



UNIVERSITÀ DEGLI STUDI ROMA TRE

Facoltà di Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica

Tesina di Elementi di crittografia

Pretty Good Privacy (PGP)

Autori

Alessandro Bertuletti

Fabrizio Ghigi

Marco Piolo

Anno Accademico 2006-2007

*PGP permette alla gente comune di avere
la propria privacy a portata di mano.
C'è un bisogno sociale crescente di questo.
Ecco perchè l'ho creato.
(Philip Zimmermann)*

Indice

Premessa	3
1 Storia del PGP	5
1.1 dal PGP 1.0 al 2.0	5
1.2 I problemi con i brevetti e problemi legali	6
1.3 Sviluppi durante il processo	7
1.4 PGP diventa commerciale	8
1.5 L'acquisizione della Network Associates	9
1.6 La situazione attuale	9
2 Come funziona il PGP	11
2.1 RSA	11
2.1.1 Come funziona l'RSA	11
2.2 IDEA	12
2.3 MD5	13
2.4 ZIP	14
2.5 Funzionamento del PGP	14
2.5.1 PGP ed email	18
3 Attaccare il PGP	19

Premessa

L'esigenza di sicurezza è sempre esistita ma con l'avvento e la popolarità crescente di Internet questo bisogno si è esteso ad un gruppo di persone molto più ampio.

La trasmissione di un'email attraverso Internet è, potenzialmente, molto meno sicura di una lettera spedita attraverso l'ufficio postale o il fax. Il contenuto dei testi che transitano normalmente in chiaro da un mittente ad un destinatario può essere da chiunque letto o modificato a piacere.

Non serve essere paranoici, però, bisogna riconoscere il rischio là dove esiste realmente.

Questo discorso non è così importante per un messaggio personale inviato a familiari o amici ma quando si tratta di scambiare posta confidenziale come, per esempio, il contatto con i clienti, si possono capire le implicazioni differenti in caso che qualcuno intercetti il messaggio.

La crittografia tutela i nostri diritti alla segretezza.

Esistono moltissimi algoritmi di crittografia, sono nati praticamente tutti per scopi militari e si sono enormemente evoluti nel corso del tempo.

È però difficile credere che l'ingegno umano possa elaborare un sistema crittografico che esso stesso non possa risolvere. Questo è il motivo per cui lo studio della crittografia ha cambiato rotta, concentrandosi sull'elaborazione di algoritmi non indecifrabili ma decifrabili in tempi non utili.

Pretty Good Privacy (PGP) è un programma che permette di usare autenticazione e privacy crittografica. Nelle sue varie versioni è probabilmente il crittosistema più usato al mondo. PGP nasce da Philip Zimmermann per le persone comuni e fa uso di alcuni dei più noti algoritmi di crittografia esistenti.

PREMESSA

PGP è stato così influente che il suo progetto è stato trasformato dalla IETF in uno standard Internet chiamato OpenPGP. Le versioni di PGP più recenti dello standard sono, bene o male, con esso compatibili.

Capitolo 1

Storia del PGP

Il Pretty Good Privacy (PGP), letteralmente “una privacy abbastanza buona”, è un pacchetto software originalmente sviluppato da Philip R. Zimmermann (P.R.Z. secondo la mania americana per gli acronimi) nel 1991 integrativo di programmi già esistenti che fornisce routine crittografiche per e-mail e applicazioni di immagazzinamento dati. Zimmermann prese vari codici già operativi e protocolli crittografici e sviluppò un programma che poteva girare su piattaforme multiple, come Windows, Unix, MS-DOS, OS/2, Macintosh, Amiga ed Atari.

1.1 dal PGP 1.0 al 2.0

La sua prima versione PGP 1.0 (ora siamo alla 6.5.1i, con la i finale per Internazionale, visti i problemi legali) usava il MD4 e il RSA per le firme e lo scambio delle chiavi segrete usate dal cifratore a blocco Bass-O-Matic, di ideazione dello Zimmermann stesso, insieme al LZHuf (Lempel-Ziv Huffman, algoritmo di compressione adattivo) per la compressione dati ed il uuencoding ASCII armour per il trasporto a 7 bit.

PRZ passò il PGP 1.0 ad alcuni amici che riuscirono a trasportarlo fuori dagli U.S.A. prima che fosse promulgata una legge che avrebbe praticamente messo fuori legge tutti i sistemi di cifratura che non avessero un accesso dedicato ai servizi segreti.

In qualche modo il programma si fece strada tra gli utenti e incominciò

ad esserci un interesse crescente intorno al programma. Interesse che spinse l'autore a sviluppare una versione 2.0 del programma: il Bass-O-Matic era debole e fu rimpiazzato dall'allora recente IDEA, MD5 rimpiazzò la versione MD4, e fu introdotto il radix-64 ASCII armour insieme al codice di compressione ZIP.

