CRITTOGRAFIA SIMMETRICA

Principi:

I) Entrambi gli endpoint hanno la stessa chiave

II) Un attaccante può voler violare l'autenticità, la confidenzialità o l'integrità di tutte e 3.

TRANSPOSITION CIPHERS: la cifratura avviene permutando l'ordine dei simboli del testo in chiaro con un metodo noto agli end-point e reversibile.

Questi strumenti non offrono alcun riferimento se usati da soli.

SUBSTITUTION CIPHERS: la cifratura non permuta i simboli ma li sostituisce con altri simboli. Con una certa funzione

Modernamente una cifratio a shift (sostituzione con simboli della stessa insieme ma diverso ordine, shift fisso) avviene con operazione di SOMMA MODULARE ; ~~assegno un numero intero alla chiave del simbolo di una chiave $k \in [0, MAX\ NUM\ SIMBOLI]$ ed $m = MAX\ NUM\ SIMB.$ espresso in modo da rendere tondo il modulo~~

M = indice del simbolo in chiaro K = Chiave $\in [0, MAX\ INDICE]$

C = Carattere cifrato: $(M + K)$ % MAX NUMERO SIMBOLI

Questa cifratura è più forte ma è facilmente violabile con un attacco BRUTE FORCE

Un requisito fondamentale per un cifrario efficace è essere resistente ad attacchi BRUTE FORCE, quindi deve essere troppo costoso per un attaccante forzare la chiave.

Lo SPAZIO DELLE CHIAVI AMMISSIBILI quindi deve essere abbastanza grande

Un cifrario a sostituzione IDEALE consiste in una PERMUTAZIONE CASUALE (non basata su un procedimento matematico).

La permutazione si basa su una MAPPATURA BIUNIVOCA fissa che consente di invertire il procedimento.

Questa cifratura è violabile tramite FREQUENCY ANALYSES:

Ogni lettera in chiaro è sempre cifrata con lo stesso simbolo quindi si può "indovinare" quale lettera è conoscendo la frequenza delle lettere nella lingua originale

Questo schema è DETERMINISTICO: un input produce sempre lo stesso output

CIFRARIO VIGENÈRE

Questo cifrario è a SOSTITUZIONE POLIALFABETICA, ovvero ogni lettera è sostituita usando una chiave diversa da quella usata per le altre lettere.

Questo perché due lettere diverse possono produrre uno stesso risultato:

$$C + K = X$$
$$K + C = X$$

Difficile

~~Impossibile~~ fare analisi di frequenza.

Anche questo cifrario si può rompere, basta campionare il messaggio sapendo di capire quanto è lunga la chiave e di conseguenza riottenendo un cifrario di cesare.

In ogni caso ci vuole un testo cifrato molto più grande da analizzare.

Per ovviare all'analisi di frequenza bisogna cambiare mappatura frequentemente e questo è troppo pesante (chiavi troppo grosse da gestire).

Combinare tra loro cifrari diversi aumenta molto l'affidabilità e la proficuità del cifrario.

KERCHOFFS PRINCIPLES: principi di cifratura moderna

I) Gli algoritmi sono pubblici

II) La sicurezza si basa sulla segretezza della chiave

III) I cifrari non devono essere sicuri in modo assoluto, non devono essere rotti con risorse e tempi accettabili.

SCHEMA DI CRIPTAZIONE MODERNA:

I) Spazio delle chiavi largo abbastanza per evitare il brute force

II) Per gli attaccanti, i dati cifrati devono essere indistinguibili da dati casuali (non ci sono info)

CLASSIFICAZIONE SCHEMI DI CIFRATURA SIMMETRICI

I) Cifrari A FLUSSO
II) Cifrari A BLOCCHI

} Schemi a SICUREZZA COMPUTAZIONALE
↓
Si sa come romperli ma è troppo difficile farlo.

III) Cifrari ONE-TIME-PAD } PERFECT SECRECY (UNCONDITIONAL SECURITY)

Questo schema NON può essere violato neanche teoricamente.

