Seminario – Network Security

Introduzione

NETWORK ATTACK MODELS:

I tipi di attacchi si possono dividere in due categorie:

- Passivi: si limitano ad ascoltare il traffico di rete.
    - *Eavesdropping/Sniffing* (Intercettazione)
    - *Wiretapping* (Intercettazione telefonica)
    - *Traffic Analysis:* Identificare i modelli di comunicazione; può essere possibile anche se l'attaccante non può leggere i singoli messaggi.
- Attivi: modifica messaggi, inserimento messaggi e alterazione delle informazioni di gestione della rete
    - *Spoofing attack:* Inviare messaggi con falso indirizzo del mittente
    - *Flooding (bombing) attack*: Grande quantità di messaggi inviati alla vittima
    - *Squatting attack:* Si tenta di occupare il computer della vittima

ESEMPI:

TCP Session Hijacking: dirottamento della sessione TCP

1. A → B : SYN,ISSa
2. B → A : SYN ACK, ISSb,ACK(ISSb)
3- A → B : ACK, ACK(ISSb)

Questo scambio di informazioni e chiamato Tree-way handshake.
ISSa,ISSb sono numeri di 32-bit e ACK può essere vista come una funzione tale che ACK(ISSx) = ISSx + 1.

Un possibile attacco a questo meccanismo è il seguente:

3. C(A) → B : SYN, ISSc
4. B → A : SYN ACK, ISSb', ACK(ISSc)
5. C(A) → B : ACK, ACK(ISSb')

C invia un pacchetto con l'IP di A e intercetta la risposta per inviare a B l'ack finale. A questo punto b crede di parlare con A e invece parla con C. C ha gli stessi diritti di A su B.

IPsec (IP Security)

IP è un protocollo connection-less e state-less, che trasmette pacchetti IP (datagrams). IPsec è l'abbreviazione di **IP Security** ed è uno standard per ottenere connessioni basate su reti IP sicure. La sicurezza viene raggiunta attraverso la cifratura e l'autenticazione dei pacchetti IP. La sicurezza viene fornita, quindi, a livello di rete. La capacità di fornire protezione a livello di rete rende questo protocollo trasparente al livello delle applicazioni che non devono essere modificate.

IPsec è una collezione di protocolli formata da

- Protocolli che forniscono la cifratura del flusso di dati
- Protocolli che implementano lo scambio delle chiavi per realizzare il flusso crittografato.

Per quanto riguarda il primo aspetto, esistono due protocolli: **Authentication Header (AH)** e **Encapsulating Security Payload (ESP)**. **ESP** fornisce autenticazione, confidenzialità e controllo di integrità del messaggio. **AH**, invece, garantisce l'autenticazione e l'integrità del messaggio ma non offre la confidenzialità; per questo motivo ESP è molto più usato di AH. Attualmente esiste un solo protocollo per lo scambio delle chiavi, il protocollo **IKE**. IPsec è parte integrante di Ipv6, mentre è opzionale in IPv4. Di conseguenza, ci si aspetta che sarà maggiormente utilizzato quando IPv6 acquisterà popolarità. Il protocollo è definito negli RFC 2401-2412. Dal 2004, sono in corso studi per l'aggiornamento dei protocolli.

Scopo del progetto

IPsec è stato progettato per rendere sicure sia comunicazioni *portal-to-portal* sia comunicazioni *end-to-end*. Nella prima configurazione il traffico viene reso "*sicuro*" a diversi computer (in alcuni casi ad un'intera LAN); nella seconda solo i peer che stabiliscono la connessione scambiano pacchetti protetti. Tuttavia l'uso predominante di IPsec è per la creazione di VPN (Virtual Private Network); per conseguire tale scopo possono essere utilizzati entrambi i metodi prima esposti.

Modalità di funzionamento

IPsec supporta due modalità di funzionamento:

- Transport mode
    1. connessione host-to-host
    2. usato dagli end-point non dai gateway
    3. viene cifrato solo il payload dei datagram IP, non l'header
    4. computazionalmente leggero
    5. ogni host che vuole comunicare deve avere tutto il sw necessario di IPsec
    6. si aggiunge solo l'header IPsec; mittente e destinazione si vedono
- Tunnel mode
    1. connessione gateway-to-gateway
    2. viene cifrato tutto il pacchetto IP originale
    3. utilizzato per realizzare le VPN
    4. computazionalmente oneroso
    5. solo i gateway devono avere il sw IPsec
    6. si hanno punti di centralizzazione quindi single point of failure
    7. si ha un doppio incapsulamento, aggiungendo l'header del gateway e l'header IPsec; mittente e destinazione non si vedono

Le due modalità sono supportate sia da AH che da ESP.
IPsec può essere utilizzato anche per connessioni tra gateway e host.

