CRITTOGRAFIA ASIMMETRICA

Il principio generale è che le chiavi di cifratura e decifratura sono DIVERSE.

Quindi:

I) la chiave di CIFRATURA è PUBBLICA

II) la chiave di DECIFRATURA è PRIVATA

I protocolli asimmetrici vengono principalmente per implementare protocolli di SCAMBIO DI CHIAVI per poi supportare i protocolli di crittografia simmetrica.

Viene poi ampiamente usata per realizzare FIRME DIGITALI

Ogni scambio di chiavi può essere non autenticato o AUTENTICATO.

KEY EXCHANGE PROTOCOL

L'obiettivo è permettere a due parti che NON HANNO CHIAVI di scambiarsi chiavi tramite un canale NON SICURO in sicurezza.

La chiave finale sarà uguale per tutte le parti (chiave di crittografia simmetrica).

Il primo protocollo di questo tipo nasce nel 1976 e si basava su uno schema di crittografia simmetrica (oggi non più).

MERKLE PUZZLE: primo protocollo funzionante. Si basa sul fatto che l'attaccante deve spendere molte più risorse rispetto agli end point per rompere il messaggio

Vengono cifrati con cifratura debole N chiavi (una delle quali verrà usata per l'algoritmo FORTE).

Ad ogni chiave cifrata con algoritmo debole viene associato un id k (cifrato insieme alla chiave).

Le fiale vengono mescolate e inviate. Chi riceve spende N tempo per violare una fiala a caso, ottiene k e lo comunica al mittente così che entrambi sappiano quale chiave usare per E.

L'attaccante deve spendere molto tempo per trovare la fiala con la chiave scelta dai due se ci sono abbastanza fiale da dover rompere $\left(\frac{N^2}{2}\right)$

PROBLEMA DEL LOGARITMO DISCRETO IN CAMPO FINITO

Questa congettura sfruttata da DIFFIE e HELLMAN definisce un problema matematico di difficile soluzione su campi finiti è definito da un INTERO % NUMERO PRIMO

Ad esempio:

Dato il campo $\mathbb{Z}_{11} = \{0, 1, \dots, 10\}$

Data una base (GENERATORE) $\in \mathbb{Z}_{11}$ e dato un risultato $\in \mathbb{Z}_{11}$ è molto difficile calcolare l'esponente di

$$b^a \,\%\, 11 = r$$

Questa congettura si può estendere anche ad altri oggetti/gruppi matematici, come le CURVE ELLITICHE

In questo protocollo l'unico segreto è l'esponente e.

Dato un campo abbastanza grande diventa computazionalmente infattibile calcolare e, quindi il segreto è sicuro.

CONGETTURA DI DIFFIE-HELLMAN

Dato un GENERATORE g di un gruppo ciclico di ~~primo~~ ordine primo p (def. 3.11)

Si definiscono 3 problemi distinti:

I) **PROBLEMA DEL LOGARITMO DISCRETO**

Se g^a è noto, è molto difficile calcolare a

II) **DIFFIE-HELLMAN COMPUTAZIONALE**

Dati g^a e g^b è difficile calcolare g^{ab}

III) **DIFFIE-HELLMAN DECISIONALE**

Dati g^a e g^b è difficile distinguere g^{ab} da g^r dove r è random.

PROTOCOLLO DI SCAMBIO DI CHIAVI DIFFIE-HELLMAN

Si genera a e si calcola g^a (lo fa A)
Si genera b e si calcola g^b (lo fa B)
} g è pubblico

A e B si scambiano g^a e g^b.

Poiché A conosce a può calcolare $(g^b)^a = g^{ab}$
Poiché B conosce b può calcolare $(g^a)^b = g^{ab}$
} la chiave è stata generata

In questo scenario l'attaccante non è in grado di calcolare g^{ab} ne siamo a o b, quindi non è possibile nemmeno distinguere g^{ab} da dati casuali nel caso in cui l'attaccante riuscisse ad ottenerlo perché non ha a e b.

Ad oggi questo algoritmo è sicuro solo perché ad oggi non è stato trovato un algoritmo efficiente per risolverlo.

VARIANTI A CURVE ELLITICHE

Oggi nel mondo reale si utilizzano sempre le versioni a curve ellitiche perché le versioni con gli interi sono vulnerabili ad alcuni tipi di attacchi che costringono ad utilizzare parametri molto più grandi. Le versioni a curva ellitica non soffrono delle stesse vulnerabilità quindi si possono usare parametri più piccoli, quindi valore assoluto.

Le ECC curve ellitiche si possono applicare anche alle firme digitali per creare certificati di firma.

FIRME DIGITALI

Sono la versione asimmetrica delle funzioni MAC, ovvero consentono di garantire l'autenticità dei dati in contesto asimmetrico.

La funzione SIGN() prende i dati ed una chiave PRIVATA.

La funzione VERIFY() prende il messaggio, la firma e la chiave PUBBLICA associata alla chiave di firma. La verifica riesce solo se le chiavi sono associate ed i dati non sono stati modificati.

Questo protocollo ha 2 caratteristiche importanti

I) NON RIPUDIABILITÀ: chi firma non può in seguito affermare di non essere il firmatario perché solo LUI ha la chiave privata che ha generato la firma. Questo ha anche valore legale.

II) PUBBLICAMENTE VERIFICABILE: poiché la chiave di verifica è pubblica, tutti possono verificare l'autenticità di una firma senza dover dialogare con il mittente. Quindi è possibile eliminare intermediari nel processo di scambio di chiavi.