1.2 I problemi con i brevetti e problemi legali

Le versioni iniziali di PGP hanno avuto anche dei problemi con i brevetti già registrati. La prima versione ricorreva a un cifrario progettato dallo stesso Zimmermann e denominato Bass-O-Matic da uno sketch del Saturday Night Live in cui si utilizzavano dei pesci e dei tritatutto in cucina. Ben presto ci si accorse però che questo cifrario non era sicuro, così venne rimpiazzato dal cifrario IDEA. Entrambi gli algoritmi, sia quello per la chiave simmetrica IDEA che quello per la chiave asimmetrica RSA, sono stati brevettati e per utilizzarli è necessaria un'autorizzazione. All'epoca ci fu un acceso dibattito sul fatto che Zimmermann avesse avuto l'autorizzazione per utilizzare il RSA nel PGP. A tal proposito, Zimmermann dichiarò che l'allora RSA Data Security (operativa adesso come RSA Security), in uno dei primissimi incontri, gli avesse dato l'autorizzazione purché fosse a scopo non-commerciale mentre RSA smentiva tutto quanto. Di conseguenza, fu un reclamo partito dalla RSADSI alla Dogana statunitense a creare il cosiddetto caso Zimmermann sull'uso dell'algoritmo RSA nel PGP.

Per complicare ulteriormente il quadro, l'algoritmo RSA venne brevettato solamente negli USA (questo per la difficoltà di rispondere alle varie modulistiche di richiesta brevetti nel mondo) con il risultato di poter essere utilizzato liberamente (tenendo sempre conto delle problematiche di brevetto viste in precedenza) in tutte le altre nazioni. Ciò nonostante, gli inventori/proprietari di IDEA furono decisamente più liberali negli USA che nell'Unione Europea. E se non ci fosse già abbastanza confusione, il brevetto sull'algoritmo RSA era parzialmente controllato dal MIT attraverso il proprietario del brevetto, il RSADSI: gli inventori del RSA lavoravano tutti al MIT al momento della sua creazione.

In qualunque modo andò la disputa Zimmermann/RSADSI, il MIT ebbe

pochi problemi con il PGP; invece, si trovò parecchio in difficoltà a causa della posizione ostile del RSADSI, contraria all'uso non commerciale del RSA nel PGP. Il risultato del conflitto sulla licenza del RSA su un fork di PGP in:

- una versione USA (conforme al brevetto RSA) che usa una libreria crittografica shareware dell'RSA
- una versione internazionale che usa il codice RSA originale creato da Zimmermann ed i suoi collaboratori

La versione USA fu distribuita direttamente dallo stesso MIT, insieme ad altri, attraverso Internet, le BBS ed utenti e gruppi di sistemi di comunicazione privata come AOL e CompuServe. Alla fine sul sito del MIT, c'era il requisito che l'indirizzo email al quale PGP sarebbe stato inviato fosse negli USA od in Canada, e che il ricevente fosse residente in uno dei due stati.

In Norvegia, Ståle Schumacher Ytterborg sviluppò e mantenne la versione internazionale del PGP, che venne chiamata PGP-i (dove i sta per internazionale). Era desiderabile al tempo che la versione internazionale fosse sviluppata e mantenuta fuori dagli USA per evitare ulteriori difficoltà con le regolamentazioni USA sull'esportazione e con il brevetto RSA.

Il clima di illegalità aiuta lo spandersi di dicerie finché il governo U.S.A. non aprì un'inchiesta su PRZ ed il suo programma, ufficialmente per problemi di licenze e violazione di alcune leggi (come l'ITAR, International Traffic in Arms Regulations, che controlla l'esportazione di armi e materiale militare, tra cui i software di crittografia) in realtà perché impediva alla NSA di esercitare il controllo desiderato. L'inchiesta durò fino al 1999 ed alla fine PRZ venne scagionato.

1.3 Sviluppi durante il processo

Nel frattempo Zimmermann e la MIT sviluppano la versione 2.5 e 2.6 sostituendo alcune librerie (essenzialmente la RSAREF) per rendere il programma perfettamente legale dal punto di vista dei brevetti e licenze, sebbene non esportabile (almeno non legalmente).

Questo fu aggirato dato che la MIT pubblicava libri con il codice sorgente completo (la versione internazionale fu ideata appositamente per l'occasione) ed un carattere di stampa adatto all'uso di OCR (programmi di conversione da immagine di testi in testi veri e propri), permettendo non solo la ricostruzione dell'intero pacchetto software, ma anche l'analisi del sistema e l'eventuale presenza di errori o backdoor (speciali accessi nascosti all'utente per un'intrusione clandestina successiva).