Security Association e Security Policy

Il concetto di **Security Association** (in breve **SA**) è alla base del funzionamento di IPsec. Una SA è un "contratto" fra le due entità coinvolte nella comunicazione; in essa vengono stabiliti i meccanismi di protezione e le chiavi da utilizzare durante il successivo trasferimento dei dati. Nell'ambito di IPsec, stabilire le security association è compito del protocollo IKE, sebbene sia possibile anche impostarle manualmente; ovviamente la procedura manuale è sconsigliata in quanto può introdurre errori che indeboliscono il tunnel. Una peculiarità delle SA è che individuano una comunicazione unidirezionale; quindi durante la creazione della connessione le entità coinvolte creano e gestiscono una SA per ognuno dei versi della comunicazione, quindi 2 SA individuano un canale full-duplex. Al fine di semplificare la gestione delle SA, viene utilizzato un apposito database detto **SAD** (**S**ecurity **A**ssociation **D**atabase), dove viene tenuta traccia delle SA attive.
Una SA è costituita dai seguenti parametri:

- Gli indirizzi IP dei peer coinvolti nella comunicazione
- Il protocollo che verrà utilizzato per il tunnel (AH o ESP)
- le tecniche di cifratura utilizzate e le relative chiavi
- Un intero a 32 bit chiamato **SPI**, acronimo per **S**ecurity **P**arameter **I**ndex

Dall'esame dei parametri di una SA, si deduce che vi sono contenute le informazioni necessarie per stabilire in che modo il traffico debba essere protetto; il passo successivo è definire quale traffico debba essere protetto: di questo si occupa la **S**ecurity **P**olicy (in breve **SP**). Una SP è una regola che stabilisce che tipo di traffico deve essere instradato nel tunnel e quindi essere coperto da IPsec; in modo analogo alle SA, le SP sono contenute in un SPD (**S**ecurity **P**olicy **D**atabase). La security policy contiene:

- Indirizzo sorgente e indirizzo destinazione del pacchetto. Tale informazione è già contenuta nella SA e quindi può sembrare ridondante. In realtà questa informazione ha senso quando viene utilizzato il Tunnel mode.
- Il protocollo e la relativa porta per instradare nel tunnel. Questa opzione dipende dall'implementazione del protocollo e non è sempre contemplata; nel caso non sia disponibile, tutto il traffico prodotto viene veicolato nel tunnel.
- Un identificativo della SA da utilizzare per processare i dati.

Una volta stabilita la security association e la security policy, può cominciare la comunicazione che sfrutterà il protocollo AH o il protocollo ESP cui verrà passato il parametro SPI, che permetterà di risalire alle tecniche crittografiche da utilizzare per la trasmissione.

Protocolli di IPsec

IKE (Internet Key Exchange)

IKE è il protocollo usato per stabilire una **security association.** È un protocollo di livello applicazione e utilizza il protocollo UDP (porta 500) come protocollo di trasporto. L'obiettivo di IKE è stabilire uno *shared session secret*, ossia una chiave condivisa corrispondente alla sessione da instaurare e a tal fine utilizza l'algoritmo Diffie-Helmann; dallo *shared secret* vengono successivamente derivate le chiavi crittografiche che verranno utilizzate per la successiva comunicazione. Al fine di autenticare le entità coinvolte nella comunicazione possono essere

utilizzate tecniche a chiave simmetrica o, alternativamente, a chiave asimmetrica; in quest'ultimo caso si fa ricorso a infrastruttura a chiave pubblica (PKI) e all'uso di certificati digitali.
Si compone di due fasi:

- Prima fase (Main Mode)
  - Autenticazione degli host basata su ip
  - Si usa l'algoritmo Diffie-Hellman per stabilire un secret shared session
- Seconda fase (Quick Mode)
  - A partire da questo canale sicuro si creano altre SA

Authentication Header (AH)

L'**Authentication Header** (**AH**) ha il compito di fornire un controllo di integrità del pacchetto, autenticità del mittente e protezione contro i replay attack. Facciamo notare che AH **non** garantisce in alcun modo la confidenzialità del messaggio. L'autenticità è garantita tramite funzioni di hash a chiave simmetrica, ossia tramite il meccanismo pre-shared keys. Per poter comunicare, due entità devono condividere la medesima chiave; tale chiave viene combinata con il messaggio originale e quindi viene calcolato il checksum tramite una funzione di hash (MD5 o SHA). Il messaggio e il checksum vengono, infine, inviati al peer remoto. Il peer remoto riceve il messaggio; dato che questo è in chiaro, lo può leggere, combinare con la chiave di cui è a conoscenza e calcolare il checksum. Se il checksum corrisponde a quello inviato, il messaggio è autentico e viene accettato altrimenti viene scartato in quanto è stato modificato in un modo non consentito dallo standard. Il protocollo AH è progettato per proteggere l'intero pacchetto IP inviato; tuttavia bisogna considerare che alcuni campi dell'header IP, come il **TTL**, variano durante la trasmissione; queste modifiche devono essere necessariamente consentite, per cui prima di calcolare il checksum, i campi cui è permesso variare vengono posti a **0**.

Formato del pacchetto:

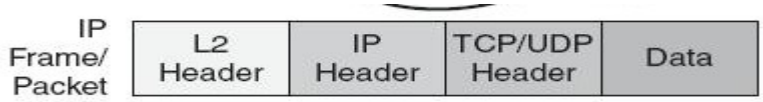
Di seguito viene illustrata la struttura dell'header AH (ogni casella rappresenta 1 byte).

