testo sensata). Dal punto di vista probabilistico tale attacco viene portato a termine con almeno la metà dei possibili tentativi.

Le **tipologie di attacchi di tipo CrittoAnalisi** ad un sistema di cifratura possono essere scomposti in base a determinate proprietà:

- **CipherText Only**: quando l'attaccante conosce solo il testo cifrato e in alcuni casi anche l'algoritmo di cifratura. In tal caso l'attacante può procedere con un attacco di forza bruta che può essere attuato quando lo spazio delle chiavi (cioè il numero totale possibile di chiavi) è piccolo,altrimenti può procedere ad analizzare informazioni del testo, ad esempio se sa che il testo è scritto in inglese o italiano, se conosce delle parti di testo in chiaro con associato il testo cifrato e così via.

- **Known Plain Text**: quando l'attaccante conosce il testo in chiaro quindi ascoltando la comunicazione tra mittente e destinatario riesce a ricavarne la chiave segreta

- **Chosen Plain text**: quando l'attaccante è in grado di scegliere i pezzi di testo in chiaro,quindi conosce l'algoritmo di crittografia,il testo in chiaro con il relativo testo cifrato e va ad analizzare tali informazioni cercando di ottenere la chiave segreta

- **Chosen Chiper Text**: quando l'attaccante conosce l'algoritmo di cifratura e il testo cifrato ed è in grado di decifrare delle parti di testo cifrato generato con la chiave segreta

- **Chosen Text** quando l'attaccante conosce l'algoritmo di cifratura e ha un testo cifrato con associato il relativo testo in chiaro ed è in grado di ottenere un presunto testo cifrato associato al testo in chiaro generato con la chiave segreta.

TECNICHE DI CRITTOGRAFIA CLASSICA

Le tecniche di cifratura classica si basano su due tipologie di tecniche :
- **tecnica a sostituzione** che prevede la sostituzione di una lettera del testo in chiaro con una lettera del testo cifrato
- **tecnica di trasformazione**.

TECNICHE A SOSTITUZIONE

► *Cifrario di Cesare*

E' tecnica di cifratura classica più antica,consiste nel sostituire ogni lettera del testo in chiaro con la relativa lettera di tre posizioni più a destra del testo cifrato.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

$$X \leftarrow M+3 \bmod 26$$

E' semplice capire che esso può essere facilmente decifrato in quanto l'algoritmo di cifratura lo conosciamo e inoltre le possibili chiavi sono 25 (shiftiamo l'alfabeto di 1 posizione ogni volta).

► *Cifratura monoalfabetica*

Tale tecnica utilizza l'idea alla base del cifrario di Cesare solo che la lettera del testo in chiaro non viene scambiata con la lettera che si trova 3 posizioni dopo di esso ma con un numero arbitrario di posizioni.
Praticamente l'alfabeto non viene shiftato di 3 posizioni ma con un numero arbitrario di posizioni
Quindi lo spazio delle chiavi è di 26! possibili combinazioni.

Vi è un possibile attacco dovuto al fatto che l'attaccante potrebbe conoscere la natura del testo ed effettuare una crittoanalisi per ricavarne la frequenza delle lettere e ad esempio confrontandola con la frequenza delle lettere di un alfabeto(ad esempio se io attaccante so che il messaggio è in inglese prendo il libro di frequenze inglesi e confronto l'analisi con tale libro e potrei ricavare le parole).
Quindi le cifrature monoalfabetiche sono semplici da violare in quanto conservano le frequenze delle lettere del testo originale.
La contromisura è quella di utilizzare i cosiddetti "omofoni" per la medesima lettera, ovvero assegnare ad una lettera (ad esempio la c) la cifratura di 1,4,5 scelti a rotazione,quindi la prima volta che incontro la c la sostituisco con 1,la seconda volta con 4 e la terza con 5, in modo da evitare attacchi dovuti alla frequenza della lettera.

► Cifratura di Playfair

Essa consiste in una matrice 5x5 in cui nelle prime posizioni viene inserita una parola chiave e nelle successive le restanti lettere dell'alfabeto (senza ripetizioni).
Quindi supponendo che la parola chiave sia MONARCHY la matrice sarà la seguente:

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Per la cifratura vengono seguite le seguenti regole:
1. La cifratura avviene per coppie di lettere
2. Se in una coppia vi sono due lettere uguali queste vengono separate da un carattere separatore (qualsiasi ad esempio x) quindi ad esempio BALLOON verrebbe trattato come segue BA LX LO ON
3. Se le lettere di una coppia si trovano nella stessa riga della matrice verranno sostituite con la lettera che si trova a destra(in ordine ciclico) ad esempio AR verrebbe sostituito con RM
4. Se le lettere di una coppia si trovano nella stessa colonna della matrice verranno sostituite con la lettera sottostante ad esempio MU verrà sostituito con CM
5. In tutti gli altri casi ogni lettera della coppia verrà sostituita con la lettera che si trova nella stessa riga e colonna dell'altra lettera che si trova in coppia ad esempio HS diviene BP

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

►Cifrario di Hill

E' un'altra cifratura multilettera dove m lettere del testo in chiaro vengono sostituite con m lettere di testo cifrato.
La cifratura avviene utilizzando una matrice K (che sarebbe la chiave) e due vettori denotati con C e P e un valore m (che rappresenta la lunghezza del blocco).
La cifratura avviene secondo il seguente schema (per m=3)

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{12} & k_{22} & k_{23} \\ k_{13} & k_{32} & k_{33} \end{bmatrix} \times \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} \mod 26$$

Esempio

Es. P = PAYMORENONEY

Chiave $\begin{bmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{bmatrix}$

m=3, primi 3 caratteri

- PAY = (15, 0, 24)

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{bmatrix} \times \begin{bmatrix} 15 \\ 0 \\ 24 \end{bmatrix}$$

- Testo cifrato: (11,13,18) = LNS

Per la decifratura viene utilizzata la matrice inversa K^{-1}. Quindi il requisito è che la matrice che funge da chiave deve essere invertibile.

►Cifratura polialfabetica

Rappresenta un miglioramento della cifratura monoalfabetica che consiste in sostituzioni monoalfabetiche differenti passo dopo passo.
Le tecniche di cifratura polialfabetica hanno le seguenti due caratteristiche:
1. Un insieme di regole di sostituzione monoalfabetica
2. Una chiave che determina la regola scelta per una determinata trasformazione

►Cifratura polialfabetica: Cifrario di Vigenerè

E' una particolare tecnica di cifratura polialfabetica che prevede l'attuazione della tecnica di cifratura di Cesare che ad ogni passo shifta l'alfabeto di una posizione a destra.

Quindi alla prima iterazione l'alfabeto cifrante sarà A B C D .... mentre alla seconda iterazione sarà B C D E... e così via.
E' possibile anche utilizzare la tabella di Vigenerè:

TESTO IN CHIARO

CHIAVE	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
b	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
c	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
d	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
e	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
f	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
g	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
h	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g
i	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h
j	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i
k	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j
l	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k
m	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l
n	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m
o	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n
p	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
q	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
r	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
s	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
t	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
u	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t
v	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u
w	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
x	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
y	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x
z	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y

Per cifrare basta prendere la cella ottenuta dall'intersezione della lettera testo in chiaro/chiave e per decifrare basta prendere la lettera che si trova nella colonna ottenuta dall'intersezione riga associata alla lettera della chiave e cella contenente la lettera del testo cifrato.
L'importanza di tale sistema è dovuta al fatto che ogni lettera del testo in chiaro ha più lettere del testo cifrato ad essa associate, quindi in molti casi quando viene utilizzato tale cifrario l'analisi della frequenza diviene un attacco quasi impossibile da effettuare.
Però anche tale cifrario non è esente da problemi, dato che per cifrare viene utilizzata una chiave un attaccante potrebbe andando ad analizzare le lettere cifrate capire quant'è lunga chiave e riuscire in qualche modo a decifrare il testo, ovvero se due sequenze di lettere in chiaro si ripetono esattamente dopo un multiplo della chiave vengono generate sequenze di lettere di testo cifrato identiche e da ciò l'attaccante potrebbe prima capire la chiave e poi decifrare parti del testo.
Per far fronte a tale problema vi sono state due possibili soluzioni:
- utilizzare una chiave lunga quanto il testo da cifrare (ma ovviamente si deve pensare a ricordarla)
- utilizzare la tecnica(proposta da Vigenerè) detta autokey ovvero affiancare alla chiave il testo da cifrare(ovviamente fin quando non si ottiene una chiave lunga quanto il testo da cifrare)

►Cifrario One Time Pad
Tale cifrario consiste nell'utilizzo di una chiave casuale lunga quanto il testo da cifrare e differente per ogni cifratura e decifratura.
Questo sistema è inviolabile in quanto essendo la chiave casuale non vi è possibilità di studiare la frequenza del testo.
Inoltre andando ad effettuare un attacco di forza bruta si otterrebbero più testi in chiaro SENSATI che non ci permetterebbero di capire quali tra essi è quello corretto.
Tale cifrario nonostante sia sicuro comporta due problemi:
- Trovare una chiave veramente casuale per ogni messaggio da scambiare
- Memorizzare e trasmettere una chiave lunga quanto il messaggio da cifrare.
Per tali motivi tale cifrario non può essere utilizzato.

TECNICHE A TRASPOSIZIONE

Nelle tecniche viste in precedenza ad ogni lettera del testo in chiaro era associata una lettera del testo cifrato. Nelle tecniche a trasposizione viene effettuata una permutazione delle lettere del testo in chiaro.

►Tecnica di Rail Fence

In cui il testo in chiaro viene scritto in diagonale e letto come una sequenza di righe. Molto semplice da capire.

MESSAGGIO SEGRETO

M S A G O E R T
E S G I S G E O

MSAGOERTESGISGEO

►Tecnica su matrice

Che consiste nello scrivere il testo in chiaro su una matrice e di avere una chiave che rappresenta la permutazione delle colonne, e di ottenere il testo cifrato leggendo le colonne in base alla chiave

Basate su matrici

EGGSOESIRMGEOAST

4	1	3	2	5
M	E	S	S	A
G	G	I	O	S
E	G	R	E	T
O				

DIFFERENZA TRA STEGANOGRAFIA E CRITTOGRAFIA

La sostanziale differenza è che mentre la crittografia trasforma il testo in chiaro in un testo non comprensibile a meno di un'informazione segreta, la steganografia nasconde totalmente il messaggio. Esempio: capelli dello schiavo

CIFRATURA A BLOCCHI

Una particolare tecnica di cifratura è la cifratura a blocchi che consiste nello scomporre il testo in chiaro in blocchi di testo, ognuno di questi blocchi viene cifrato in maniera differente ottenendo un testo cifrato delle stesse dimensioni del blocco di partenza.