Successivamente il team di Zimmermann iniziò a lavorare ad una nuova versione di PGP chiamata PGP 3. Questa nuova versione doveva avere considerevoli miglioramenti nella sicurezza, compresa una nuova struttura di certificato che non aveva i piccoli problemi di sicurezza dei certificati usati da PGP 2.x oltre che permettere ad un certificato di avere chiavi separate per la firma e per la cifratura.

PGP 3 ha introdotto l'uso dell'algoritmo simmetrico CAST-128 (chiamato anche CAST5) e degli algoritmi asimmetrici DSA e Elgamal, nessuno di questi coperto da brevetto.

1.4 PGP diventa commerciale

Dopo l'indagine terminata nel 1996, Zimmermann ed il suo gruppo fondarono una compagnia per produrre nuove versioni di PGP. Si fusero con Viacrypt (alla quale Zimmermann aveva venduto i diritti commerciali di PGP e che aveva acquistato la licenza RSA direttamente dalla RSADSI), che cambiò nome in PGP Incorporated. Il nuovo team Viacrypt/PGP cominciò a lavorare alle nuove versioni di PGP basate su PGP 3. A differenza di PGP 2, che aveva esclusivamente una interfaccia a linea di comando, PGP 3 fu progettato sin dall'inizio per essere una libreria software, permettendo agli utenti di lavorare sia da linea di comando che grazie ad una interfaccia grafica. L'accordo originale tra Viacrypt e il gruppo di Zimmermann precisava che Viacrypt avrebbe rilasciato versioni con numeri pari, mentre Zimmermann con numeri dispari. Viacrypt, quindi, creò una nuova versione (basata su PGP 2) che chiamò PGP 4. Per eliminare il dubbio che PGP 4 fosse un successore di PGP 3 (PGP 4 derivava appunto da PGP 2), PGP 3 fu rinominato e rilasciato come PGP 5 nel Maggio del 1997.

All'interno di PGP Inc., c'erano ancora dubbi riguardo ai brevetti. RSAD-SI criticava la cessione della licenza RSA della Viacrypt alla nuova società nata dalla fusione dei due gruppi. PGP Inc. adottò un'informale standard interno chiamato Unencumbered PGP (letteralmente, PGP senza intralci): non usare nessun algoritmo con problemi di licenza.

1.5 L'acquisizione della Network Associates

Nel Dicembre del 1997, la PGP Inc. venne acquistata dalla Network Associates, Inc., e Zimmermann ed il gruppo PGP divennero suoi impiegati. La NAI continuò a sperimentare l'esportazione attraverso la pubblicazione del software, diventando la prima compagnia ad avere una strategia di esportazione legale, grazie alla pubblicazione del codice sorgente. Sotto questa protezione, il team PGP aggiunse al programma originale nuove funzionalità, come la criptazione dei dischi, firewall per desktop, rilevamento delle intrusioni, e le VPN di IPsec. Dopo la legalizzazione delle esportazioni del 2000 non fu più necessario pubblicare il codice sorgente, e la NAI smise di farlo, nonostante le obiezioni del gruppo PGP. Gli utenti di PGP sparsi in tutto il globo furono decisamente delusi, ed inevitabilmente, si formarono alcune teorie cospirative contro la NAI.

All'inizio del 2001 Zimmermann lasciò la NAI. Ha lavorato come Crittografo Capo per la Hush Communications, che fornisce un programma per email basato su OpenPGP, Hushmail. Ha anche lavorato con Veridis e altre società.

Nell'Ottobre 2001 la NAI ha annunciato che i suoi diritti sul PGP erano in vendita e che avrebbe sospeso lo sviluppo di PGP. Nel Febbraio 2002, la NAI ha cancellato ogni progetto riguardante PGP.

1.6 La situazione attuale

Nell'estate del 2002, alcuni ex-membri del gruppo PGP hanno acquistato i diritti di PGP dalla NAI (eccetto per la versione a linea di comando), e, nell'Agosto dello stesso anno, hanno formato una nuova compagnia, la PGP Corporation. PGP Corp. sta supportando gli utenti attuali di PGP e onora i

precedenti contratti di assistenza. Zimmermann ora lavora come consulente speciale alla PGP Corp., e continua il suo lavoro alla Hush Communications e alla Verdis, e infine gestisce una propria compagnia di consulenza.

NAI conserva i diritti sulla versione a linea di comando di PGP e continua a venderla come McAfee E-Business Server. Prima del Gennaio 2004, era vietato alla PGP Corp. di produrre una versione a linea di comando di PGP, a causa dei suoi precedenti accordi con la NAI.