SCAMBIO DI CHIAVI CON ATTACCANTE ATTIVO

Se presupponiamo che l'attaccante diventi ATTIVO, quindi faccia qualcosa invece di ascoltare e basta, dobbiamo adattare il protocollo.

Il protocollo D-H non fornisce garanzie sull'autenticità dei dati quindi è totalmente vulnerabile ad attacchi MAN-IN-THE-MIDDLE.

In questo attacco, il traffico è intercettato da un proxy che fa D-H con i 2 end points, potendo poi leggere tutto il traffico in chiaro.

La soluzione è associare D-H a firme digitali per costruire un protocollo di SCAMBIO DI CHIAVI AUTENTICATO.

Durante D-H A invia anche un certificato di firma che B può usare per verificare l'identità di chi invia (e per DH).

Per fare ciò però B deve conoscere la chiave pubblica di A per verificare la firma (ottenuta in modo particolare con un'altro protocollo che dipende dal contesto applicativo).

NOTE SULLE CHIAVI NELLA CRITTOGRAFIA ASIMMETRICA

Come detto, nella crittografia asimmetrica esiste sempre una coppia di chiavi accoppiate in modo univoco tra loro.

Questa coppia prende il nome di KEY PAIR.

Nell'ambito delle firme digitali è possibile creare associazioni fisse del tipo KEY PAIR ⇔ UTENTE poiché solo lui conosce la chiave privata per firmare. Anche poiché anche la chiave pubblica è unica in un certo KEY PAIR (quindi data una certa chiave privata, anche la chiave pubblica identifica univocamente l'utente a cui il key pair appartiene.

La maggior parte dei protocolli assumono che gli altri utenti siano a conoscenza delle associazioni tra chiavi pubbliche e relativi utenti.

ALGORITMI STANDARD DI FIRMA DIGITALE

DSA (Digital signature Algorithm): algoritmo di firma che si basa sul problema del logaritmo discreto (lo stesso usato dal protocollo D-H)

Questo protocollo è una alternativa largamente diffusa e standard alle firme basate su chiavi RSA (versione tradizionale).

ECDSA (elliptic curve DSA): versione basata su curve ellittiche del DSA

Viene largamente preferito oggi, è di fatto lo standard di riferimento per garantire applicazioni a prova di futuro.

Questi protocolli, per funzionare correttamente, richiedono di generare un numero casuale k che permette di ottenere la firma randomizzata in modo sicuro.

A differenza dei protocolli MAC, lo schema DSA è in grado di fornire firme NON DETERMINISTICHE (ovvero dato un payload e una chiave privata la firma risultante varia ad ogni operazione di firma).

Questo protocollo consente poi di verificare la firma senza conoscere il valore di k. Se k non è randomico è possibile effettuare calcoli inversi per ottenere la chiave privata, compromettendo lo schema di firma.

Se un generatore random non può essere calcolato ogni volta per limiti tecnici è possibile usare versioni del protocollo che non lo richiede prelevando una derivata dalla chiave privata e dal payload.

Per confronto, il protocollo di firma RSA produce firme deterministiche

EdDSA (Edward DSA): versione alternativa a ECDSA che può fornire firme • DETERMINISTICHE. Questo standard è piuttosto diffuso in ambito open-source e non è uno standard NIST.

PROBLEMA RSA

Il problema RSA è un problema matematico che fornisce la base a molti protocolli di crittografia asimmetrica e costituisce una alternativa al problema dell'algoritmo del logaritmo discreto alla base di D-H.

Dati 2 NUMERI PRIMI SEGRETI p e q è possibile calcolare un MODULO PUBBLICO $n = p \cdot q$.

Dati i valori: $m = c^d \% n$ $\quad c = m^e \% n$ } Se i valori e e d non sono noti è molto difficile calcolare c e m anche conoscendo n

Le due operazioni descritte sono quindi BIDIREZIONALI, teoricamente quindi conoscendo tutte le informazioni, è possibile invertire le operazioni ma il protocollo è non UNIDIREZIONALE mantenendo segrete certe informazioni

Ad esempio, utilizzando k come CHIAVE PUBBLICA e d come CHIAVE PRIVATA, è possibile generare una FIRMA m con dei dati DATI C

Con $m = c^d \% n$ per poi verificare sole firma senza conoscere d

La firma quindi può essere prodotta solo da chi conosce d, rendendo l'operazione unidirezionale.

Da notare che i dati in input (C) devono passare da una funzione HASH perché devono essere dati numerici.

Questa implementazione è "didattica", i protocolli veri introducono complessità sulla funzione HASH per difendere il protocollo da vari attacchi avanzati.

I 3 protocolli di firma standard basati su RSA sono:

I) PKCS 1 - PSS (più moderno)
II) PKCS 1 - v. 1.5 (datato)

PSS → SCHEMA DI PADDING

(in realtà non centrano niente con il padding della crittografia simmetrica)

NOTE SU RSA

Gli algoritmi RSA esistono solo su numeri interi, non esistono versioni a curve ellittiche.

Le firme ECDSA HANNO DIMENSIONI INFERIORI a parità di livello di sicurezza

La generazione delle firme è PIÙ VELOCE CON ECDSA mentre la verifica della firma è più veloce PIÙ VELOCE CON RSA ma potrebbe diventare più lenta di crescere del livello di sicurezza (aumento del tempo di verifica divergente tra i 2 protocolli).

In generale quindi ECDSA è più conveniente in quasi tutti i contesti già oggi e dovrebbe diventarlo ancora di più nei prossimi anni.