Sono inutilizzabili nella pratica

SCHEMA ONE TIME PAD (OTP)

Alla base c'è una operazione di XOR poiché, dato il carattere $m = \{0,1\}$ (BIT) e una chiave $k = \{0,1\}$, la probabilità che il risultato c sia 0 o 1 è esattamente del 50% se la chiave k è CASUALE

La chiave deve essere TOTALMENTE CASUALE e lunga LUNGA QUANTO IL MESSAGGIO. Infine si esegue uno XOR BIT-A-BIT.

Questo garantisce una segretezza totale, non esiste un modo per capire se il risultato è giusto.

Questo però non è materialmente fattibile.

Sudre un attacco molto facile da portare a termine è la MANIPOLAZIONE DEI DATI, ovvero un attaccante può modificare il testo criptato ed essere sicuro di aver modificato anche il dato in chiaro.

Chi riceve non può accorgersi della manipolazione.

L'algoritmo OTP è quindi MALLEABILE

COMPUTATIONAL SECURITY

Gli algoritmi devono essere PRATICI:

I) Chiavi CORTE ma lunghe abbastanza per evitare il brute force

II) Data una chiave, la cifratura e decifratura dovrebbe essere EFFICIENTE

III) Senza chiave, la decifrazione deve essere FORTEMENTE INEFFICIENTE (rompere la chiave è troppo costoso e la probabilità di riuscirci con risorse LIMITATE dovrebbe essere TRASCURABILE)

Una funzione efficiente mantiene il costo ragionevolmente basso anche con chiave grande mentre una funzione inefficiente ha costo fortemente divergente al crescere della chiave

In generale:

$C_{ENC} = M \qquad C_{ATTAK} = 2^{**}M$

↑ super polinomiale

L'unico motivo per il quale uno schema può diventare obsoleto è la scoperta di una funzione di attacco efficace.
Il mero aumento della potenza computazionale non è un problema basta aggiornare i parametri crittografici.

CIFRARI A FLUSSO (STREAM CHIPHERS)

Molto simili all'one time pad ma non si usano chiavi VERAMENTE CASUALI.

Si usa invece una STREAM KEY, ovvero una chiave PSEUDOCASUALE ovvero una chiave generata da una FUNZIONE GENERATORE PSEUDOCASUALE la quale, dati pochi dati VERAMENTE CASUALI genera tanti dati indistinguibili da dati casuali:

POCHI DATI CASUALI → MOLTI DATI PSEUDOCASUALI

Il vantaggio rispetto a OTP è che basta sapere la chiave data in input e non la chiave usata per la cifratura.
La cifratura si fa effettua sempre con XOR.
La sicurezza dello schema risiede nella robustezza della funzione generatore pseudocasuale (detto STREAM CIPHER appunto) che rimane deterministico. Bisogna evitare che si possa indovinare la chiave data dallo stream cipher.

Questo schema eredita la caratteristica di malleabilità dell OTP.

Un'esigenza fondamentale per ogni algoritmo/protocollo è garantire la MULTI-MESSAGE SECURITY, ovvero permettere di usare la stessa chiave per cifrare diversi messaggi slegati tra loro.

Per garantire questo, uno schema di cifratura deve cifrare i dati sempre in maniera diversa, ovvero lo stesso dato cifrato 2 volte deve risultare in 2 cipher-text diversi anche usando la stessa chiave.

In sostanza quindi, anche se il generatore pseudo-random è deterministico, l'intera funzione di ~~cifratura~~ cifratura deve essere non-deterministica.

Per fare ciò gli schemi a flusso introducono un parametro aggiuntivo in input al generatore, detto NONCE.

Il valore del NONCE può essere non casuale ma deve essere usato solo 1 volta, per garantire che la funzione finale di cifratura sia non deterministica.

Per la costruzione di stream cipher si possono usare questi componenti:

I) Un generatore nativo pseudocasuale: molto veloce

II) Block cipher usati come primitive per funzionare come stream cipher

III) Primitive di crittografia asimmetrica usati per fare stream cipher: molto lento e complesso. Non si usa.

Solitamente il numero viene generato da un end-point ed inviato assieme al messaggio cifrato, in chiaro.

CIFRARI A BLOCCHI (BLOCK CIPHERS)

I block cipher non sono schemi di cifratura ma modelli matematici che permettono di costruire protocolli di crittografia (tipo AES)

I cifrari a blocchi consentono di costruire schemi di cifratura SIMMETRICA.