In cooperazione con Zimmermann, Verdis ha sviluppato e distribuito una versione di OpenPGP compatibile con quella a linea di comando, Filecrypt. I codici sorgenti di Filecrypt e GnuPG sono entrambi disponibili, così come quelli di versioni precedenti per varie piattaforme.

A partire dal suo acquisto dei diritti della NAI nel 2002, la PGP Corp. ha offerto assistenza tecnica in tutto in mondo. Nel 2002 ha rilasciato PGP 7.2 per Mac OS 9, seguito rapidamente da PGP 8.0 e PGP 8.0 Personal per Mac e Windows, e da una versione freeware ed una a sorgente aperto. Nel 2003, ha rilasciato PGP 8.0.1DE per gli utenti Tedeschi, PGP Universal (basato su una nuova concezione di PGP per server, che implementa una architettura di sicurezza che si auto-gestisce). Nel 2004, ha rilasciato PGP Desktop 8.1 per Mac e Windows, PGP Command Line 8.5 per Windows, Solaris e Linux (l'accordo riguardo la linea di comando tra la PGP Corp. e la NAI scadeva nel Gennaio 2004) e PGP Universal 1.2. Nel 2005 hanno rilasciato PGP Desktop 9.0, PGP Universal 2.0 e PGP Command Line 9.0.

Capitolo 2

Come funziona il PGP

Il PGP sfrutta quattro programmi base combinati secondo necessità: RSA, IDEA, MD5, ZIP.

2.1 RSA

L'algoritmo RSA è stato descritto nel 1977 da Ronald Rivest, Adi Shamir e Leonard Adleman al MIT; le lettere RSA vengono proprio dalle iniziali dei cognomi.

L'RSA è un algoritmo di crittografia asimmetrica, utilizzabile per cifrare o firmare informazioni.

2.1.1 Come funziona l'RSA

Per semplificare il funzionamento immaginiamo che A debba spedire un messaggio segreto a B. Occorrono i seguenti passaggi:

1. B sceglie due numeri primi molto grandi (per esempio da 300 cifre) e li moltiplica con il suo computer (impiegando meno di un secondo).
2. B invia il numero che ha ottenuto ad A. Chiunque può vedere questo numero.
3. A usa questo numero per crittografare il messaggio.
4. A manda il messaggio a B, che chiunque può vedere ma non leggere.

5. B riceve il messaggio e utilizzando i due fattori primi, che solo lui conosceva decifra il messaggio.

A e B hanno impiegato pochi secondi a cifrare e decifrare, ma chiunque avesse intercettato le loro comunicazioni impiegherebbe milioni di anni per scoprire i due fattori primi, con cui si può decifrare il messaggio.

In realtà questo sistema non è così semplice e per trasmettere grandi quantità di dati occorre tanto tempo, quindi A e B si scambieranno con questo sistema una chiave segreta (che non occupa molto spazio), che poi useranno per comunicare tra loro usando un sistema a crittografia simmetrica, più semplice, sicuro e veloce.

2.2 IDEA

Dopo il DES, o meglio quando si intuiva che il sistema non sarebbe ancora resistito per molto agli attacchi degli analisti, è stato proposto un cifrario chiamato IDEA.

IDEA (International Data Encryption Algorithm) è nato nel 1991 sotto il nome di IPES (Improved Proposed Encryption Standard), ed è stato progettato da due famosi ricercatori in Svizzera: Xuejia Lai e James L. Massey. Come il DES è un codice cifrato a blocchi di 64 bit, la differenza sta nel fatto che questa volta però la chiave è di 128 bit, che dovrebbe eliminare qualsiasi possibilità di riuscita di ricerca della chiave procedendo per tentativi, le chiavi possibili sono infatti 2^{128} .

La cifratura con IDEA comporta una divisione del blocco di 64 bit del testo normale in 4 sottoblocchi di 16 bit. Ogni sottoblocco subisce 8 passi in cui sono coinvolte 52 sottochiavi diverse a 16 bit ottenute dalla chiave a 128 bit. Le sottochiavi sono generate in questo modo:

1. La chiave a 128 bit è divisa in 8 blocchi di 16 che costituiscono le prime 8 sottochiavi.
2. Le cifre della chiave a 128 sono spostate di 25 bit a sinistra in modo da generare una nuova combinazione, il cui raggruppamento ad 8 bit fornisce le prossime 8 sottochiavi.

3. 3. Il secondo passo è ripetuto finché le 52 sottochiavi sono generate.

Ogni passo comporta calcoli abbastanza semplici come XOR (operazioni di OR esclusivo), addizione e moltiplicazioni in modulo 16 (significa che i risultati non possono superare i 16 bit quindi quelli eccedenti vengono scartati).

Durante gli 8 passi il secondo e il terzo blocco si scambiano di posto mentre all'ultimo passo i 4 sottoblocchi vengono concatenati per produrre un blocco di testo cifrato a 64 bit.