CIFRATURA A FLUSSI E CIFRATURA A BLOCCHI

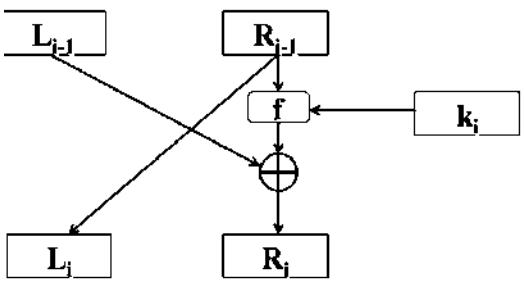
Nella cifratura a flussi i dati vengono cifrati uno per volta. Esempio Vigenerè.
Nella cifratura a blocchi il testo in chiaro viene scomposto in blocchi ed ognuno di essi viene cifrato come un testo differente.
Di solito la dimensione di questi blocchi è di 64 o 128 bit.
La cifratura a blocchi consiste nel prendere un testo in chiaro e scomporlo in blocchi di dimensione n e di ottenere blocchi di testo cifrato sempre di dimensione n. Esistono 2^n blocchi di testo in chiaro distinti e pertanto la crittografia può essere reversibile(cioè si può effettuare tranquillamente la decifratura).

Struttura di Feistel

Le cifrature a blocchi hanno una struttura detta di ***Feistel*** che consiste in una serie di fasi di elaborazione identiche in cui viene effettuata una sostituzione su una metà dei dati da elaborare e successivamente viene effettuata una permutazione che scambia le due metà.
La chiave viene estesa in modo da poter utilizzare in ogni iterazione una chiave differente.
Un algoritmo di cifratura che ha questa struttura è il DES.
L'idea di Feistel si basa su due idee avute dapprima da ***Shannon.***
Queste due idee si formalizzano nelle tecniche di diffusione e confusione.
Entrambe le tecniche sono state ideate da Shannon per cercare di evitare che il crittoanalista riesca ad analizzare la struttura statistica del testo cifrato e riuscire in qualche modo a risalire al testo in chiaro.
- La tecnica di ***diffusione*** consiste nell'associare ad una lettera del testo cifrato più lettere del testo in chiaro e quindi utilizzare una lettera del testo in chiaro per ottenere più lettere del testo cifrato.
In questo modo l'analisi della frequenza non può essere attuata in quanto tale frequenza viene sfasata.
- La tecnica di ***confusione*** consiste nell'applicare un algoritmo di sostituzione complesso (e non come quelli utilizzati nella CRITTOGRAFIA CLASSICA) in modo da complicare la relazione che vi è tra testo cifrato e chiave.

CIFRATURA DI FEISTEL

Nell'algoritmo di Feistel gli input sono un blocco di testo in chiaro di 2w bit e una chiave di K bit.
Il blocco è suddiviso in due metà che per convenzione sono indicate con R_0,L_0.
Queste due metà sono combinate per produrre il testo cifrato.
In una generica fase $_i$ l'input è L_{i-1} , R_{i-1} che derivano dalla fase precedente e una sottochiave K_i che deriva dalla chiave iniziale K. Si fa in modo che le sottochiavi K_i siano tutte differenti.

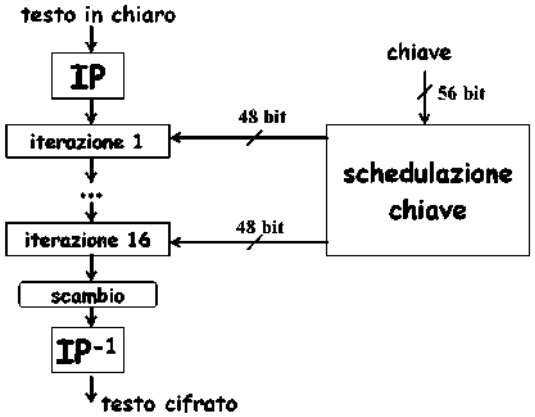


In ogni fase viene effettuata una sostituzione:la metà di destra del blocco(R_{i-1}) rappresenta la metà di sinistra della fase successiva(L_i).
La metà di destra(R_{i-1}) inoltre entra in una funzione f (detta funzione di fase)che dipende sostanzialmente dalla sottochiave K_i e l'output entra in un OR con la metà di sinistra(L_{i-1}) dando in output un blocco che rappresenta la parte di destra(R_i) della successiva fase.
Le fasi hanno tutte la precedente struttura.
Si può notare che tale algoritmo dipende dalle seguenti caratteristiche:
► **Dimensione del blocco:** maggiore è la dimensione del blocco maggiore sarà la sicurezza: per convenzione il blocco era di dimensioni di 64 bit anche se attualmente è di 128 bit(AES).
►**Dimensione della chiave:** maggiore è la dimensione della chiave maggiore sarà la sicurezza. Ovviamente più grande è la chiave maggiore sarà il tempo di esecuzione dell'algoritmo. Sempre per convenzione si utilizza una chiave di 128 bit.
►**Numero di fasi**: anche questo assicura una sicurezza maggiore, il cifrario di Feistel utilizza 16 fasi che sono più che sicure
►**Algoritmo di generazione delle sottochiavi:** tale algoritmo dovrebbe permette di evitare l'analisi crittografica
►**Funzione di fase**
Per la decifratura l'algoritmo è lo stesso non viene invertito ne sostituito, lo stesso vale per la funzione di fase, ma vengono invertite le sottochiavi. Cioè alla prima iterazione viene usata la chiave K_t, alla seconda K_{t-1} e così via.

DES(Data Encryption Standard)

E' l'algoritmo di cifratura simmetrico più utilizzato fino a poco tempo fa.
Tale algoritmo (chiamato anche DEA) prende in input blocchi di 64 bit e una chiave di 56 bit e dà in output blocchi di 64 bit.
E' stato dimostrato che fino a pochi anni fa era un algoritmo molto sicuro anche se vi furono due critiche una dovuta alla chiave di 56 bit che nella prima pubblicazione era di 128 bit (quindi erano stati tolti senza un motivo valido 72 bit) , e l'altra dovuta alla costruzione poco chiare delle strutture interne del DES(le S-Box).

STRUTTURA DEL DES



In input l'algoritmo prende un testo in chiaro(64bit) e la chiave(56 bit).
Il testo in chiaro viene trasformato utilizzando lo schema a sinistra mentre la chiave entra in un modulo detto di schedulazione della chiave rappresentato a destra dello schema.

Per quanto riguarda il testo in chiaro ad esso viene applicata una permutazione iniziale che permuta l'input e successivamente tale input entra in 16 fasi (in cui viene applicata una funzione).
Infine il blocco ottenuto dalla 16-esima fase viene suddiviso in due blocchi che vengono scambiati e successivamente viene applicata una permutazione (inversa rispetto alla permutazione iniziale).

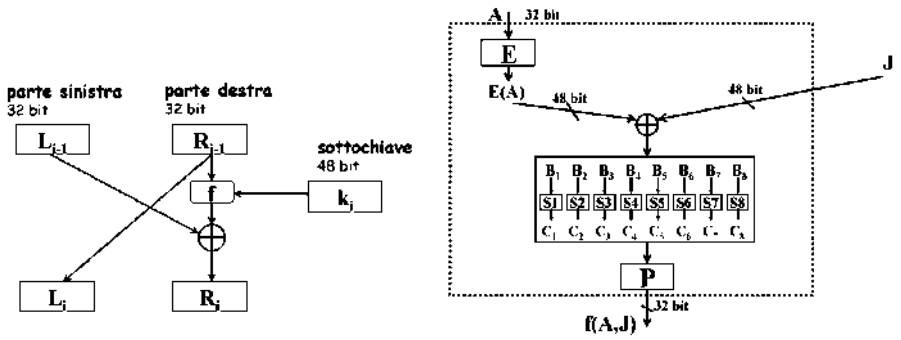
- **PERMUTAZIONE INIZIALE E FINALE**

Tali permutazioni sono applicabili grazie all'utilizzo di tabelle.
In queste tabelle sono presenti 64 celle in cui ogni cella rappresenta un bit.
Nella prima cella sarà scritto il bit che rappresenterà il primo bit dell'input permutato,nella seconda cella il secondo bit e così via.
Ovviamente lo stesso vale per la permutazione finale in cui la tabella è stata invertita.

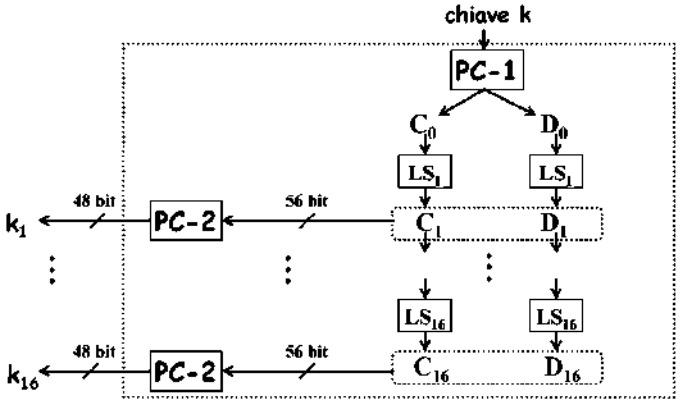
- **SINGOLA FASE**

In ogni fase si ha in input un blocco di 32 bit e una sottochiave di 48 bit.
Quindi la prima operazione da effettuare è l'espansione dei 32 bit del blocco a 48 bit.
L'ESPANSIONE consiste nel duplicare 16 bit utilizzando sempre una tabella di permutazione
32 1 2 3 4 5
4 5 6 7 8 9
8 9 10 11 12 13 e così via
Una volta effettuata tale espansione si ha un blocco di 48 bit che entrano in un XOR con i 48 bit della chiave.
Da tale XOR si hanno 48 bit in output che entrano in un modulo di SOSTITUZIONE.
La SOSTITUZIONE prevede l'utilizzo di 8 S-BOX.
I 48 bit in input alla SOSTITUZIONE vengono divisi in 8 blocchi di 6 bit ciascuno(ogni blocco entra in una S-BOX).
Per quanto riguarda la S-BOX essa è una tabella composta da 4 righe e 16 colonne.
L'input della SBOX è di 6 bit di cui il primo e l'ultimo vengono utilizzati per conoscere la righe da considerare mentre gli altri 4 bit vanno a rappresentare la colona.
L'intersezione tra riga e colonna darà il valore di una cella che conterrà un numero compreso tra 0 e 15.
Una volta ottenuto tale numero viene convertito in binario (al massimo è composto da 4 bit pari a 1) e tale output sarà di 4 bit.
Questo per tutti e 48 i bit quindi l'output del modulo di SOSTITUZIONE sarà di 32 bit.
Tali 32 bit entreranno in una permutazione finale che dipende sempre da una tabella conosciuta.

IN DEFINITIVA LE FASI SONO
- PERMUTAZIONE INIZIALE
- DIVISIONE DEL BLOCCO IN DUE PARTI(DESTRA E SINISTRA DI 32 BIT CIASCUNO)
- ESPANSIONE DELLA PARTE DI DESTRA DA 32 A 48 BIT
- XOR DEL NUOVO BLOCCO DI DESTRA DI 48 BIT CON CHIAVE DI 48 BIT
- SOSTITUZIONE (8 SBOX)
- PERMUTAZIONE
- PERMUTAZIONE FINALE INVERSA DI QUELLA INIZIALE.