0	1	2	3
Header successivo	Dimensione Payload	RISERVATO	
Security Parameter Index (SPI)			
Numero di Successione			
Dati per l'autenticazione (lunghezza variable)			

- Header successivo: indica che tipo di protocollo verrà dopo.
- Dimensione Payload: Indica la dimensione del pacchetto AH, calcolata in parole di 32 bit.
- RISERVATO: Spazio lasciato per sviluppi futuri. Tutti i bit di questo campo vengono impostati a 0.
- Security Parameter Index: Questo campo identifica i parametri di sicurezza in combinazione con l'indirizzo IP. In genere è un numero pseudo-casuale che identica la security association cui fa parte questo pacchetto.
- Numero di successione: Una successione di numeri monotonicamente crescenti che serve ad impedire i reply attack.
- Dati per l'autenticazione: Contiene le informazioni necessarie ad autenticare i dati.

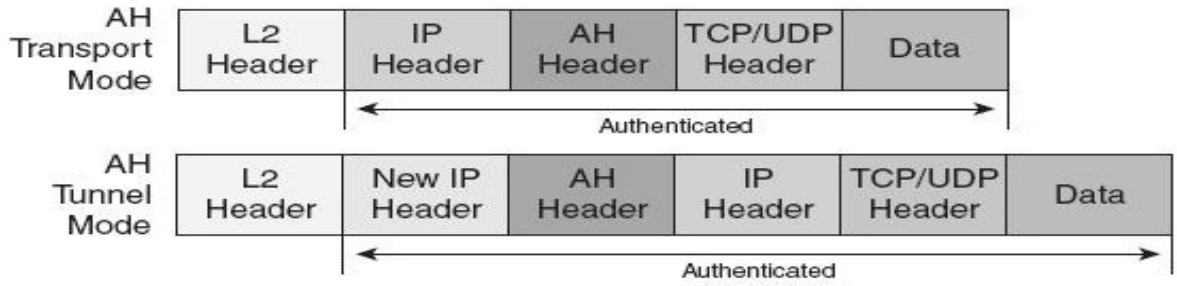
Transport mode e Tunnel mode

AH supporta nativamente sia il transport mode che il tunnel mode. In transport mode vengono protetti solo i protocolli di livello superiore a quello di rete (TCP, UDP, etc); in tunnel mode il pacchetto IP originale viene incapsulato in un nuovo pacchetto IP, dopo essere stato elaborato da AH. Confrontiamo il pacchetto IP originale con quello creato da AH; in presenza di un collegamento basato su IPsec il pacchetto viene, ovviamente, alterato.



Pacchetto IP originale

A seconda della modalità di funzionamento di IPsec (tunnel mode o transport mode), il pacchetto originale viene alterato in modo diverso.



Da quanto appena detto si evince subito che il protocollo AH è incompatibile con le varie tecniche di NAT; difatti se vengono alterati i campi indirizzo nell'header IP (in entrambe le modalità), in ricezione la checksum fallisce subito.

Encapsulating Security Payload (ESP)

Encapsulating Security Payload, è un protocollo che fa parte della suite IPsec. Il suo obiettivo è fornire confidenzialità e controllo di integrità e autenticità alla comunicazione. Contrariamente a quanto fa AH, l'header IP non viene coperto dai controlli. Al pari di AH, però, supporta sia il tunnel mode che il transport mode.

Formato del pacchetto

Di seguito viene riportato il formato dell'header **ESP** (ogni casella rappresenta 1 byte).

0	1	2	3
Security Parameters Index (SPI)			
Sequence Number			
Payload * (variable)			
Padding (0-255 byte)			
Pad Length			Next Header
Authentication Data (variable)			

- Security Parameters Index (SPI): Al pari di quanto avviene in AH, questo campo, in combinazione con l'indirizzo IP, individua la Security Association cui appartiene il pacchetto.
- Sequence Number: Una successione di numeri monotonicamente crescente, che identifica il