La decodifica è identica eccetto il fatto che le sottochiavi sono ottenute in maniera diversa dalla chiave principale a 128.

IDEA è al momento il cifrario a chiave segreta più utilizzato quanto riguarda i software commerciali di crittografia vista la sua velocità di codifica e decodifica e la sua elevata sicurezza.

2.3 MD5

L'MD5 (acronimo di Message Digest algorithm 5) è un algoritmo per la crittografia dei dati a senso unico realizzato da Ronald Rivest nel 1991 e standardizzato con la RFC 1321.

Questo tipo di codifica prende in input una stringa di lunghezza arbitraria e produce in output una firma digitale sotto forma di stringa a 128 bit (ovvero con lunghezza fissa di 32 valori esadecimali, indipendentemente dalla stringa di input). La codifica avviene molto velocemente e si presuppone che l'output (noto anche come MD5 Checksum o MD5 Hash) restituito sia univoco (ovvero si ritiene che sia impossibile ottenere con due diverse stringhe in input una stessa firma digitale in output) e che non ci sia possibilità, se non per tentativi, di risalire alla stringa di input partendo dalla stringa di output (la gamma di possibili valori in output è pari a 16 alla 32esima potenza).

La crittografia tramite algoritmo MD5 viene applicata in tutti i settori dell'informatica che lavorano con il supporto delle firme digitali o che comunque trattano dati sensibili. Ad esempio, viene utilizzata per controllare che uno scambio di dati sia avvenuto senza perdite, semplicemente attraverso il confronto della stringa prodotta dal file inviato con quella prodotta dal

file ricevuto. Con lo stesso metodo si può verificare se il contenuto di un file è cambiato (funzione utilizzata dai motori di ricerca per capire se una pagina deve essere nuovamente indicizzata). È diffuso anche come supporto per l'autenticazione degli utenti attraverso i linguaggi di scripting Web server-side (PHP in particolare): durante la registrazione di un utente su un portale internet, la password scelta durante il processo verrà codificata tramite MD5 e la sua firma digitale verrà memorizzata nel database (o in qualsivoglia contenitore di dati). Successivamente, durante il login la password immessa dall'utente subirà lo stesso trattamento e verrà confrontata con la copia in possesso del server, per avere la certezza dell'autenticità del login.

2.4 ZIP

Lo ZIP è un algoritmo di compressione utilizzato in molti sistemi di crittografia.

La compressione dei dati fa risparmiare tempo nella trasmissione via modem (anche se del tempo aggiuntivo è comunque richiesto dall'operazione stessa di compressione) ma soprattutto aumenta la sicurezza della cifratura. Molte tecniche di crittoanalisi infatti sfruttano le ridondanze presenti nel testo in chiaro per violare il cifrario: tali ridondanze sono ridotte dalla compressione, migliorando la resistenza a questo tipo di attacchi. È possibile comprimere soltanto quei files che non risultano esser troppo corti.

2.5 Funzionamento del PGP

Il messaggio in chiaro viene comunque compresso con lo ZIP per poter velocizzare l'operazione. La compressione si fa anche quando si usano le codifiche perché, oltre a far risparmiare spazio e tempo (ed il tempo può essere significativo quando si hanno a che fare con i numeri usati dagli algoritmi), rafforza la sicurezza eliminando la ridondanza del testo.

La cifratura sfrutta l'algoritmo RSA con le coppie di chiavi P, pubblica, e S, segreta, insieme al cifratore a blocco IDEA, la cui chiave è k . Il PGP genera casualmente la chiave k a 128 bit, e cifra il messaggio M: $\text{IDEA}(M, k)$.

La chiave pubblica del ricevente $P1$ viene usata per lo scambio della chiave segreta: $P1(k)$. In totale viene spedita la coppia $\{IDEA(M, k); P1(k)\}$.

Per leggere il messaggio, prima si recupera chiave segreta k tramite la propria chiave privata $P0$, e poi si decifra l'IDEA con la chiave ottenuta.

La firma è ottenuta in due passi distinti, prima si usa l'algoritmo *hash* sul messaggio M : $MD5(M)$. Quindi si applica la propria chiave privata per ottenere la firma: $S0[MD5(M)]$. Si spedisce la coppia $\{M; S0[MD5(M)]\}$. In genere, la firma è posta prima del messaggio effettivo che si sta spedendo, ma se si preferisce l'opzione "firma chiara", la firma è posta sotto il messaggio.

L'autenticazione si basa sulla verifica del nostro messaggio $\{M; S0[MD5(M)]\}$ da parte del ricevente, che possiede la nostra chiave pubblica. Applicandola si ottiene: $MD5(M) = P0\{S0[MD5(M)]\}$. Poi si confronta il risultato con il nostro calcolo di $MD5(M)$, effettuabile sul messaggio ricevuto. Se i due coincidono, la verifica ha avuto esito positivo.