Nello XOR viene utilizzata in una generica fase i la sottochiave Ki.
Tale sottochiave viene generata utilizzando il modulo di SCHEDULAZIONE DELLA CHIAVE.



Tale modulo prende in input una chiave di 64 bit(8 bit di parità aggiunti ai 48 bit della chiave) a cui dapprima viene effettuata una PERMUTAZIONE INIZIALE in cui i 64 bit vengono trasformati in 56 bit, utilizzando sempre una tabella conosciuta a priori ed eliminando un bit ogni 8 bit quindi verranno eliminati i bit in posizione 8 16 24 32 40 48 56 (64-8=56).
Questi 56 bit vengono suddivisi in due blocchi di 32 bit (denotati con C_i e D_i).
Sia per Ci che per Di viene effettuata in ciascuna fase uno shift di 1 o 2 posizioni a sinsitra a seconda di una tabella sempre conosciuta a tutti.
Infine i 56 bi entrano in una PERMUTAZIONE FINALE in cui viene sempre seguita una tabella conosciuta e vengono eliminati ogni 8 bit un bit in modo da ottenere un output di 48 bit che entrerà nello XOR.
IN DEFINITIVA LA SCHEDULAZIONE DELLA CHIAVE
- AI 64 BIT DELLA CHIAVE VIENE APPLICATA UNA PERMUTAZIONE INIZIALE
- I 56 BIT IN OUTPUT VENGONO DIVISI IN DUE BLOCCHI DI 32 BIT
- ENTRAMBI I BLOCCHI VENGONO SHIFTATI A SINISTRA DI UNA O DUE POSIZIONI
- I DUE BLOCCHI VENGONO UNITI
- I 56 BIT ENTRANO IN UNA PERMUTAIZONE FINALE OTTENENDO 48 BIT

DECIFRATURA IN DES

La decifratura utilizza la stessa tecnica del Cifrario di feistel che prevede che né l'algoritmo né tantomeno la funzione vengano invertiti ma vengono invertite solo le sottochiavi.

PROBLEMI DEL DES

Il DES è stato un algoritmo di cifratura molto potente ma vi erano due problemi riguardo esso:

- *LUNGHEZZA DELLA CHIAVE*: molti dubitarono della dimensione della chiave che da 128 bit è scesa a 56. Ma è dimostrato che con 56 bit il numero totale di chiavi è di 2^{56} che con una macchina che svolge una crittografia al millisecondo si avrebbero dei risultati dopo più di un migliaio di anni.
- *NATURA DELL'ALGORITMO*: molti dubitarono nella natura del DES in quanto il criterio di costruzione delle SBOX non era stato reso pubblico quindi per molti esse erano state create per poter individuare il testo in chiaro pur non conoscendo la chiave.

POTENZA DEL DES

E' dimostrato che una delle caratteristiche del DES è che riesce ad ottenere il cosiddetto Availance Effect ovvero una piccola variazione del testo in chiaro produce una grande variazione del testo cifrato.

MODALITA' OPERATIVE DEL DES

Il DES permette di cifrare testi di dimensioni multiple di 64/128 bit ma se il testo da cifrare non rientra in queste categorie vengono utilizzate le seguenti modalità:
- *ECB* (Eletronic codebook chaining)
- *CBC* (Cipher block chaining)
- *CFB* (Cipher feedback)
- *OFB* (Output feedback)
- *CTR* (Counter)

ELETRONIC CODEBOOK CHAINING ECB

E' la modalità più semplice che consiste nello scomporre il messaggio M originale in n blocchi da 64 bit ciascuno e di effettuare una cifratura DES per ognuno di essi.

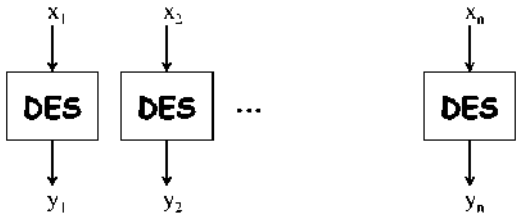
VANTAGGI

- I blocchi sono cifrati in parallelo(più veloce)
- Se vi sono blocchi più piccoli di 64 viene fatto un padding con 100000 (restanti posizioni).
- I blocchi sono indipendenti quindi gli errori non si propagano

SVANTAGGI

- Non vi è dipendenza tra i blocchi essi possono essere sostituiti senza che l'algoritmo se ne renda conto
- Se vi sono messaggi che si ripetono anche la cifratura si ripete

NOTA BENE
Per convenzione quando si incontrano un 1 seguito da tutti 0 significa che il testo da inviare è terminato, quindi anche se i blocchi sono tutti di dimensione 64 bit allora verrà utilizzato un ulteriore blocco contenente tutti 1 e 64 bit pari a 0.

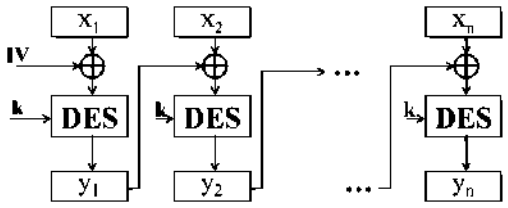


CIPHER BLOCK CHAINING CBC

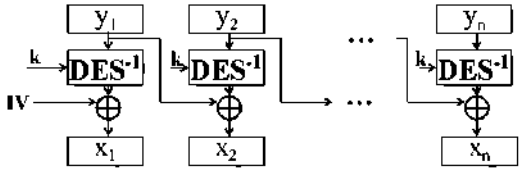
La modalità è simile alla precedente cioè il testo in chiaro viene scomposto in blocchi di 64 bit ciascuno. Alla prima iterazione il primo blocco entra in uno XOR con un vettore di inizializzazione IV (vettore conosciuto e che non influenza la sicurezza del sistema) e l'output entra in un modulo DES.

Nella seconda iterazione il secondo blocco di testo in chiaro entra in uno XOR con il precedente blocco cifrato e l'output entra in un modulo DES e così via per tutti i blocchi.
Nella decifratura il procedimento è lo stesso viene invertito solo lo XOR (ma l'inverso di uno XOR è lo XOR stesso).
Quindi ad ogni passo nello XOR entrerà il blocco di testo cifrato e l'output del successivo blocco di testo cifrato elaborato dal DES (invertito).

CIFRATURA :



DECIFRATURA :



VANTAGGI

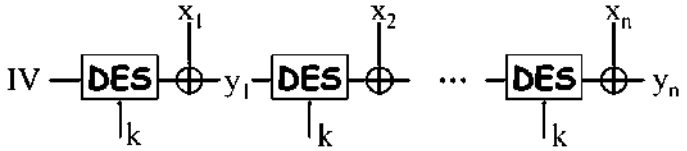
- La decifratura è parallelizzabile in quanto si usa solo il blocco precedente cifrato
- Se il vettore di inizializzazione è errato solo la cifratura del primo blocco è errata
- Vi è dipendenza tra blocchi quindi ci rendiamo conto degli errori

SVANTAGGI

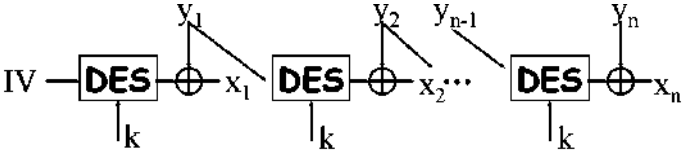
-Meno veloce di ECB
- Vi è propagazione di errori

CIPHER FEEDBACK (CFB)

Anche in questa modalità bisogna suddividere il testo in chiaro in blocchi di 64 bit.
La cifratura è sequenziale.
Al primo passo il vettore di inizializzazione entra in un modulo DES con la chiave e l'output entrerà in uno XOR con il primo blocco del testo in chiaro, tale output entrerà in un modulo DES con la seconda sottochiave e l'output entrerà in uno XOR con il secondo blocco di testo in chiaro e così via.



Nella decifratura il vettore di inizializzazione entra in un modulo DES l'output entra in uno XOR con il testo cifrato e viene dato in output il testo in chiaro,inoltre il blocco di testo cifrato entra nel passo successivo nel modulo DES per dare in output un blocco che entrerà nello XOR con l'altro blocco cifrato per dare in output il secondo blocco di testo in chiaro e così via.

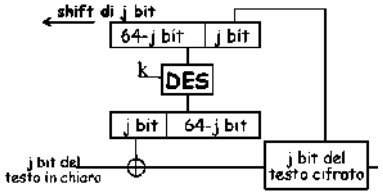


VANTAGGI

- La decifratura è parallelizzabile
- Se nella decifratura IV è errato è errato solo il primo blocco

ALTERNATIVA -----> j-bit CFB

Tale alternativa viene utilizzata in quanto i blocchi di tali modalità sono di 64 bit dato che cifrare una somma del genere impiega molto tempo si è pensati di decidere a priori una certa dimensione j e di iniziare ad effettuare le computazioni su essa.
Ovvero si inizia cifrando un blocco di 64 bit quindi questi entreranno nel DES ma nello XOR entreranno solo i primi j che poi continueranno l'iterazione finchè non verranno cifrati del tutto e spediti al destinatario che può cominciare a decifrarli.
Inoltre questi j bit vengono riposizionati nel blocco iniziale in quanto tenendo presente che nella modalità CFB classica il blocco cifrato ad ogni iterazione serve per cifrare il blocco successivo.



Per la decifratura è lo stesso discorso.

VANTAGGI

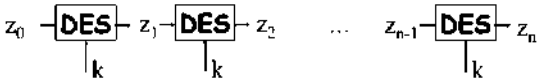
- Valore di j scelto a seconda delle esigenze
- Miglioramento perchè può essere effettuata una cifratura/decifratura senza dover aspettare i 64 bit

SVANTAGGI

- Più piccolo è j più lento è l'algoritmo

OUTPUT FEEDBACK (OFB)

Il messaggio viene scomposto in blocchi di 64 bit.
A partire dal vettore di inizializzazione IV viene creata una sequenza z (indipendente dal messaggio).
Per creare tale sequenza si usano le seguenti regole:
- Z_0= IV
-Z_1= Z_0 DES_k



In seguito per ottenere il testo cifrato si fa uno XOR tra il blocco e la corrispettiva sequenza Z.

$$y_i = x_i \oplus z_i$$

Anche in questo caso si può utilizzare la tecnica dei j-bit.

VANTAGGI

- XOR veloci perchè precomputati

SVANTAGGI

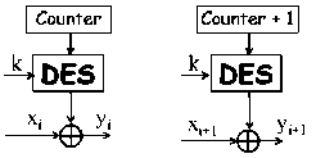
- - Non vi è dipendenza tra blocchi

COUNTER (CTR)

In tale modalità viene utilizzato un contatore (scelto da chi cifra).
Tale contatore entra nel DES e l'output entra in uno XOR con il blocco di testo in chiaro e il loro output rappresenta il primo blocco di testo cifrato,all'iterazione successiva il contatore viene incrementato e si riapplica la procedura.