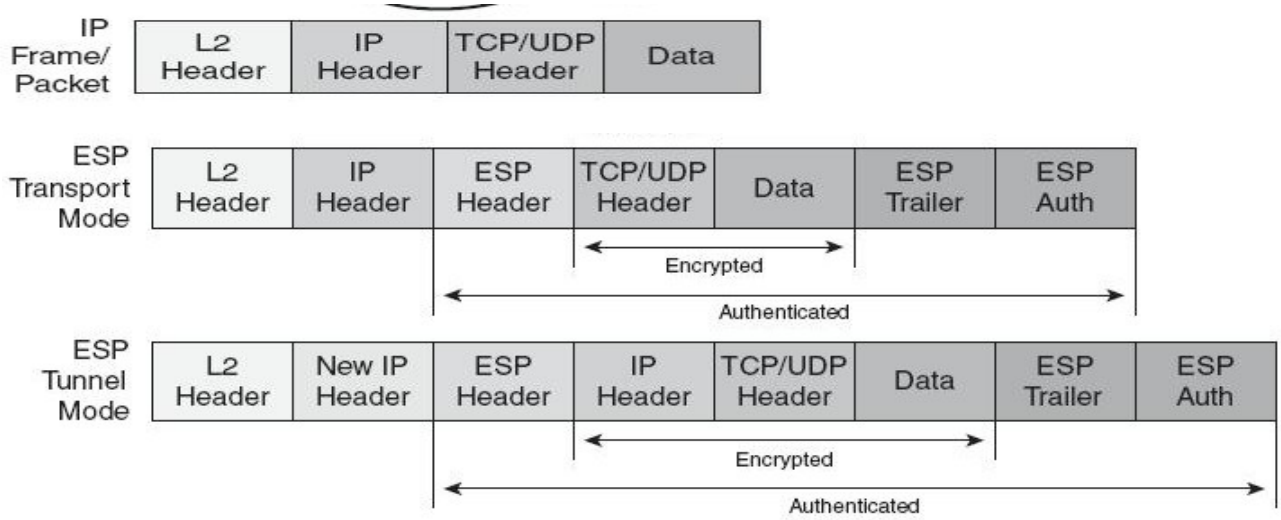
pacchetto all'interno delle Security Association e previene da reply attack

- Payload: I dati che devono essere trasferiti
- Padding: È un campo di riempimento. È necessario in quanto alcuni codici di cifratura lavorano su blocchi di lunghezza fissa. Serve a far crescere la dimensione dei dati fino a divenire multiplo del blocco che l'algoritmo in uso riesce a gestire.
- Pad Length: Rappresenta, in bit, la dimensione dei dati di padding aggiunti.
- Next Header: Identifica il protocollo dei dati trasferiti
- Authentication Data: Contiene i dati usati per autenticare il pacchetto.

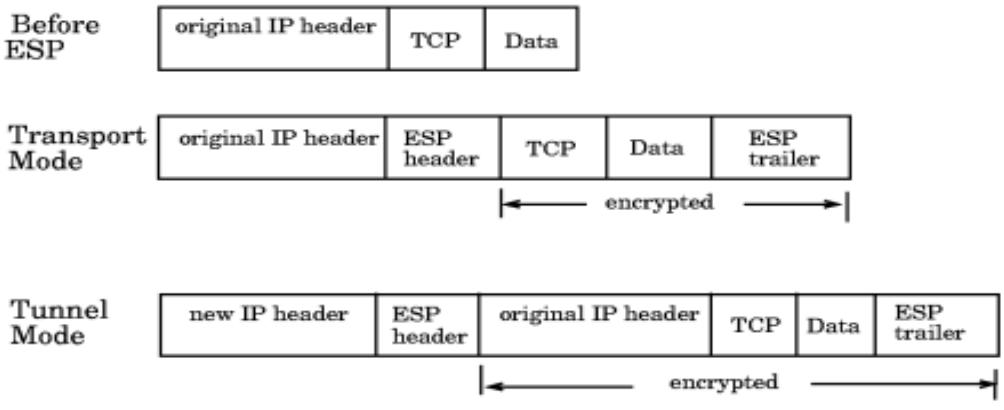
Come si può vedere dalla struttura del pacchetto (ma sarà illustrato meglio in seguito), ESP "*avvolge*" i dati dei protocolli di livello superiore, contrariamente a quanto fa AH che antepone un header.

Tunnel mode e Transport mode

Essendo un protocollo per il trasferimento dati della suite IPsec, ESP supporta sia il Tunnel mode che il Transport mode. Confrontiamo il pacchetto IP originale con quello elaborato con ESP.



Quando si usa l'autenticazione un valore di autenticazione è calcolato per i dati crittografati utilizzando una chiave simmetrica e inserito alla fine del pacchetto (ESP Auth) similmente a come avviene in AH. Altrimenti se non si utilizza l'autenticazione:

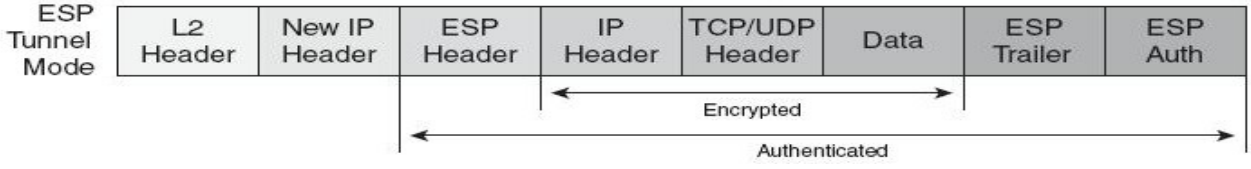


Per quanto riguarda gli algoritmi di cifratura possono essere utilizzati Data Encryption Standard (DES), 3DES, AES e Blowfish. Il controllo di integrità e autenticità viene eseguito tramite HMAC (funzioni di hash); l'hash viene calcolato tramite una funzione di hash (MD5 o SHA-1), utilizzando

una chiave condivisa; l'hash ottenuto viene allegato al messaggio e inviato. In ricezione viene controllata l'integrità del messaggio. Dagli schemi visti prima si nota che l'indirizzo IP più esterno non viene coperto dal controllo di integrità. Tale opzioni rende il protocollo ESP adatto ad essere utilizzato in alcuni tipi di NAT, in particolare in quelli statici. Tuttavia esistono soluzioni ad-hoc per il funzionamento congiunto di IPsec e NAT come il NAT Traversal.