Il problema si sposta sul sapere se la chiave pubblica che uno sta usando per la verifica appartiene realmente o meno alla persona da cui dice di provenire. Per questo esistono i certificati che consentono di legare un'entità alla sua chiave tramite un intermediario di fiducia; nel PGP è un conoscente diretto od indiretto che garantisce. Uno dei vantaggi del PGP è di non aver bisogno di una infrastruttura di supporto per questo, la distribuzione dei certificati è manuale o automatica, semplicemente allegandoli ai messaggi, senza aver bisogno di una gerarchia di autenticazione (anche se è capace di leggere ed importare nel suo formato la certificazione X.509).

Esiste anche la possibilità di firmare e cifrare il messaggio M , combinando le due strategie già viste come segue creando $X = \{M; S0[MD5(M)]\}$ e quindi lo cifra: $\{IDEA(X, k); P1(k)\}$.

L'operazione inversa della precedente consiste dunque prima nel recuperare la chiave segreta tramite la propria chiave privata, decifrare con la chiave segreta fornisce la firma da verificare nel modo usuale. E' ovvio che il PGP procede automaticamente nella sequenza di azioni senza coinvolgere l'utente se non nell'opzione di cifratura e nella lettura dei messaggi.

Le chiavi dell'utente sono conservate in forma cifrata; il PGP alloca le chiavi in due file distinti sull'hard disk: uno per le chiavi pubbliche e l'altro per quelle private. Questi file sono detti portachiavi (*keyrings*), e quando

si usa il PGP le chiavi pubbliche dei riceventi sono aggiunte al portachiavi pubblico. Perdere il portachiavi privato rende incapaci di leggere i messaggi cifrati con tali chiavi. I vari portachiavi, poi, sono protetti da una parola d'ordine che, se è facilmente prevedibile, è l'unica debolezza dell'insieme. In realtà non c'è una parola singola a proteggere gli anelli, ma una frase intera, che passata attraverso una funzione *hash* decifra i file contenenti gli anelli; se dimenticata si perde tutto.

I certificati digitali semplificano il lavoro di stabilire se una chiave pubblica appartiene al possessore conclamato; è dunque una sorta di credenziale. Il certificato generico consiste in tre cose:

- una chiave pubblica
- informazioni certificate: informazioni sull'identità dell'utente, come il nome, ID utente e così via
- una o più firme digitali

Le firme digitali sul certificato attestano che le informazioni presenti sono garantite da un'altra persona od identità di fiducia. La firma digitale non attesta l'autenticità del certificato; comprova solo che le informazioni di identità che l'accompagnano sono univocamente legate alla chiave pubblica. I certificati sono usati nel caso di dover scambiare chiave pubbliche con qualcun altro, assegnandogli inequivocabilmente la provenienza; nel PGP la creazione e la distribuzione è diretta, ma a questo scopo esistono anche dei depositi pubblici in rete detti Certificate Servers, oppure sistemi più strutturati come le Public Keys Infrastructures (PKI), con le proprie Autorità di Certificazione (CA) autorizzate legalmente a rilasciare, revocare, depositare, recuperare, verificare e firmare certificati ad utenti (un esempio di questi è il certificato X.509).

Il certificato PGP comprende (ma non si limita a):

- Chiave pubblica del possessore del certificato, insieme all'algoritmo per la chiave: RSA, DH, DSA
- Informazioni sul possessore del certificato, come il nome del suo user ID, la foto e così via

- La firma digitale del possessore del certificato (auto firma), fatta con la chiave privata
- Altre firme di attestazione
- Periodo di validità del certificato
- L'algoritmo simmetrico preferito per la chiave, scelto tra IDEA, CAST, o triplo DES

Come quantificare la fiducia che riponiamo nelle entità che presentano e firmano i certificati? Ci sono tre modelli di fiducia: diretta (per conoscenza esplicita della persona), gerarchica (una persona è garantita da una figura di calibro superiore, così anche lei e via dicendo), a ragnatela (che è circa una composizione delle precedenti). La ragnatela è la base del PGP: quando un utente firma un certificato altrui, diventa presentatore della specifica chiave, e quando il processo va avanti (un utente firma il successivo e così via), si forma una ragnatela di fiducia (*web of trust*). Nel PGP ogni utente agisce come autorità di certificazione convalidando la chiave, ma la sua qualifica differisce di persona in persona (a differenza di un'entità autorizzata legalmente, che è parificata), che in effetti gli assegna la propria opinione personale per identificarne il livello di fiducia riposta. Nel proprio portachiavi per ogni chiave, l'utente ha due indicatori: se la chiave è ritenuta valida o meno, ed il livello di fiducia assegnato dall'utente alla chiave stessa in base al suo possessore. Quindi la fiducia che poniamo sul possessore è anche la fiducia che poniamo sui certificati che lui convalida.