VANTAGGI

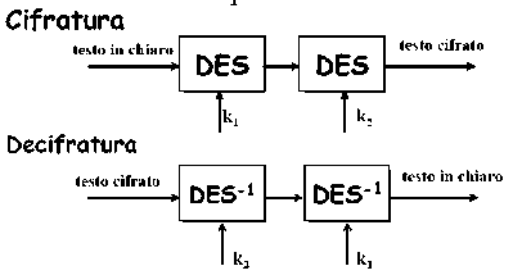
-Accesso casuale perchè posso decifrare un qualunque blocco semplicemente incrementando il contatore.



Successivamente sono stati progettati degli algoritmi basati sul DES che cercavano di migliorarne la Sicurezza.

DES DOPPIO

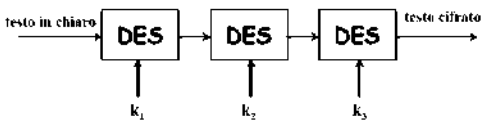
Una possibile soluzione è utilizzare un DES DOPPIO.
Ovvero suddividere sempre il testo da inviare in blocchi di 64 bit ciascuno e utilizzare due chiavi K1 e K2 di 56 bit ciascuno.
Quindi cifrare una prima volta i 64 bit del testo in chiaro utilizzando la prima chiave K1 e l'output rappresenterà l'input del nuovo modulo DES che prenderà la seconda chiave K2.



Un possibile attacco a tale cifrario è del tipo Known Plaintext Cipher (conosco delle parti di testo in chiaro).
Tale attacco viene definito anche come Meet in the Middle
Supponendo di conoscere una coppia X,Y (X = testo in chiaro, Y = testo cifrato) è possibile effettuare tutte le 2^{56} cifrature per k_1 ottenendo 2^{56} valori cifrati per X.
Inoltre eseguo tutte le 2^{56} decifrature per k_2 per Y inserisco i valori nella tabella e cerco le corrispondenze.
SPAZIO 2^{56} per inserire i valori cifrati di X
TEMPO : 2^{56} per cifrare X (e costruire la tabella), 2^{56} per decifrare Y, tempo per cercare le corrispondenze che se uso esce mi permette di impiegare tempo 1 altrimenti almeno 56 se è un array ordinato.
Quindi con tale attacco al più impiego 2^{56} e l'attacco di forza bruta è di 2^{112} quindi equivalente al cifrario con chiave 56 bit

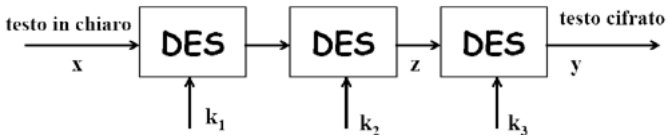
DES TRIPLICATO

Viene attuata la stessa tecnica del DES DOPPIO solo 3 volte e con 3 chiavi differenti.



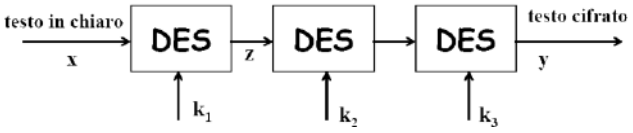
Anche in questo chiaro è attuabile un attacco Meet in the Middle che può essere effettuato in due modalità:

1.



Prendiamo il testo in chiaro X e lo cifriamo prima una volta con le 2^{56} possibili K_1 e poi l'output viene cifrato con 2^{56} possibili K_2,ne costruiamo la tabella ed effettuiamo la decifratura del testo cifrato Y con 2^{56} possibili k_3.
SPAZIO 2^{112}
TEMPO 2^{112} per ottenere Z, 2^{56} per decifrare y,2^{56} per trovare l'associazione

2.



Prendiamo il testo in chiaro X lo cifriamo in DES con le 2^{56} possibili K_1 poi decifriamo il testo cifrato Y con le 2^{56} possibili K_3 e l'output viene decifrato con le 2^{56} possibili k_2 e ricerchiamo le associazioni.
SPAZIO 2^{56}
TEMPO 2^{56} per cifrare x,2^{112} per decifrare y,2^{56} per cercare le associazioni
E' quindi dimostrato che è equivalente ad un cifrario che prende una chiave di 2^{112} bit e non 2^{168}

DES TRIPLO

Viene utilizzato l'algoritmo DES tre volte.
La prima e l'ultima viene attuata una cifratura utilizzando una chiave K,mentre la seconda volta viene applicato un DES-1 utilizzando una chiave K'.
Quindi la dimensione totale della chiave dovrebbe essere di 112 bit

AES (Advanced Encryption Standard)

E' uno standard ideato nel 2001 successivo al DES (nei prossimi anni lo rimpiazzerà del tutto).
Progettato da Rinijdael su proposta del NIST che aveva indotto una challenge per la scrittura di un algoritmo di cifratura che non avesse le stesse problematiche del DES (quindi problemi relativi alla chiave o progettazioni non spiegate).
L'AES a differenze di molti altri algoritmi simmetrici a blocchi non utilizza la struttura di Feistel ma possiede una struttura molto più complessa.

REQUISITI

L'unità elementare per l'AES è il byte che sono rappresentati in notazione esadecimale.
Ciascun byte viene rappresentato come un elemento del campo finito $GF(2^8)$

$\{b_7b_6b_5b_4b_3b_2b_1b_0\} \rightarrow b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x^1+b_0$

$\{11010100\} \rightarrow x^7+x^6+x^4+x^2$

ADDIZIONE: Somma dei coefficienti dei polinomi in modulo 2

MOLTIPLICAZIONE: Viene effettuata prima la moltiplicazione di polinomi e poi effettuato il modulo con un polinomio irriducibile(cioè divisibile solo per 1 e se stesso) di grado minore o uguale a 8

VEDI SLIDE AES PER LE VARIE OPERAZIONI(da 22 a 39)

Le prime due cose da dire riguardano la dimensione del blocco e della chiave:
- **BLOCCO** 128 bit
- **CHIAVE** 128 192 256 (senza vincoli)

(PER L'ESEMPIO USIAMO 128 BIT)
In base alla dimensione della chiave il numero di iterazioni dell'algoritmo varia secondo la seguente tabella:

Nr	Grandezza chiave in bit
10	128
12	192
14	256

RAPPRESENTAZIONE DELLA CHIAVE

La chiave a seconda dell'implementazione può essere di 128,192 o 256 bit.
Essa viene rappresentata come una matrice quadrata con 4 righe e il numero di colonne che dipende dalla dimensione presa. (Di solito DIMENSIONE CHIAVE/ 32)
Per il nostro esempio avremo 4 colonne.

$K_{0,0}$	$K_{0,1}$	$K_{0,2}$	$K_{0,3}$
$K_{1,0}$	$K_{1,1}$	$K_{1,2}$	$K_{1,3}$
$K_{2,0}$	$K_{2,1}$	$K_{2,2}$	$K_{2,3}$
$K_{3,0}$	$K_{3,1}$	$K_{3,2}$	$K_{3,3}$

RAPPRESENTAZIONE DEL BLOCCO

Il blocco è di 128 bit quindi 16 byte.
Anch'esso è rappresentato come una matrice quadrata 4x4(in cui in ogni cella c'è 1 byte)

in_0	in_4	in_8	in_{12}
in_1	in_5	in_9	in_{13}
in_2	in_6	in_{10}	in_{14}
in_3	in_7	in_{11}	in_{15}

ELEMENTO DI LAVORO

Le iterazioni vengono effettuate su una matrice quadrata 4x4 detta State

$S_{r,c}$ byte in riga r e colonna c

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

La prima computazione effettuata riguarda il copiare la matrice in input (il blocco) nella matrice State seguendo il seguente schema

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

in_0	in_4	in_8	in_{12}
in_1	in_5	in_9	in_{13}
in_2	in_6	in_{10}	in_{14}
in_3	in_7	in_{11}	in_{15}

=

$S_{0,0}=in_0$	$S_{0,1}=in_4$	$S_{0,2}=in_8$	$S_{0,3}=in_{12}$
$S_{1,0}=in_1$	$S_{1,1}=in_5$	$S_{1,2}=in_9$	$S_{1,3}=in_{13}$
$S_{2,0}=in_2$	$S_{2,1}=in_6$	$S_{2,2}=in_{10}$	$S_{2,3}=in_{14}$
$S_{3,0}=in_3$	$S_{3,1}=in_7$	$S_{3,2}=in_{11}$	$S_{3,3}=in_{15}$

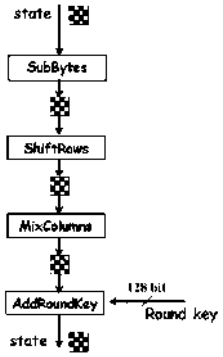
Una volta effettuato un AddRoundKey iniziale(identico a quello finale).

AddRoundKey

E' uno XOR bit a bit tra il blocco e una porzione della chiave estesa.

L'AES si basa su dei round.

La struttura di un singolo **ROUND** è la seguente



SubBytes

La matrice STATE entra nel modulo SubBytes che effettua una trasformazione utilizzando una S-box

Tale S-box è una matrice 16x16 che contiene una permutazione di tutti i 256 valori a 8 bit.

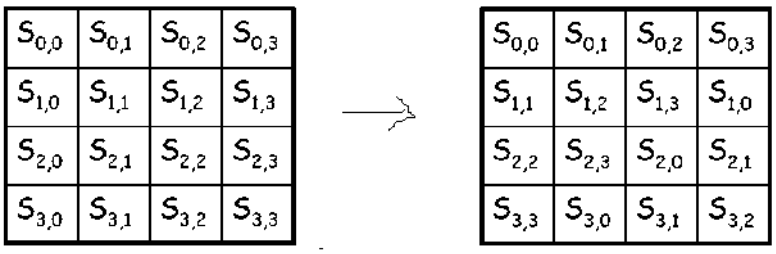
x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Nello STATE in ogni cella vi è 1 byte(8 bit rappresentati in forma decimale) i primi 4 rappresentano la riga(x) gli altri 4 rappresentano la colonna (y).

Prenderemo la riga della S-box associata alla X e la colonna associata alla Y.

Il valore contenente nella cella ottenuta dall'intersezione tra tale riga e tale colonna rappresenta il nuovo valore.
(Per la trasformazione inversa basta invertire la matrice)

ShiftRows
Consiste nell'effettuare uno shift ciclico a sinistra sulle ultime tre righe dello State.
Quindi la seconda riga è shiftata di una posizione,la terza di due e la quarta di tre.



MixColumns
Viene effettuata una moltiplicazione sulle colonne.
In particolare tale moltiplicazione è effettuata in modulo (x^4+1) utilizzando il polinomio
$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$

AddRoundKey
Viene effettuato uno XOR tra la chiave e lo state.
Dato che lo state è una matrice di byte e la chiave è una matrice di word.
Viene presa una colonna dello STATE ed effettuato uno XOR con una word della key (e così via).