NAT Traversal

NAT traversal (o più in breve NAT-T) è il nome di un protocollo facente parte della suite Ipsec. L'obiettivo di questo protocollo è fornire la possibilità di stabilire un tunnel IPsec anche quando uno dei due *peer* coinvolti subisce un'operazione di nat per raggiungere l'altra entità coinvolta nella comunicazione.
Il NAT è una tecnica molto utilizzata per il riuso degli indirizza IP. Tuttavia gli host dietro un router (o un firewall) che effettua operazioni di NAT non godono di connettività end-to-end. Sebbene esistano diversi tipi di NAT, l'obiettivo generale è l'alterazione degli header del pacchetto. Questo comportamento è in netto contrasto con IPsec che ha tra i suoi obiettivi il controllo dell'**integrità** del pacchetto. In particolare il NAT è incompatibile con AH sia in tunnel mode che in transport mode, in quanto AH verifica l'integrità di tutto il pacchetto IP. ESP, invece, non copre l'header IP con controlli di sorta né in Tunnel mode né in Transport mode, per cui risulta adatto nel caso in cui il nat eseguito sia di tipo SNAT (statico); in altre parole, la modifica apportata dal router deve coinvolgere **solamente** l'header IP e non anche la porta del livello superiore.
Il NAT crea problemi anche con IKE in quanto esso richiede l'autenticazione degli host coinvolti nella comunicazione e tale autenticazione prevede un controllo sugli indirizzi IP; per cui l'alterazione dell'indirizzo da parte di un'apparecchiatura di NAT provoca il fallimento dell'autenticazione.
In genere, nei dispositivi preposti alla gestione dei tunnel IPsec e nei client VPN, il NAT-T non è abilitato di default ma deve essere impostato a mano; tuttavia il suo utilizzo rimane opzionale: difatti durante la creazione della **security association**, i peer determinano se uno dei due subisce operazioni di NAT e solo in questo caso viene usato il NAT-T; questa operazione viene fatta durante la prima fase della negoziazione **IKE**. In prima battuta, i *peer* verificano che entrambi siano in grado di supportare in NAT-T; Una volta stabilito che entrambi supportano il NAT-T, vengono inviate delle frame "NAT-Discovery" (NAT-D), in modo da verificare chi dei due subisca il NAT, o al limite se lo subiscano entrambi. Una volta stabilito chi subisce il NAT, la comunicazione si sposta su una nuova coppia di porte UDP e l'entità "*nattata*" comincia a inviare delle frame **keepalive**; queste frame servono a mantenere fisse le porte di comunicazione sul router e ad impedirgli di riassegnarle ad una nuova comunicazione.
Descriviamo come viene incapsulato il pacchetto originale ESP.



L2 Header	New IP Header	Header TCP/UDP	Header NAT-T	Header ESP	Header IP	Header TCP/UDP	Data	ESP Trailer	ESP auth

ESP in Tunnel mode con NAT-T

I campi segnati in grigio scuro sono quelli relativi al NAT-T; questi campi vengono inseriti subito dopo l'header IP esterno, che non viene alterato, così come non vengono alterati i campi successivi. In ricezione viene fatta l'operazione inversa.

SSL/TLS (Secure Socket Layer/Transport Layer Security)

L'SSL è uno standard creato dalla Netscape Corporation per garantire la sicurezza nel Web. La versione 3, attualmente, è la base per un altro standard sotto sviluppo, TLS, che quindi non è ancora formalmente adottato.

SSL lavora in termini di connessione e sessione tra client e server.

SSL Session: è un'associazione tra due peer.
SSL Connection: è un insieme di meccanismi usati per trasportare dati in una sessione.

Una singola sessione può avere più connessioni. Due peer possono avere più connessioni attive allo stesso momento, anche se questo non è comune. I dati associati ad una sessione hanno le seguenti caratteristiche:

- session id
- certificato x.509v3 dei peer
- metodo di compressione per ridurre il volume dei dati
- cipher specification: include tutti i parametri rilevanti per i meccanismi di cifratura e dei codici di autenticazione dei messaggi (MAC).
- "master secret": 48 byte condivisi

Una connessione descrive come vengono inviati e ricevuti i dati dai peer. Le informazioni associate alla connessione sono le seguenti:

- dati
- le chiave del server e del client usate per la cifratura
- la chiave MAC, usata per calcolare il MAC.
- Vettore di inizializzazione per il cifrario (se necessario)
- client e server sequence numbers

SSLv3 consiste in due livelli che supportano numerosi meccanismi di sicurezza:

- Lower Level:
  - SSL record Protocol
- Upper Level:
  - SSL Handshake Protocol
  - SSL Change Cipher Spec Protocol
  - SSL Alert Protocol
  - SSL Application Data Protocol

Meccanismi di crittografia supportati

Durante la fase iniziale della comunicazione, i peer determinano:

- il meccanismo di crittografia per mantenere confidenzialità e integrità (classical Cipher)
- Un meccanismo di tipo interchange keys viene usato per stabilire la chiave di sessione
- L'algoritmo MAC

Se l'*interchange cipher è RSA*, il server deve avere un certificato RSA per lo scambio, avremo le seguenti combinazioni di cifrari classici e algoritmi per MAC.