I tre livelli di validità assegnabili alla chiave sono: valida, marginalmente valida, invalida.

I tre livelli di affidamento assegnabili alla chiave sono: completa, marginale, nessuna (a parte quella implicita che riguarda la propria chiave).

Cosicché quando ci arriva una nuova chiave pubblica, per vedere se appartiene davvero ad una certa persona si analizza il certificato. Quando il certificato è firmato da una chiave, si cerca nel proprio portachiavi la suddetta e si controlla la sua affidabilità: se va bene, di conseguenza anche il certificato è approvato e la chiave corrisponde univocamente ad un certo utente. Una conferma è data dalla verifica della firma presentata (notare

che la chiave che ha firmato corrisponde univocamente ad una persona, che è quindi un presentatore).

Il PGP richiede almeno una firma di fiducia completa o due marginali per stabilire se una chiave è valida.

2.5.1 PGP ed email

Anche se col PGP è possibile cifrare ogni tipo di dato o file, l'uso prevalente è per proteggere le e-mail che non hanno un sistema di sicurezza nativo. Il PGP e l'S/MIME sono i due sistemi di sicurezza per le e-mail attualmente specificati come standard dal NIST.

Sono disponibili plug-in che implementano le funzionalità del PGP per molte delle più popolari applicazioni e-mail (come Outlook, Outlook Express, Eudora, Evolution, Mutt, Mozilla Thunderbird, Mail e molti altri). Molti sono inclusi con molte distribuzioni PGP.

Dal punto di vista della sicurezza, ognuno di questi plug-in è un PGP indipendente; ognuno potrebbe essere affetto da errori di implementazione o interagire in maniera insicura con il PGP o altri software. Utilizzare questi plug-in non fornisce necessariamente lo stesso livello di sicurezza assicurato dall'uso del PGP in maniera corretta e come applicazione a sé stante. Al più, questi programmi aggiuntivi possono fornire la stessa sicurezza del PGP e nel caso peggiore potrebbero non fornire alcuna sicurezza. Distinguere tra questi casi non è banale neanche per gli esperti nel campo. Il miglior consiglio per il normale utente è di provare l'intero sistema mandando un messaggio a sé stesso o ad un conoscente periodicamente, specialmente dopo ogni modifica od aggiornamento del software. La modalità di utilizzo più sicura consiste nel cifrare a mano e firmare messaggi ed e-mail manualmente. Ad ogni modo, come tutte le altre considerazioni sulla sicurezza, si arriva necessariamente ad un compromesso fra i vincoli del sistema e le esigenze dell'utente. Comunque, qualunque sia il rischio in un sistema di sicurezza di qualità, non utilizzarlo è comunque più rischioso.

Capitolo 3

Attaccare il PGP

Il PGP come già detto in precedenza è un sistema ibrido che sfrutta una combinazione di 4 elementi interni passibili di attacco. I programmi usati sono: un cifratore simmetrico (IDEA), un cifratore asimmetrico (RSA), una funzione hash a senso unico (MD5), e un generatore di numeri pseudocasuali (PRNG). La funzione ZIP seppur usata non è direttamente una funzione di cifratura.

1. Il cifratore simmetrico IDEA (International Data Encryption Algorithm) è stato inventato da Xuejia Lai e James Massey nel 1991, opera su blocchi dati da 64 bit con chiavi da 128 bit, e sin da allora non c'è stato nessun progresso nella sua analisi; l'unico metodo di attacco è la forza bruta. Semplificando al massimo tutti discorsi fin qui fatti, possiamo affermare che in media la chiave verrà trovata a metà dello spazio delle chiavi, cioè 127 bit, che in decimale corrisponde a: 170141183460469231731687303715884105728 chiavi da provare in media prima di trovare quella giusta. Il che esclude categoricamente una ricerca esaustiva.
2. Il cifratore asimmetrico RSA si basa, come dovrebbe essere noto, sulla congettura della difficoltà nel fattorizzare numeri prodotti dalla moltiplicazione di due giganteschi numeri primi. Non è però provato né che il problema sia difficile, né che la fattorizzazione sia l'unica o la migliore via da seguire. Ciononostante, che ci siano dei progressi in queste direzioni è più che remoto; anche se nessuna scorciatoia è plausibile al

giorno d'oggi, la potenza dei computer, l'abbassarsi continuo dei costi dell'hardware e il miglioramento delle tecniche di fattorizzazione, potenzialmente cospirano contro la sicurezza del RSA, la quale rimane inviolata con le attuali capacità di calcolo (si veda anche l'appendice 4 a questo proposito). Le tabelle qui sotto possono dare un'idea della difficoltà (1 MIPS-anno è un computer a 1 Milione d'Istruzioni Per Secondo acceso per un anno) della fattorizzazione, e nella seconda c'è una stima di equivalenza nelle ricerca/fattorizzazione esaustiva delle chiavi usando l'algoritmo setaccio dei campi numerici (NFS):