ESPANSIONE DELLA CHIAVE
In input all'AES arriva una chiave di 128 bit(16 byte) ovvero 4 word.
Questa matrice di 4 word entra nel modulo di espansione della chiave per poter generare le sottochiavi da utilizzare in ogni round e nell'AddRoundKey iniziale.
La chiave viene copiata nelle prime 4 word della chiave espansa.
Il resto viene riempito 4 word per volta.
Ogni word w[i] dipende dalla word precendente w[i-1] e dalla word che si trova 4 posizioni indietro w[i-4].
Per ogni word in una posizione multiplo di 4 viene utilizzata una funzione f.
Tale funzione f è costituita da:
1. **RotWord**: shift a sinistra di un byte della word-> da $[b_0\ b_1\ b_2\ b_3]$ si passa a $[b_1\ b_2\ b_3\ b_0]$
2. **SubWord**: viene utilizzata la Sbox utilizzata nei round per effettuare le sostituzioni su ciascun byte della word
3. Viene effettuato uno XOR tra l'output precedente con una costante che non è altro che una word in cui i tre byte a destra sono sempre 0 l'altro è definito da una tabella conosciuta.
Per le word nelle restanti posizioni viene effettuato uno XOR sempre tra la word precedente e quella di 4 posizioni indietro.

STREAM CIPHER

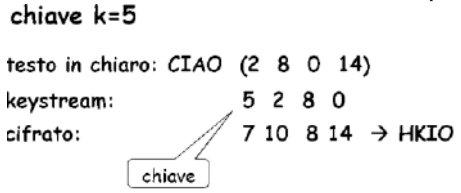
I cifrari simmetrici(a chiave unica) si suddividono in:
- cifrari a blocchi: in cui il testo iniziale viene scomposto in blocchi che vengono cifrati in maniera indipendente
- stream cipher(cifrari a flusso): in cui il testo viene cifrato un bit(o byte) per volta utilizzando delle sequenze pseudo-casuali.

Le caratteristiche degli Stream Cipher sono le seguenti:
- vanno a cifrare il testo un bit(o un byte) per volta
- sono molto più veloci dei cifrari a blocchi
- utilizzano come chiave una KeyStream che è una sequenza casuale ottenuta dalla chiave e dal messaggio
- la cifratura consiste nell'effettuare uno XOR tra il messaggio e la KeyStream

Gli StreamCipher più utilizzati sono:
- ***Cifario Autokey***
- ***RC4***

CIFRARIO AUTOKEY

Ad ogni lettera del testo in chiaro viene associato un numero (da 0 a 25) utilizzando la stessa associazione che si usava nel cifrario di Cesare.
La keystream è generata nel seguente modo:
$z_0=k$ $z_i=M_{i-1}$ (dove M è il messaggio e k la chiave)
Per effettuare la cifratura si fa uno XOR tra il testo in chiaro e la keystream (Z)



Tale cifrario non è sicuro in quanto ci sono al più 26 chiavi quindi un attacco di forza bruta (detto Known CipherText Attack) impiegheremmo un tempo minuscolo per essere attuato.

RC4

Tale algoritmo viene utilizzato nei protocolli SSL,TLS e in WEP e WPA.
Le operazioni vengono effettuate sui byte e la chiave è composta da un numero variabile di bit o byte.
La keystream generata viene ripetuta con una probabilità parti a 10^{100}
Viene utilizzato un vettore S da elaborare.

INIZIALIZZAZIONE DI S

S è un vettore di 256 byte in cui inizialmente andremo ad inserire un valore pari alla posizione del byte.
Quindi il primo byte avrà valore 1,il secondo 2 e così via.
Inoltre viene creato un vettore temporaneo T in cui :
- se K è uguale a 256 byte allora K viene copiato in T
- altrimenti se K è minore di 256 byte allora K viene ripetuto in T finchè non lo si riempe
- Si scambia S[i] con un S[j] che dipende da S[i] e dalla chiave k[i]

GENERAZIONE KEYSTREAM

A tal punto il vettore S viene utilizzato per generare la Keystream.
Viene calcolato un indice t=S[i]+S[j] e infine K=S[t]

Infine una volta generata la keystream viene effettuato lo XOR tra la keystream e il blocco del testo in chiaro

CRITTOGRAFIA ASIMMETRICA

La seconda tipologia di crittografia è quella asimmetrica *detta anche a chiave pubblica.*
In questa tipologia di crittografia vengono utilizzate due chiavi:
- una **privata** per cifrare il messaggio
- una **pubblica** per decifrare il messaggio (tale chiave è resa disponibile a tutti in un file pubblico)

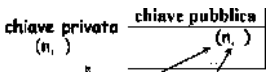
Chiunque voglia spedire un messaggio a qualche destinatario legge nel file pubblico la chiave pubblica associata a tale destinatario e cifra il messaggio utilizzandola. Il destinatario una volta ricevuto il messaggio utilizza una chiave privata per decifrarlo.
In questa tipologia di crittografia *non* vi sono problemi che riguardano la *SEGRETEZZA DELLA CHIAVE* in quanto mittente e destinatario non devono tenere segreta la chiave o tantomeno trovare un modo per scambiarsela in modo sicuro, semplicemente il mittente genera una coppia composta da chiave pubblica e privata utilizzando quella privata per decifrare e quella pubblica per decifrare la inserisce in un file pubblico utilizzato dal destinatario per cifrare il messaggio.

FUNZIONE ONE-WAY

Alla base di tutte le tecniche di cifratura asimmetrica vi è una funzione che gode di due grandi proprietà:
- è facile da calcolare
- è difficile da invertire (a meno di una informazione aggiuntiva)

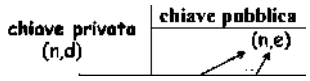
RSA (Rivest Shamir Adleman)

E' uno dei più famosi algoritmi di cifratura asimmetrica conosciuti e utilizzati.
Negli algoritmi asimmetrici è semplice capire che l'algoritmo di cifratura è banale ciò che diviene meno banale è trovare una coppia di chiavi(pubblica e privata) che sono in relazione tra di loro ma che sono difficili da calcolare se se ne conosce una delle due (nel caso nostro vogliamo che sia difficile risalire alla chiave privata pur conoscendo la chiave pubblica)
Lo schema delle chiavi in RSA è il seguente:
- viene scelto un numero dato dal prodotto di due numeri interi primi della stessa lunghezza (cioè divisibili per 1 e per se stessi) $n = pq$ p,q primi
- n sarà il primo valore inserito sia nella chiave pubblica che nella chiave privata



- a partire da p,q vengono generati due numeri e,q tenendo in considerazione la seguente regola
  . il massimo comune divisore tra il numero e e (p-1) (q-1) deve essere 1 (numeri relativamente primi)
  .una volta trovato e il prodotto ed=1 mod(p-1)(q-1)

$$\gcd(e, (p-1)(q-1))=1$$
$$ed = 1 \mod (p-1)(q-1)$$

- il numero e sarà il secondo valore della chiave pubblica mentre il numero d sarà il secondo valore della chiave privata 

Una volta generate le chiavi in tal modo si possono capire gli algoritmi di cifratura RSA e decifratura RSA.

CIFRATURA CON RSA

Il messaggio viene cifrato elevando M alla e il tutto in modulo n (dove n ed e sono letti nel file pubblico)

Cifratura di M

$$C \leftarrow M^e \bmod n$$

DECIFRATURA CON RSA

Il messaggio viene decifrato elevando C alla d in modulo n(dove n,d fanno parte della chiave privata)

Decifratura di C

$$M \leftarrow C^d \bmod n$$

DIMOSTRAZIONE CORRETTEZZA CIFRATURA/DECIFRATURA

$$\begin{aligned} C^d \bmod n &= (M^e)^d \bmod n \\ &= M^{ed} \bmod n \qquad ed = 1 \bmod (p-1)(q-1) \\ &= M^{1+k(p-1)(q-1)} \bmod n \\ &= M \cdot (M^{(p-1)(q-1)})^k \qquad \text{Teorema di Eulero: } M \in Z_n^* \Rightarrow M^{(p-1)(q-1)} = 1 \bmod n \\ &= M \bmod n \\ &= M \end{aligned}$$

poichè $0 \le M < n$

Per analizzare la sicurezza di RSA dobbiamo studiare quali sono i possibili attacchi su di esso .
Il caso banale è che l'attaccante conoscendo la chiave pubblica (n,e) vuole calcolare la chiave privata (n,d) n già lo conosce quindi deve solo calcolare d.
Per farlo basta elevare e^{-1} in modulo (q-1)(p-1)

$d = e^{-1} \bmod (p-1)(q-1)$ quindi

- dovrebbe fattorizzare n in modo da ottenere p,q
- dovrebbe calcolare (p-1) e (q-1)
- dovrebbe calcolare d come nella formula precedente

LAS VEAGAS (Algoritmo utilizzato per la fattorizzazione)

Questo algoritmo preso in input (n,e) calcola (p,q) con probabilità 1/2 oppure non dà risposte con probabilità 1/2 (ABBIAMO IL 50% DI POSSIBILITA')
A parte tale algoritmo tutte le altre soluzioni hanno una complessità sub-esponenziale se non esponenziale proprio quindi dal punto di vista computazionale non è fattibile effettuare un attacco di questo tipo.

ACCORDO TRA CHIAVI

DIFFIE-HELLMAN

E' una delle tecniche utilizzate per l'accordo su chiavi.
Esso si basa sull'intrattabilità del problema del logaritmo discreto.
Alla base di tale algoritmo vi è la ricerca di un numero intero p e di un generato g di Z_p^*

TROVARE p,g

Per trovare il numero intero p e il generatore g si seguono le seguenti regole:
- scegli a caso due numeri primi p_1,p_2
- $p=1+2p_1p_2$
- verifica se p è primo,se no torna al primo passo
- g=scegliGeneratore(p,(2,p_1,1,p_2,1))

ScegliGeneratore(p,(p_1,e_1,p_2,e_2....p_k,e_k)
-g= elemento a caso in Z_p^*
- se $g^{(p-1)/p_1}!=1$ mod p e $g^{(p-1)/p_k}!=1$ mod p allora g è generatore altrimenti torna al passo 1

Una volta trovati il primo p e il generatore g per trovare la chiave si seguono le seguenti regole:
- il mittente sceglie un numero x in Z_p
- il destinatario sceglie un numero y in Z_p
- il mittente calcola g^x mod p e lo invia al destinatario
- il destinatario calcola g^y mod p e lo invia al mittente
- il mittente calcola $g^{(xy)}$mod p in quanto conosce g^y e sa x quindi $g^{(xy)}$mod p=$g^{(y)x}$
- lo stesso vale per il destinatario
-il valore g(xy)mod p sarà la chiave

$$K = g^{xy} \bmod p$$
$$= (g^y)^x \bmod p$$

Come detto in precedenza la sicurezza di tale algoritmo si basa sull'intrattabilità del logaritmo discreto.