Interchange cipher	Classical cipher	MAC algorithm
RSA (chiave di 512 bit)	No	MD5, SHA
	RC4 (40 bit key)	MD5
	RC2 (40 bit key), CBC mode	MD5
	DES (40 bit key), CBC mode	SHA
RSA	No	MD5, SHA
	RC4 (128 bit key)	MD5, SHA
	IDEA, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA

Se l'*interchange cipher è Diffie-Hellman,* bisogna distingure tre casi:

- *Diffie-Hellman*: denota il sistema in cui il certificato contiene i parametri di cifratura ed è firmato da una CA.
- *Ephemeral Diffie-Hellman:* si riferisce ad un sistema in cui il certificato DSS o RSA viene usato per firmare i parametri di cifratura.
- *Anonimous Diffie-Hellman:* si riferisce all'uso di Diffie-Hellman senza autenticazione. Molto vulnerabile e fortemente sconsigliato.

Le possibili combinazioni sono:

Interchange cipher	Classical cipher	MAC algorithm
Diffie-Hellman, DSS certificate	DES (40 bit key),CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA
Diffie-Helmann (key <=512 bit)	DES (40 bit key), CBC mode	SHA
RSA certificate	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA
Ephemeral Diffie-Hellman	DES (40 bit key),CBC mode	SHA
DSS certificate	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA
Ephemeral Diffie-Hellman	DES (40 bit key),CBC mode	SHA
(key <=512 bit)	DES, CBC mode	SHA
RSA certificate	DES, EDE mode, CBC mode	SHA
Anonymous Diffie-Hellman	RC4 (40 bit key)	MD5
	RC4 (128 bit key)	MD5
	DES (40 bit key),CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA

Se l'*interchange cipher è Fortezza*, usato dal DoD (USA Department of Defense), la tabella è la seguente:

Interchange cipher	Classical cipher	MAC algorithm
Fortezza	None	SHAS
	Fortezza, CBC mode	SHA
	RC4 (128 bit key)	SHA

RC4: L' **RC4** (Rivest Cipher) è uno tra i più famosi e diffusi algoritmi di cifratura a chiave simmetrica, utilizzato anche nel protocollo WEP. Nonostante la sua diffusione, è un algoritmo facilmente violabile ed il suo uso è caldamente sconsigliato se il livello di sicurezza ricercato deve essere elevato. L'RC4 genera un flusso di bit pseudo casuali (keystream 256 byte): tale flusso è combinato mediante un'operazione di XOR con il testo in chiaro per ottenere il testo cifrato. L'operazione di decifratura avviene è l'inverso (questo perché lo XOR è un'operazione simmetrica).

IDEA: L'IDEA è un algoritmo di cifratura a blocchi simmetrico, per cui necessita di una unica chiave per cifrare e per decifrare. Opera su blocchi di dati di 64 bit utilizzando una chiave di 128 bit effettuando una serie di 8 passaggi identici (definiti in inglese *round*) durante i quali vengono eseguite le seguenti operazioni, tutte su numeri a 16 bit:

- XOR
- addizione mod 2^{16}
- moltiplicazione mod $2^{16} + 1$

CBC mode: tecnica in cui lo stesso blocco di testo in chiaro, se ripetuto, produce blocchi di testo cifrato differenti

Lower Level: SSL Record Protocol

L'SSL record protocol fornisce una base sicura di comunicazione. Il protocollo fornisce confidenzialità e integrità del messaggio inviato. Esso accetta messaggi dai livelli superiori. Gli step sono:

- Ogni messaggio viene diviso, se necessario, in più parti. (SSL trasporta blocchi di grandezza al massimo $2^{14} = 16384$ byte).
- I singoli blocchi vengono compressi
- Viene calcolato il MAC (hash) dei blocchi compressi
- Si cifra MAC + blocco
- Si preparano gli header.
  Gli header contengono:
  - il tipo di messaggio
  - numero di versione principale (3 per SSLv3 e TLS)
  - numero di versione minima (0 per SSLv3, 1 per TLS)
  - lunghezza del blocco