Grandezza della chiave	MIPS anni richiesti alla fattorizzazione
512	$3 * 10^4$
768	$2 * 10^8$
1024	$3 * 10^{11}$
2048	$3 * 10$

Simmetrico	Asimmetrico
56 bit	384 bit
64 bit	512 bit
80 bit	768 bit
112 bit	1792 bit
128 bit	2304 bit

Un'altro attacco possibile al RSA è quello basato sul crittogramma scelto, visto che la chiave di cifratura è pubblica e poi quello basato sulla cifratura con piccoli esponenti.

Un'arma contro il RSA non basato sull'algoritmo in se, è l'attacco temporale che nota il tempo impiegato dall'algoritmo per i calcoli, da cui deduce la grandezza dei numeri. Il PGP è però immune a ciò visto che usa il teorema del resto cinese (CRT si veda appendice 1 e 5) per velocizzare le operazioni, il che elimina parte della dipendenza dal tempo, ed inoltre la de/cifratura è fatta off-line, per cui non visibile ad un intruso.

3. La funzione *hash* come il MD5 è tale da ridurre un messaggio arbitrario in un riassunto di lunghezza fissata (128 bit), ed in teoria a senso unico,

ossia dall'uscita non dovrebbe essere possibile ricavare l'ingresso e privo di collisioni, cioè non esistono due ingressi diversi che diano la stessa uscita. Quindi non rimane che la forza bruta per forzare l'unicità del senso: trovare un messaggio M' tale che $MD5(M') = MD5(M)$, con M il messaggio richiederebbe provare circa un miliardo di messaggi per $1.07 \cdot 10^{22}$ anni. Anche l'uso del paradosso del compleanno per forzare le collisioni non ha successo se non in un tempo grottesco (585 anni nelle stesse condizioni di prima).

L'analisi differenziale controlla i crittogrammi i cui testi in chiaro hanno specifiche differenze e le analizza nel modo in cui si propagano. Ha successo reale su di uno stadio del MD5, ma non può niente contro 4 stadi consecutivi.

4. I file contenenti i portachiavi sono cifrati grazie al *hash* della frase d'ordine; il fatto è che una lingua parlata ha la sua ridondanza ed entropia, come già visto. Per l'Inglese corrente ogni carattere (codificato da 8 bit ASCII) ha 1.3 bit di entropia/informazione per carattere; dunque una frase da 128 bit di codifica (l'uscita del MD5) avrà 128 bit di entropia solo se l'ingresso è di almeno 98 lettere (ovvero 128 bit di entropia in ingresso): $(8/1.3) * (128/8) = (128/1.3) = 98.46$ caratteri. Non è proponibile una cosa del genere, quindi è davvero possibile capire la frase d'ordine che cela le chiavi se si usano meno caratteri. Se però la frase d'ordine fosse una sequenza casuale di lettere, la ridondanza verrebbe a mancare e l'entropia per carattere aumenta $\log(26)/\log(2) = 4.7$ bit. Se le lettere sono casuali ne bastano solo 27 per avere un'uscita della funzione hash che sia di 128 bit d'entropia: $(8/4.7)*(128/8) = 27.23$ caratteri.
5. I generatori di numeri pseudo casuali (PRNG) sono in realtà due nel PGP: il generatore ANSI X9.17 ed una funzione che misura l'entropia tra le pause di battitura dei tasti dell'utente, la trueRand. Quest'ultimo forma un "bacino" di numeri casuali (il file randseed.bin) che saranno l'ingresso del ANSI X9.17; lo stato interno del X9.17 ed il randseed vengono "lavati" prima e dopo la generazione dei numeri casuali. Questi

generatori casuali formano poi la chiave sessione dell'IDEA, che risulta essere particolarmente sicura.

6. Attacchi al supporto hardware: questo tipo di attacchi esula dal contesto e si rimanda alle fonti in caso di interesse, citiamo solo l'esistenza di programmi installati abusivamente che memorizzano la battitura dei tasti (leggendo le frasi d'ordine), i sistemi di sorveglianza esterna TEMPEST (che legge a distanza i monitor) o programmi che fungano da cavalli di troia per accedere a file privati.

In sostanza il PGP fornisce uno dei migliori programmi di sicurezza esistenti, quasi da essere lo standard pratico di fatto nella cifratura mondiale.