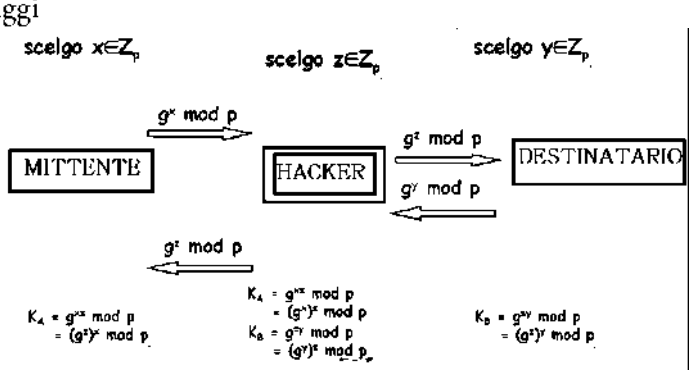
LOGARITMO DISCRETO

Dati tre numeri a,b,n calcolare x tale che a^x=b mod n
E' dimostrato che se n è primo (come p in Diffie Hellman) il tempo medio per tale ricerca è di 1.9222222229

ATTENZIONE

Tale algoritmo non resiste ad un attacco Man in The Middle.
Se da un lato vi è il mittente che sceglie un numero x inviando g^x mod p al destinatario e dall'altro lato vi è il destinatario che fa lo stesso con y non è sicuro che tra di essi si interpone una terza entità che

potrebbe aver scelto un numero z e calcolato g^z inviandola al mittente e al destinatario e tracciando direttamente i loro messaggi



CIFRATURA DH (DIFFIE HELLMAN)

Dall'algoritmo sull'accordo sulle chiavi si è pensato successivamente di utilizzare tale tecnica come cifrario asimmetrico in cui la chiave privata è composta da (p,g,a) mentre la chiave pubblica da (p,g,g^a mod p)

chiave privata	chiave pubblica
(p,g,a)	(p,g,g^a mod p)
	...

CIFRATURA

Per utilizzare la cifratura Diffie-Hellman sia mittente che destinatario scelgono due numeri (che nell'esempio indico con b per il mittente e a per il destinatario).
Il destinatario è colui che avrà come terzo valore della chiave privata il numero appena scelto (a) e inserirà nel file pubblico la chiave pubblica contenente come terza cifra g^a mod p
Quando il destinatario vuole inviare un messaggio sceglie un numero b e (sempre ricordando la tecnica sull'accordo sulle chiavi deve inviare g^b mod p al destinatario) per cui il primo input del testo che invierà sarà g^b mod p, successivamente deve applicare la chiave pubblica del destinatario che era g^a mod p quindi avrà come chiave $(g^a \bmod p)^b \bmod p$ utilizzerà tale chiave per ottenere il testo cifrato.

Cifratura di M
$g^b \bmod p$
$k \leftarrow (g^a \bmod p)^b \bmod p$
$C \leftarrow ENC(k, M)$

DECIFRATURA

Per la decifratura viene utilizzata la stessa tecnica.
Il destinatario possiede la a e conosce il valore g^b mod p per cui si calcola il valore della chiave che sarà $(g^b \bmod p)^a \bmod p$ e utilizza tale valore per decifrare il testo

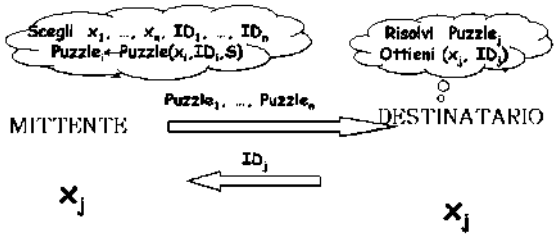
Decifratura di g^b mod p, C
$k \leftarrow (g^b \bmod p)^a \bmod p$
$M \leftarrow DEC(k, C)$

PUZZLE DÌ MERKLE

Oltre all'algoritmo DH un'ulteriore proposta per l'accordo sulle chiavi.
Il puzzle di Merkle a differenza dell'algoritmo DH (basato sul logaritmo discreto) non è basato su nessuna assunzione computazionale.

Si deve dire che tale algoritmo dal punto di vista implementativo non è realizzabile in quanto ci sono delle accortezze che lo rendono poco sicuro.
Il mittente genera n chiavi distinte e le inserisce in un puzzle.
La soluzione del solo puzzle ha un tempo ragionevole mentre cercare di risolvere tutti i puzzle richiede un tempo elevato.
Innanzitutto l'algoritmo per la creazione del Puzzle prende in input tre valori:
- **x** che è la soluzione del Puzzle
- **Id** che è l'identificativo del Puzzle
- **S** che è un valore noto

Il mittente genera n puzzle e li invia al destinatario che calcola uno solo dei n puzzle supponiamo che ad esempio trovi la soluzione x_j per far capire al mittente quale puzzle ha scelto gli invia l'ID_j del puzzle associato.



Queste operazioni impiegano il seguente tempo:
- la **costruzione** di n puzzle impiega TETA(n)
- la **risoluzione** di un solo puzzle impiega TETA(t)
- l'attaccante deve **risolvere al più n/2** puzzle quindi impiega TETA(nt)

Supponendo che n=TETA(t) allora l'ultima computazione è pari TETA(n^2)
Tale implementazione sembra essere stabile se però il mittente invia le tre informazioni associate al puzzle cifrandole con uno stesso algoritmo (ad esempio CBC-DES)

$$\text{Computa } y \leftarrow CBC\text{-}DES_k(x, ID, S)$$

Ma supponiamo che decide di inviare le informazioni associate al puzzle cifrando ognuna di essa con un algoritmo (ad esempio usa il DES) in tal caso quando il destinatario invia in chiaro l'id_j del puzzle l'attaccante lo legge e lo cifra con tutte le possibili chiavi.

$$\text{Computa } y \leftarrow DES_k(x), DES_k(ID), DES_k(S)$$

Dopo aver fatto ciò correla ogni cifratura trovata con i puzzle inviati inizialmente dal mittente trovando il puzzle con Id_j dato che conosce la chiave (perchè l'ha ricavata nella cifratura di Id_j) riesce a trovare la soluzione x del puzzle e quindi conosce la chiave che usano per scambiarsi i messaggi

FIRME DIGITALE

La tecnica di firme digitali è stata introdotta per fare in modo di poter riconoscere l'autenticità di chi inviava messaggi.
Gli algoritmi sono di crittografia asimmetrica.
L'idea è la seguente:
- un mittente quando vuole inviare un messaggio lo deve firmare
- il destinatario che riceve un messaggio verifica che la firma è associata all'entità corretta

Per capire la sicurezza della firma digitale devono essere studiati i:
- *TIPI DI ATTACCO*
- *SCOPI DELL'ATTACCO*

TIPI DI ATTACCO(cosa l'attaccante conosce)

I tipi di attacco sono:
- **Key Only Attack** : l'attaccante conosce solo la chiave pubblica del mittente
- **Known Message Attack** : l'attaccante conosce una serie di messaggi con le relative firme
- **Chosen Message Attack**: l'attaccante sceglie una serie di messaggi e chiede al mittente di firmarli

SCOPI DELL'ATTACCO(a cosa vuole arrivare l'attaccante)

Gli scopi degli attacchi possono essere delle seguenti tipologie:
- **Total break**: l'attacante vuol determinare kpriv del mittente per poter firmare qualsiasi messaggio
- **Selective forgery** : l'attacante vuole determinare dato un messaggio M la firma associata ad esso
- **Existential forgery**: l'attacante vuole determinare una coppia (M,F) messaggio firma che venga riconosciuta correttamente

RSA PER FIRME DIGITALI

L'algoritmo RSA di crittografia asimmetrica viene anche utilizzato per le firme digitali.
Il procedimento è lo stesso dell'algoritmo solo che non si applica un generico algoritmo di cifratura ma un algoritmo detto di firmatura (che indicheremo con FIRMA).
Quindi il mittente che vuole firmare un messaggio eleva il Messaggio alla d e lo invia al destinatario

Firma di M

$$F \leftarrow M^d \bmod n$$

Il destinatario per verificare che la firma è corretta eleva la firma alla e e deve ottenere il messaggio. (Se i messaggi combaciano allora la firma è quella corretta)

Verifica firma di M

vera se $M = F^e \bmod n$

falsa altrimenti

POSSIBILI ATTACCHI

1. **Selective Forgery KeyOnlyAttack**(l'attaccante sceglie i messaggi da far firmare conoscendone poi solo la chiave pubblica): tale attacco NON E' REALIZZABILE in quanto vogliamo calcolare M^d mod n il che è equivalente a rompere il crittosistema RSA (problema fattorizzazione)
2. **ExistentialForgery KeyOnlyAttack**(l'attaccante vuole determinare delle coppie (M,F) che vengano verificate correttamente e conosce solo la chiave pubblica: in tal caso può scegliere a caso una firma F elevarla alla e ed ottenere il messaggio M

1. **Scelgo F a caso**
2. $M \leftarrow F^e \bmod n$

3. **ExistentialForgery KnownMessageAttack** (l'attacante vuole individuare coppie(M,F) verificabili e conosce dei messaggi M_1 e M_2 con le relative firme F_1 F_2.

Proprietà di omomorfismo

$F_1 = M_1^d \bmod n \quad F_2 = M_2^d \bmod n$

$(F_1F_2)^e \bmod n = F_1^e F_2^e \bmod n = M_1M_2 \bmod n$

$F_1F_2 \bmod n$ è una firma valida per $M_1M_2 \bmod n$

4. **Selective Forgery Chosen Message Attack** (l'attaccante vuole determinare la firma di un messaggio M chiedendo al mittente di firmarlo): l'attaccante prende un messaggio $M=M_1M_2$ e chiede singolarmente al mittente di firmare M_1 e poi M_2 per la proprietà di Omomorfismo vista in precedenza F_1F_2 è una firma valida per M_1M_2

Se il messaggio da firmare è molto grande l'algoritmo risulta essere poco efficiente.
Per tale motivo vengono utilizzate le funzione Hash.

FUNZIONI HASH

Il valore Hash di un messaggio è una rappresentazione non ambigua e non falsificabile di un messaggio.
Le caratteristiche della funzione hash sono:
- comprime
- è efficiente
- sicurezza forte: è difficile trovare 2 messaggi diversi che hanno lo stesso valore hash
- one way: facile trovare il valore hash associato ad un messaggio ma è difficile trovare il messaggio a cui è associato il valore hash

Dato che una funzione hash comprime allora si prende il messaggio di grandi dimensioni e si calcola la funzione hash ottenendo un messaggio di dimensioni ridotte su cui si può invocare l'algoritmo di firma.