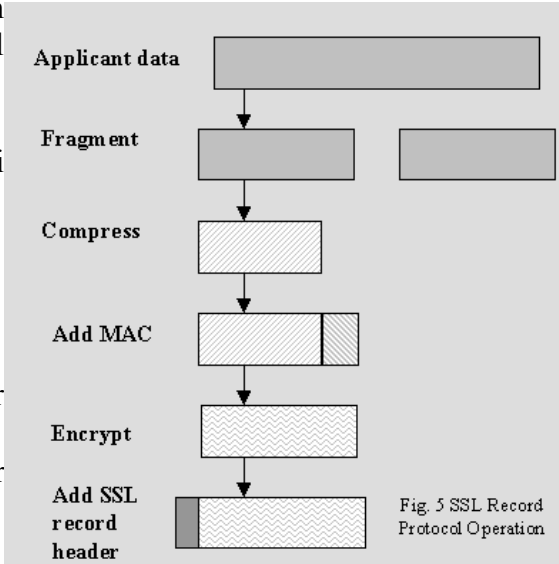


Fig. 5 SSL Record Protocol Operation

Upper Level: SSL Handshake Protocol

L'SSL Handshake Protocol stabilisce una comunicazione. Si compone di quattro fasi di fasi che permettono ai peer di accordarsi su chiave, cifrario e algoritmo per l'hash. Il numero delle fasi e

l'esatta sequenza con cui vengono scelti i parametri dipendono dal livello di sicurezza desiderato. Per semplicità vedremo come funziona nel caso in cui come interchange criptosystem si usa RSA.

---- **PRIMA FASE (Richiesta)** ----------

1. Inoltro richiesta di creazione della connessione SSL tra il client C e il server S.

   C → S : { version, N1, session_id, cipher_list, compression_list }

   - *version:* versione SSL del client
   - *N1:* nonce
   - *session_id:* che è zero se si cerca di stabilire la connessione altrimenti è il numero di sessione stabilita
   - *cipher list:* lista dei cifrari che il client conosce
   - *compression list:* lista degli algoritmi hash che il client conosce

2. Risposta alla richiesta

   S → C : { version, N2, session_id, cipher, compression }

   - *version:* versione massima che possono utilizzare entrambi
   - *N2:* nonce
   - *session_id:* numero di sessione assegnato
   - *cipher:* cifrario scelto (assumiamo RSA)
   - *compression:* algoritmo hash scelto

---- **SECONDA FASE (Identificazione)** ----------

3. Il server si identifica

   S → C : {server_cert}

   - *server_cert:* certificato X.509v3 del server

4. Invio chiave pubblica

   $S \rightarrow C : \{ mod, exp, \{hash(N1, N2, mod, exp)\}Ks^{-1} \}$

   - *hash:* concatenazione di MD5 e SHA-1

5. Il server richiede al client di identificarsi

   S → C : { cert_type, good_CA }
   - *cert_type:* tipo certificato richiesto
   - *good_CA:* lista CA accettate dal server

6. Fine seconda fase

   S → C : { end_round_2 }

---- **TERZA FASE (Master Secret)** ----------

7. Invio certificato del client

   $C \rightarrow S : \{ client\_cert \}$

8. Invio pre master secret

   $C \rightarrow S : \{ pre \}$

   - *pre:* 48 bit random cifrati con la chiave pubblica del server

(Entrambe le parti calcolano, a partire da pre, il master secret)

9. Invio del master secret

   $C \rightarrow S : \{ hash(master, opad, hash(messages, master, ipad)) \}$

   - *opad/ipad:* parametri dell'algoritmo hash HMAC
   - *messages:* concatenazione dei messaggi da 1 a 8.
   - *master:* master secret calcolato

---- **QUARTA FASE (Change Cipher Spec)** ----------

10. Aggiornamento della procedura

    $C \rightarrow S : \{ hash(master, opad, hash(messages, master, ipad)) \}$

11. Risposta

    $S \rightarrow C : \{ hash(master, opad, hash(messages, master, ipad)) \}$

12. Invio del Change Cipher Spec

    $S \leftrightarrow C : \{ 11111111 \}$

A questo punto Client e Server possono comunicare.

Upper Level: SSL Change Cipher Spec Protocol

Consiste nell'inviare un singolo byte (tutti 1). Viene inviato dopo che è avvenuta la negoziazione o rinegoziazione dei parametri di cifratura.

Upper Level: SSL Alert Protocol

Segnala che esiste una condizione non usuale. Esisto due tipi di alert:

- *closure alert:* indica che colui che lo manda non invierà più alcun messaggio
- *error alert:*
  - *fatal:* ne consegue la chiusura della connessione (decompression_failure, handshake_failure ecc...)
  - *warnings:* non si chide la connessione (no_certificate, certificate_expired/revoked)

Upper Level: SSL Application Data Protocol

Effettua il passaggio dal livello applicazione all'SSL Record Protol Layer.

DNSSEC (DNS Security Exstensions)

Il **Domain Name System Security Extensions (DNSSEC)** è una suite di IETF specifiche per garantire la sicurezza di informazioni fornite dal DNS, usato su reti IP. Si tratta di un insieme di estensioni del DNS che forniscono ai clienti (resolver):

- Autenticazione del DNS
- L'integrità dei dati (ma non la disponibilità o riservatezza)

E 'opinione diffusa che proteggere il DNS è di fondamentale importanza per garantire Internet nel suo complesso. L'originale design del DNS non comprende la sicurezza. Il DNSSEC tenta di aggiungere la sicurezza, pur mantenendo la compatibilità con il vecchio sistema. DNSSEC è stato progettato per proteggere i resolver da dati falsi ottenuti da di DNS falsi, come quelli creati da DNS cache poisoning. Tutte le risposte in DNSSEC sono firmati digitalmente. Con la verifica della firma digitale, un resolver DNS è in grado di verificare se le informazioni sono corrette e complete. DNSSEC non prevede la riservatezza dei dati. DNSSEC non protegge contro attacchi DoS direttamente, anche se indirettamente, fornisce alcuni benefici.