Il gruppo di schemi di cifratura AES definisce appunto una famiglia di schemi che implementano diversi block cipher.

A tale livello formale un block cipher è una FAMIGLIA DI PERMUTAZIONI PSEUDO-CASUALI CON CHIAVE

Fondamentalmente un block cipher è un cifrario a sostituzione (si basa quindi su permutazioni dei simboli).

Un block cipher IDEALE è semplicemente la mappatura di sostituzione polialfabetica. L'alfabeto è fatto da simboli binari (insiemi di BIT).

La BLOCK-SIZE definisce la lunghezza di ogni simbolo (blocco)

Il cifrario a blocchi ideale ha lo stesso problema di praticità descritti in precedenza poiché bisogna mantenere mapping enormi (block-size molto grande) per mantenere la cifratura sicura.

I block cipher reali utilizzano funzioni che, tramite l'uso di una CHIAVE, si comportano come la tabella di mapping senza il bisogno di mantenere/distribuire la tabella stessa.

I protocolli AES sono implementati in HW sulle CPU in modo da effettuare operazioni di cifratura molto più velocemente.

Di per se i cifrari a blocchi dividono i messaggi in blocchi lunghi M BIT (128, 192, 256 ...) e poi fa delle operazioni su quei blocchi.

ELECTRONIC CODEBOOK (ECB): ogni blocco viene cifrato con la chiave e i blocchi cifrati insieme formano il testo cifrato.

Questa modalità NON È SICURA perché espone a frequency analysis sui blocchi. Questo metodo è sufficiente il modo "diretto" di usare AES ma non si usa quasi mai.

CIPHER BLOCK CHAINING (CBC):

Ogni blocco viene offuscato prima di essere cifrato.
Per l'offuscamento si usa:

I) Un INITIALIZATION VECTOR xor il primo blocco

II) Il ~~blocco~~ cipher text del blocco n-1 xor il plain text del blocco n

Questo propaga della casualità e risolve la vulnerabilità
Per decifrare si fa esattamente l'inverso.

INITIALIZATION VECTOR: Concettualmente simile al NONCE
Grande quanto un blocco.

COUNTER MODE (CRT)

In questa modalità dell'AES possiamo creare uno stream cipher partendo da un block cipher.

Questa modalità usa vari elementi:

- KEY: chiave dell'AES, fissa
- NONCE: valore casuale ma sempre uguale per singola istanza
- COUNTER: elemento incrementale

Ogni blocco è costituito da NONCE + COUNTER e quindi ogni blocco differisce dal precedente grazie alla progressività del counter.

Ogni blocco viene cifrato dal block cipher e l'output viene usato per fare lo XOR con il plain text.

Il block cipher quindi produce dati semi-casuali, usati come stream key per cifrare il messaggio.

A seconda dell'implementazione le dimensioni di counter e nonce variano in base all'implementazione.

La dimensione del counter definisce la dimensione massima del messaggio cifrabile con singola invocazione.

Ogni invocazione (messaggio) richiede un nuovo nonce.

Un vantaggio degli stream cipher rispetto ai block cipher è la possibilità di cifrare plain text di qualsiasi dimensione mentre i block cipher richiedono l'uso di PADDING, ovvero bisogna che la dimensione del plain text sia multipla della block size.

Il padding deve essere DISTINGUIBILE dal testo originale una volta decifrato il testo.

Lo standard attuale per il padding è PKCS7 che consiste nell'aggiungere N volte il valore N di tali BIT mancanti, se il plain text è già della dimensione giusta si aggiunge un intero blocco di padding. Questo standard è definito sui BYTES.

Per evitare di riciclare (anche inavvertitamente) i nonce e gli initialization vectors si possono usare modalità di cifratura che facciano uso di SYNTHETIC INITIALIZATION VECTOR (SIV)

L'utilizzo di SIV consente di usare lo stesso nonce/IV per cifrare dati diversi perché il plain text viene usato per derivare dall'IV/nonce dato un altro dato sintetico.

Due notare che se si accoppiano nonce/IV e plain text il risultato rimane deterministico, quindi esposto a frequency analysis.

Questa sicurezza aggiuntiva viene pagata con minori prestazioni.