FIRMA RSA CON FUNZIONI HASH

Per effettuare una firma su un messaggio M ne calcolo il valore Hash h(m) e lo elevo alla d

Firma di M

$F \leftarrow [h(M)]^d \bmod n$

Per verificare la firma la elevo alla e ottenendo h(M)

Verifica firma di M

vera se $h(M) = F^e \bmod n$

falsa altrimenti

FIRMA CON ELGAMAL

Tale algoritmo si basa sull'intrattabilità del logaritmo discreto.
Alla base vi è il concetto di numero primo p e generatore g di Z_p^*

La **chiave privata** è composta da:
- p: numero primo
- g : generatore di Z_p^*
- s : numero casuale minore di p

chiave privata

(p,g,s)

La **chiave pubblica** è composta da:
- p: numero primo
- g: generatore di Z_p^*
- β: $g^s \bmod p$

chiave pubblica
(p,g,β)

FIRMA

Firma di M
$r \leftarrow$ a caso in Z_p^* con $gcd(r,p-1)=1$
$\gamma \leftarrow g^r \bmod p$
$\delta \leftarrow (M-s\gamma)r^{-1} \bmod p-1$
$firma_{(p,g,s)}(M,r) = (\gamma,\delta)$

VERIFICA

Verifica firma di M
vera se $g^M = \beta^\gamma \gamma^\delta \bmod p$
falsa altrimenti

DSS (Digital Signature Standard)
Anche tale approccio utilizza una funzione hash.
IDEA
Il codice hash generato viene fornito in input ad una funzione di firma insieme a un numero casuale k.
La firma dipende dalla chiave privata di chi deve firmare e da una chiave pubblica globale,l'output di tale funzione è composto da y e DELTA
Il destinatario genera il codice hash associato al messaggio e invia alla funzione di verifica tale codice e la firma.
La funzione di verifica dipende dalla chiave pubblica del mittente e dalla chiave pubblica globale.
L'output di tale funzione sarà un valore uguale a DELTA se la firma è valida.

chiave privata
(p,q,α,s)

file pubblico

utente	chiave pubblica
Alice	(p,q,α,β)
...	...

Firma di M
$r \leftarrow$ numero casuale in $[1,q-1]$
$\gamma \leftarrow (\alpha^r \bmod p) \bmod q$
$\delta \leftarrow (SHA(M)+s\gamma)r^{-1} \bmod q$
$firma_{(p,q,\alpha,s)}(M,r) = (\gamma,\delta)$

Verifica firma di M
$e' \leftarrow SHA(M)\delta^{-1} \bmod q$
$e'' \leftarrow \gamma\delta^{-1} \bmod q$
vera se $\gamma = (\alpha^{e'}\beta^{e''} \bmod p) \bmod q$
falsa altrimenti

FIRMA DIGITALE REMOTA (HSM)
L'idea è quella di inviare l'hash di un documento ad un modulo remoto che ne effettua la firma.
I moduli remoti sono gli HSM (Hardware Secure Module) ovvero delle macchine gestite da un certificatore accreditato.

LE FUNZIONI HASH

Una funzione HASH è una funzione che prende in input una lunghezza arbitraria di bit e da in output una lunghezza compressa di tali bit.

In particolare l'idea è che dato un messaggio il valore HASH ad esso associato è un valore non falsificabile e non ambiguo
Le funzioni HASH possono essere utilizzate in diversi ambiti:
- FIRMA DIGITALE(come visto in precedenza)

- INTEGRITA' DEI DATI: quando salvo un documento memorizzo il suo valore hash quando lo riapro calcolo il valore hash del documento aperto e vedo se tale valore è uguale a quello memorizzato

Una funzione HASH ha tre **caratteristiche**:
1. **SICUREZZA DEBOLE:** dato un messaggio M è difficile trovare un altro messaggio M' con lo stesso valore hash cioè tale che h(M)=h(M')
2. **SICUREZZA FORTE:** è difficile trovare due messaggi M e M' con lo stesso valore hash (l'hash deve essere almeno di 2^{160}
3. **ONE WAY**: facile da calcolare difficile da invertire

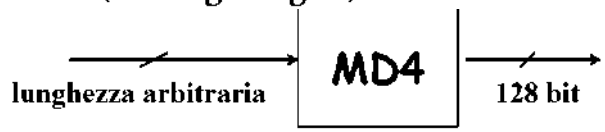
CALCOLO DEL NUMERO DI MESSAGGI DA INDIVIDUARE PER OTTENRE UNA COLLISIONE

- n: numero dei diversi valori hash
- t: numero dei messaggi da scegliere in X
- ε: probabilità di successo

$$t \approx \sqrt{n \cdot 2\ln\left(\frac{1}{1-\varepsilon}\right)}$$

$t=2^{80}$ con $n=2^{160}$

MD4 (Message Digest)



OBIETTIVI

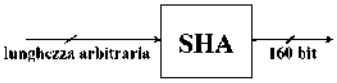
- Sicurezza forte
- Sicurezza diretta: cioè non basata su problemi computazionalmente difficili come ad esempio (RSA,DH)
- Velocità
- Semplicità: semplice da realizzare
- Compattezza: non vengono utilizzate tabelle o strutture dati complesse

L'unità elementare è un blocco di 512 bit quindi se arrivano blocchi di altre dimensioni viene effettuato un padding. Opera quindi su un numero di parole multiplo di 16.
Vi sono 3 round.
Ogni round consiste di 16 operazioni

MD5

Vale lo stesso discorso di MD4 sia per gli oviettivi che per le dimensioni dei blocco.
Vi sono 4 round in cui l'ultimo round contiene l'aggiunta di una nuova funzione.
Ogni round consiste in 16 operazioni

SHA (Secure Hash Algoritm)



SHA essendo un algoritmo basato su funzioni hash ha la caratteristicha di produrre un message digest di lunghezza fissa partendo da un messaggio di lunghezza variabile.

Vi sono differenti implementazioni di Sha:
- SHA-1 che produce un messaggio di 160 bit
- SHA-2 denotati con SHA-256 che produce messaggi di 256 bit
  SHA-512 che produce messaggi di 512 bit e così via.

Il più diffuso è lo SHA-1.

FUNZIONAMENTO

PASSO1: AGGIUNTA DI BIT DI RIEMPIMENTO finchè la lunghezza del messaggio è uguale a un numero più piccolo(di 64 bit)di un multiplo di 512 bit

PASSO2: AGGIUNTA DELLA LUNGHEZZA ovvero al messaggio ottenuto al passo precedente vengono aggiunti 64 bit che rappresentano la lunghezza del messaggio originale da ciò otteniamo una sequenza che è un multiplo di 512

PASSO3: INIZIALIZZAZIONE DEL BUFFER HASH viene utilizzato un buffer di 160 bit suddiviso in 5 registri da 32 bit (A B C D E)

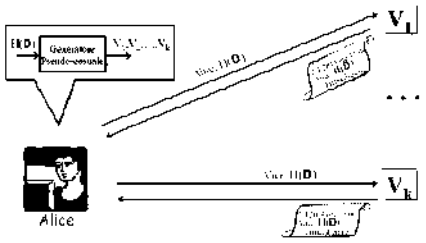
PASSO4: ELABORAZIONE DEI BLOCCHI: il messaggio ottenuto al passo 2 viene suddiviso in blocchi di 512 bit. Ognuno di questi blocchi passa in una funzione composta da 4 round di 20 operazioni che sono delle funzioni di logica primitiva. In questa funzione quindi entra il blocco,una costante k e i valori dei 5 registri. Al termine della computazione sono ottenuti 5 nuovi valori dei registri che verranno utilizzati per le computazioni successive.

MARCA TEMPORALE

La marca temporale è una prova che un documento sia stato prodotto
- prima
- dopo
- ad

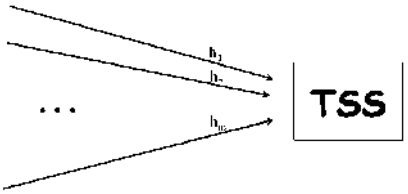
un determinato momento.
In Italia il CAD(Codice Amministrazione Digitale) predispone la seguente realizzazione:
Dato un documento il proprietario calcola il valore hash e lo invia ad un CA(certificatore accreditato).
Il problema è dove custodire la marca temporale?
Vi sono due possibili soluzioni:
1. Protocolli distribuiti: senza l'aiuto di un'autorità fidata quindi avere più testimonianze nel tempo
2. Protocolli con link: con autorità fidate collegate tra di loro

PROTOCOLLI DISTRIBUITI



Il livello di sicurezza è abbastanza forte in quanto con un numero k di testimonianze è difficile per Alice corrompere k persone.
Però vi sono dei problemi che possono insorgere:
- la vita di una firma potrebbe essere piccola e quindi non valere più dopo un determinato periodo di tempo

PROTOCOLLI CON LINK



Si parla di Time Stamping Service ovvero viene utilizzata un'autorità fidata che riceve delle richieste in intervalli di tempo predefiniti le collega tra di loro ed invia ad ognuno di essi una marca temporale

SISTEMI DI AUTENTICAZIONE

Per utilizzare un qualsiasi servizio di solito un utente deve autenticarsi.
Possiamo vedere un sistema di autenticazione come:
- un qualcosa che l'utente possiede (carte di identità,chiavi)
- un qualcosa che l'utente conosce (pin,password)
- un qualcosa che l'utente è (biometria)

Uno dei problemi più ricorrenti è il cosiddetto problema delle password.
Le password devono essere memorizzate in un sistema che deve in qualche modo rappresentarle.
Una possibile idea è applicare la funzione hash sulla password da memorizzare.
Gli attacchi alle password possono essere:
- spiare durante la digitazione (anche da lontano con telecamere super potenti)
- tramite intercettazioni
- tentare a caso o con programmi sistematici(attacchi con dizionario)

ATTACCO CON DIZIONARIO
Si basa sulla possibilità che un utente possa usare come password una frase di senso compiuto, quindi si tenta di inserire delle password che corrispondono a delle parole presenti in un dizionario.

SISTEMI DI AUTENTICAZIONE

ONE TIME PASSWORD: in questa tecnica ogni password è usata una sola volta
Le implementazioni sono:
- **LISTA CONDIVISA**: vi è un sistema che utilizza la lista di tutte le possibili password che possono essere usate una sola volta e poi vengono cancellate
- **SCHEMA DI LAMPORT** : che consiste in Alice(che deve autenticarsi) che prende un messaggio e ne calcola la funzione hash per un certo numero di volte t-i(il massimo numero di volte in cui può farlo è pari a t). Calcolato tale valore lo invia al sistema che ne controlla la correttezza. Quindi Alice all'iterazione successiva dovrà calcolare tale funzione hash t-i-1 volte
- **HMAC su OTP** (HOTP): Alice invia al sistema di autenticazione un valore HOTP(K,C) dove K è la chiave C è il contatore che entrano in un modulo HMAC-SHA1 in cui vengono troncati dei byte e impostati i bit più significativi a 0
- **TOTP** (Time Based su OTP): il modulo prende in input una chiave K e un tempo T, converte il tempo in un contatore C e inviare la chiave K e tale contatore C al modulo HOTP(visto nel punto precedente)

$$C = \frac{\text{tempo corrente - tempo inizio}}{\text{time step}}$$

Tempo inizio concordato
Time step: default 30 secondi

Il contatore è calcolato con la seguente formula
Questa tipologia di sistema ha il problema relativo ai ritardi degli input (o ai ritardi di ricezione del tempo tra i vari sistemi)

MAC (Message Autentication Code)

Viene inviato un messaggio M con una chiave K ad un modulo MAC e viene restituito un valore MAC(M,K) di n bit.
NOTA BENE: Il MAC garantisce solo autenticità del messaggio e integrità

TIPI DI ATTACCHI

- ***Known Message Attack***: l'attaccante conosce una lista di messaggi con i relativi MAC
-***Chosen Message Attack***:l'attaccante conosce i messaggi e chiede di calcolare i MAC
-***Adaptive Chosen Message Attack***: l'attaccante conosce i messaggi e chiede di calcolarne i MAC basandosi sulle riposte precedenti