DNSSEC lavora firmando digitalmente le risposte, usando la crittografia a chiave pubblica. Per effettuare questa operazione, diversi nuovi tipi di record DNS sono stati creati: RRSIG, DNSKEY, DS e NSEC. Quando viene utilizzato DNSSEC, ogni risposta ad una ricerca conterrà il record RRSIG, in aggiunta ai tipi di record che sono stati richiesti. Il record RRSIG contiene la firma digitale della risposta. La firma digitale può essere verificata tramite la corretta chiave pubblica in un DNSKEY record. Un resolver DNS può quindi stabilire se le risposte che ha ricevuto sono corrette, se il nome del server autorevole per il dominio richiesto non supporta DNSSEC, o se vi è una sorta di errore. Il record DNSKEY è trovato tramite una *Authentication Chains*, a partire da una chiave pubblica fidata *Trust Anchor*. Questa chiave pubblica può quindi essere utilizzata per verificare una *designated signer* (DS). Un DS record di un dominio padre può essere utilizzato per verificare una DNSKEY di un sottodominio, che possono quindi contenere altre DS record per verificare ulteriormente i sottodomini.

Processo

Quando il resolver riceve una risposta tramite il normale processo di ricerca, la controlla per assicurarsi che la risposta è corretta. In teoria, il resolver dovrebbe iniziare con la verifica del DS e DNSKEY record del DNS root. Poi si verifica il DS del dominio di primo livello (com, it ecc.) trovato nella root, e lo si usa per verificare il record DNSKEY della zona. Alla fine di questo processo ricorsivo, si verifica la RRSIG record trovati nella risposta ottenuta dal DNS autoritativo, per il dominio che stiamo cercando.

Ci sono diversi casi di eccezione:

- Se il dominio che stiamo cercando non supporta DNSSEC, non sarà possibile ottenere un RRSIG valido nella risposta e non ci sarà un DS record per per quel dominio.
- Se c'è il record DS ma non si ottiene nella risposta un RRSIG valido, si può intuire che si è verificato un attacco (man in the middle attack, errore di configurazione ecc...).
- È possibile, anche, che il dominio che stiamo cercando non c'è. Invece di restituire un record RRSIG nella risposta, si ottiene un NSEC o NSEC3 record. Tali record hanno, comunque,

un RRSIG record, che può essere verificato come sopra.

- Può accadere che il dominio implementa DNSSEC ma non lo implementano ne il DNS di primo livello ne la root. Si crea un isola di sicurezza che deve comunque essere verificata in altro modo.

Trust Anchors and Authentication Chains

Al fine di essere in grado di dimostrare che una risposta è corretta, è necessario conoscere almeno una chiave che è corretta. Questi punti di partenza sono noti come *Trusted Anchors* e sono in genere ottenuti tramite sistema operativo o tramite un altra fonte affidabile. Quando DNSSEC è stato concepito, si pensava che l'unico Trust Anchor necessario fosse il DNS root. Tuttavia, nella radice DNSSEC non è stato reso operativo, il che ha portato allo sviluppo di fonti di fiducia alternative. Una *Autentication Chains* è legata ad una serie di DS e DNSKEY record, a partire da un Trusted Anchor dell'authoritative name server per il dominio in questione. Senza una completa catena di autenticazione, una risposta ad una ricerca DNS non può essere autenticata.

DIFFIE-HELLMAN:

Alice e Bob scelgono p = 53 e g = 17

$K_{alice}^{-1} = 5$ $\quad\quad$ $K_{Bob}^{-1} = 7$

$K_{alice} = 17^5 \bmod 53 = 40$ $\quad\quad$ $K_{Bob} = 17^7 \bmod 53 = 6$

Bob → Alice

$S_{BobAlice} = K_{alice}^{Kbob-1} \bmod 53 = 40^7 \bmod 53 = 38$

Alice → Bob

$S_{AliceBob} = K_{bob}^{Kalice-1} \bmod 53 = 6^5 \bmod 53 = 38$

RSA:

Si scelgono due numeri primi molto grandi *p* e *q* e calcolato $n=pq$. $\phi(n)$ è il numero di numeri più piccoli di n primi con n.
Poi si sceglie un intero $e < n$ che deve essere primo con $\phi(n)$. Scegliamo un secondo intero *d* tale che $ed \bmod \phi(n)=1$. Fatto questo la chiave pubblica sarà (e,n) e la privata *d*.

Cifrare → $c = m^e \bmod n$

Decifrare → $m = c^d \bmod n$