SCOPI DELL'ATTACCO

- **Total break**: attaccare per determinare la chiave
-**Selective forgery**: dato un messaggio M determinare se esiste y tale che y=MAC(M,K)
-**Existential forgery**: determinare una coppia M,y tale che y=MAX(M,K)

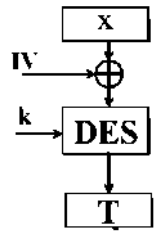
MAC: COSTRUZIONI

Vi sono due tipologie di costruzioni:
- basate su cifrari a blocchi (**DAA**)

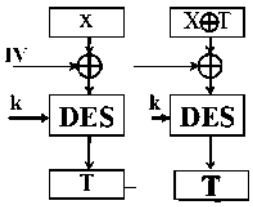
- basate su funzioni HASH

MAC su CIFRARI A BLOCCHI (DAA)

E' basato sul classico cifrario a blocchi CBC.
Vi sono dei problemi di sicurezza:
T è il MAC di X



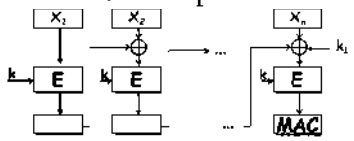
T è anche il MAC di X || X(XOR)T



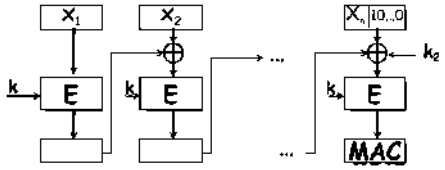
Quindi l'avversario conoscerebbe subito il codice MAC di due blocchi (X e X||X(XOR)T) perchè vale sempre T
Il miglioramento è stato attuato introducendo l'utilizzo di tre chiavi (K,e due sottochiavi K_1 e K_2) ed è identificato con CMAC

CMAC

Tale algoritmo viene implementato con l'utilizzo di AES o del Triplo DES.
Il testo viene suddiviso in blocchi della stessa dimensione.
Abbiamo due possibilità:
- la lunghezza del messaggio è **MULTIPLO** della lunghezza dei blocchi e in tal caso viene utilizzata la chiave K_1 che dipende solo dalla chiave K e dal blocco b



- la lunghezza del messaggio **NON E' MULTIPLO** della lunghezza dei blocchi in tal caso viene utilizzata la chiave K_2 che dipende solo dalla chiave K e dal blocco b. In tal caso verranno **aggiunti dei bit di riempimento** alla destra dell'ultimo blocco: per convenzione il primo di essi sarà uguale a 1 e gli altri saranno tutti 0



GENERAZIONE SOTTOCHIAVI

Per generare le sottochiavi K_1 e K_2 si applica la cifratura al blocco costituito unicamente da bit nulli.
- K_1 è ottenuta cifrando il testo ed effettuando uno scorrimento a sinistra di un bit e successivamente uno XOR con una costante che dipende dalla dimensione del blocco
- K_2 è derivata allo stesso modo dalla chiave K_1

MAC su FUNZIONI HASH

HMAC

Gli ***obiettivi*** che hanno portato alla nascita di tale tecnica sono i seguenti:
- utilizzare le funzioni hash disponibili senza modifiche
- sostituire le funzioni hash disponibili con altre funzioni più veloci ed efficienti
- Garantire le stesse caratteristiche di una funzione hash (sicurezza debole,forte,one way)
- Utilizzare e gestire le chiavi in modo semplice

COMPONENTI DI HMAC

Le componenti di HMAC sono:
- **H**: una funzione Hash che utilizza un vettore di inizializzazione (quindi MD5,SHA-1 ecc)
- **n**: lunghezza del valore hash(128,160 bit)
- **M**: messaggio che viene suddiviso in L blocchi b di dimensione maggiore di n (b>n)
- **ipad**: byte ripetuto b/8 volte (byte= rappresentazione esadecimale di 36)
- **opad**: byte ripetuto b/8 volte (byte= rappresentazione esadecimale di 5C)
- **K** : chiave segreta che > Se ha una lunghezza più piccola di b allora ne viene fatto il padding con zeri
  > Se ha una lunghezza più grande di b viene utilizzata una funzione HASH(per comprimere e poi viene fatto il padding con zeri

FUNZIONAMENTO

1. opad XOR K
2. ipad XOR K
3. H(2,M)
3. H(1,3)

In alcuni casi avviene un troncamento cioè vengono utilizzati solo i primi bit del HASH.
Abbiamo identificato con H la funzione HASH. Supponendo che la funzione HASH utilizzata sia SHA-1 chiameremo l'algoritmo HMAC-SHA-1 ed ad esempio HMAC-SHA-1-80 per indicare che si utilizzano solo i primi 80 bit.
La sicurezza di tale algoritmo ovviamente dipende dalla funzione hash utilizzata.
Il miglior attacco conosciuto è basato sul paradosso del compleanno

$$\text{Occorrono } 2^{|\text{hash}(\cdot)|/2} \text{ coppie } (M, HMAC_K(M))$$

DISTRIBUZIONE CHIAVI PUBBLICHE

Quando abbiamo trattato gli algoritmi di crittografia asimmetrica abbiamo detto che l'entità si occupava di generare una coppia chiave pubblica/privata e che la chiave pubblica era inserita in un file pubblico.
Per file pubblico intendiamo diverse possibilità:
- **Annuncio pubblico**: scambio diretto con la controparte, tramite una pubblicazione su un sito web, tramite email e così via -> è semplice intuire come tale possibilità non è tanto sicura infatti dal punto di vista legale utilizzare esso non ha tanto valore

- **Directory pubblica:** • un'entità privata che gestisce le chiavi pubbliche di tutti
• ogni partecipante la gestisce a suo modo

anche in questo caso vi è un problema che riguarda la violazione dell'entità privata che consentirebbe di scrivere e sostituire qualsiasi chiave pubblica.

- **Autorità per le Chiavi Pubbliche**: è un'autorità che gestisce directory di chiavi pubbliche, essa stessa possiede una chiave pubblica visibile a tutti gli utenti, ogni qualvolta un utente chiede una chiave pubblica tale autorità la spedisce.
Svantaggi: l'autorità rappresenta un server sempre online il che significa che è funzionante 24 ore su 24 e se vi sono molte richieste vi potrebbero essere dei problemi.
Il protocollo utilizzato prevede che assieme alla richiesta venga inviato un timestamp (in quanto una chiave pubblica potrebbe variare).
Una volta che Alice e Bob hanno ottenuto le rispettive chiavi pubbliche devono essere sicuri che esse siano associate alla persona giusta(cioè che non siano state violate)
Viene utilizzato un protocollo di **MUTUA AUTENTICAZIONE.**
Cioè Bob invia ad alice un messaggio contenente la sua chiave pubblica (di BOB) e un numero casuale. Alice decifra il messaggio (lo può fare solo se la chiave pubblica che possiede Bob è quella di Alice in quanto è in relazione con la chiave privata di Alice) legge il valore casuale lo riscrive e ne aggiunge un altro inviando tutto a Bob.+
Per lo stesso motivo di Alice Bob decifra il messaggio leggendo i due valori casuali e se il suo è giusto invia il secondo (cioè quello inserito da Alice) ad Alice.
Solo se i valori casuali corrispondono Alice e Bob sono sicuri che non vi è stata nessuna violazione.
Tale tecnica non viene spesso utilizzata in quanto come detto in precedenza l'autorità per le chiavi pubbliche deve essere attiva 24 h su 24.

-**Certificati digitale:** è un file in cui è riconosciuta l'appartenenza di una chiave pubblica ad una certa entità (persona o dispositivo)

I CERTIFICATI DIGITALI

Un certificato digitale sicuramente possiede:
- la **chiave pubblica** associata all'utente a cui appartiene
- la **firma di un'autorità certificata** (certificatore) : è una stringa binaria

Altre proprietà che possono essere contenute sono:
- periodo di **validità** del certificato
- altre **informazioni sulla chiave** (algoritmi,utilizzo)
- stato della chiave
- altre **informazioni sull'utente**

IL FORMATO PIU' DIFFUSO è ITU-TX.509

Se Alice e Bob devono comunicare non chiedono direttamente all'Autorità di Certificazione di dare il certificato dell'altro ma prima avviano un protocollo di MUTUA AUTENTICAZIONE e successivamente comunicano
Quando la chiave privata di un utente viene compromessa può essere richiesta la revoca del certificato.

I **motivi** che possono essere utilizzati sono i seguenti:
- compromissione chiave privata
- informazioni non più valide
- utilizzo non più valido
- compromissione algoritmo

I **metodi** che possono essere utilizzati per revocare un certificato sono:
- si creano certificati a breve scadenza
- si utilizza un CRL (Certificate Revocation List) cioè una lista di (numeri seriali associati ai certificati)certificati revocati(quelli scaduti non sono presenti) firmata da una CA

X.509

E' lo standard per i certificati digitali più conosciuto e utilizzato.
Vi sono differenti versioni.

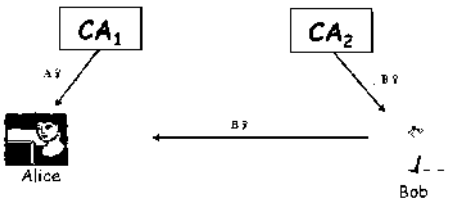
VERSIONE 1
a) versione del certificato
b) numero seriale
c) id dell'algoritmo
d) periodo di validità: due date prima e ultima della validità del certificato
e) nome utente
f) PKA(public key autority) autorità per le chiavi pubbliche

VERSIONE 2
Oltre ciò che è contenuto nella versione 1 anche:
g) id univoco dell'emittente
h) id univoco dell'utente

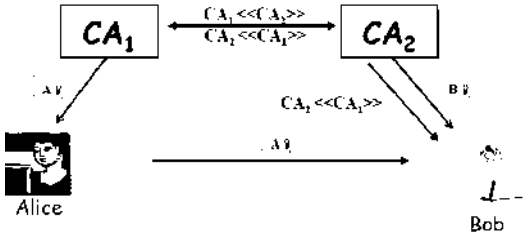
VERSIONE 3
versione 2 più
i) estensione

ALTRE VERSIONI
Viene aggiunta la firma del certificatore

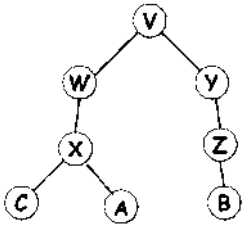
SE VI SONO PIU' CA?
Cioè se Alice si affida a CA_1 e Bob a CA_2



In tal caso i certificatori si "certificano" ovvero CA_1 certifica il certificato emesso da CA_2 (per Bob) e viceversa in questo modo Alice chiede a CA_1 il certificato di Bob e a sua volta CA_1 lo richiede a CA_2 e viceversa.



GERARCHIA DI CERTIFICATI



In tal caso se A deve verificare il certificato di B si procede nel seguente modo:
- X verifica il certificato di A (X <<A>>)
- W verifica il certificato di X
- V verifica il certificato di W
- V verifica il certificato di Y
- Y verifica il certificato di Z
- Z verifica il certificato di B