INTRODUZIONE---------- 1

CAP I: LO STANDARD IKE / IPSEC---------- 3

I.1 STORIA DELL' IPSEC----------4

I.2 POSIZIONAMENTO ALL'INTERNO DELLO STACK TCP/IP ----------5

I.3 IPSEC ----------6

I.3.1 I Servizi offerti..... 6

I.3.2 I Database..... 8

I.3.3 I Protocolli..... 8

I.3.4 Il protocollo AH..... 10

I.3.5 Il protocollo ESP..... 14

I.3.6 Collocazione..... 18

I.4 IKE – INTERNET KEY EXCHANGE ---------- 20

I.4.1 Introduzione..... 20

I.4.2 Shred secret..... 21

I.4.3 Le due fasi..... 23

I.4.4 La fase uno..... 26

I.4.5 La fase due..... 28

I.4.6 Altri scambi..... 30

I.5 ISAKMP---------- 31

I.5.1 Introduzione..... 31

I.5.2 Fase di Autenticazione..... 31

I.5.3 Requisiti di ISAKMP..... 32

I.5.4 ISAKMP Header..... 33

I.5.4 ISAKMP Generic Payload Header..... 34

I.5.5 Identification Payloads..... 35

CAP II: PUBLIC KEY INFRASTRACTURE ---------- 38

II.1 CERTIFICATI DIGITALI---------- 39

II.1.1 Lo Standard X.509 per i certificati..... 41

II.2 PKI – INFRASTRUTTURE A CHIAVE PUBBLICA ---------- 48

II.2.1 Fase di Registrazione..... 53

II.2.2 Fase di Inizializzazione..... 54

II.2.3 Fase di Certificazione..... 55

II.3 CICLO DI VITA DEI CERTIFICATI ---------- 56

II.6.2 Rrevoca e Sospensione di un certificato 57

II.4 CERTIFICATION PATHS ---------- 60

CAP III: INTEROPERABILITÀ - I TEST ---------- 64

III.1 SCOPI ---------- 65

III.2 DESCRIZIONE DELL' AMBIENTE DI TEST ---------- 66

III.2.1 I Protagonisti 66

III.2.2 Obiettivi 67

III.2.3 Strumenti di analisi 69

III.2.4 Piattaforme e configurazioni base 70

III.3 PROCEDURE DI TEST ---------- 77

III.3.1 ID Type 77

III.3.2 Rispetto dei TOL 81

III.3.3 Politiche e rispetto delle priorità 82

III.4 CERTICATE ENROLLMENT ---------- 85

III.4.1 La gestione dei certificati nel GAIA4 85

III.4.2 La gestione dei certificati nel CISCO 88

III.5 RISULTATI ---------- 91

III.5.1 Fase I MAIN MODE 93

III.5.2 Fase I AGGRESSIVE MODE 100

III.5.3 Commenti 103

CONCLUSIONI ---------- 104

APPENDICE A: CRITTOGRAFIA ---------- 106

A.1 CRITTOGRAFIA SIMMETRICA E ASIMMETRICA ---------- 107

A.2 FIRMA DIGITALE ---------- 109

A.3 ALGORITMI ASIMMETRICI ---------- 111

A.3.1 Diffie Helmann 111

A.4 ALGORITMI PER LA FIRMA DIGITALE ---------- 113

A.4.1 RSA 113

A.5 ALGOROTMI PER LE FUNZIONI DI HASH ---------- 114

A.5.1 Definizione funzioni di Hash 114

A.5.2 SHA-1 116

ACRONIMI ---------- 117

BIBLIOGRAFIA ---------- 120

INTRODUZIONE

La comunicazione è lo scambio di informazioni tra due o più persone o enti; affinché l'effetto della comunicazione sia quello desiderato, però, l'informazione deve essere accessibile esclusivamente a coloro che ne hanno diritto.

Il bisogno dello scambio sicuro di informazioni nasce infatti con l'uomo e assume un'importanza molto rilevante nell'epoca della rivoluzione della comunicazione.

Oggi le telecomunicazioni ci permettono di percorrere distanze smisurate con un semplice "click", di inviare e ricevere informazione nelle più disparate forme. Per far ciò, le nostre telefonate, le nostre e-mail…le nostre informazioni attraversano satelliti, catene di elaboratori elettronici e le occasioni di intercettazione diventano innumerevoli e incontrollabili. La crittografia è il solo strumento in grado di garantire la riservatezza di tutti e il successo telematico. È compito di questa disciplina fornire servizi di sicurezza.

Con l'ampliarsi dei sistemi di telecomunicazioni cresce l'esigenza della sicurezza dei dati, intesa come:

- Autenticazione
- Integrità
- Riservatezza

Questa tesi è incentrata principalmente sullo studio dei protocolli di sicurezza che compongono lo standard IKE / IPSec e delle infrastrutture a chiave pubblica ed è volta a verificare l'interoperabilità tra apparati di aziende differenti che implementano i servizi di sicurezza messi a disposizione dalla suite di protocolli IKE / IPSec.

Il lavoro di ricerca e sviluppo necessario per conseguire gli obiettivi di questa tesi è stato completato all'interno delle strutture aziendali della ELSAG S.p.A., una società del gruppo Finmeccanica. In particolare sono stati coinvolti i laboratori di

Security System Integration e *Sviluppo* dell'AMTEC, società appartenente alla divisione Sistemi e Servizi di Sicurezza di ELSAG S.p.A..

L'azienda ha mostrato un grande interesse nell'opportunità di sottoporre a test il software GAIA4 implementato sul proprio prodotto di sicurezza, il router SAS 862.

L' interoperabilità tra il SAS 862 ed altri apparati già presenti e molto diffusi sul mercato è, per l'azienda, un obbiettivo di primaria importanza per un più rapido inserimento del proprio router nel mercato stesso.

Il lavoro è stato strutturato in 3 parti:

- Una presentazione della suite IKE/IPSec: sono illustrati i protocolli principali che la compongono, i principi che sono alla loro base e gli schemi funzionali.
- Uno studio approfondito delle Public Key Infrastucture (PKI) e dei Certificati Digitali: viene illustrata la struttura generale delle PKI e lo standard X.509v3 per i certificati digitali
- Una progettazione, seguita dall'esecuzione, di un piano di testing: vengono presentate le metodologie di test, raccolti e commentati i risultati. Particolare attenzione è stata rivolta ai principi di funzionamento degli apparati coinvolti nei test, allo scopo di meglio comprendere le eventuali incompatibilità, e ove fosse il caso, di avere uno strumento preciso per identificare e risolvere il problema.

CAP I: LO STANDARD IKE / IPSEC

La suite di protocolli che costituisce l'ambiente IPSec ha l'obiettivo di fornire un sistema di sicurezza delle comunicazioni, basato sui datagrammi IP. L'idea alla base di IPSec è quella di rendere sicura l'intera rete creando un sistema che protegga attraverso metodi crittografici i datagrammi IP. IPSec differisce da altri protocolli indirizzati ad assicurare protezione a livello di sessione/presentazione, come ad esempio SSH e SSL, grazie ai quali le comunicazioni tra due processi che colloquiano tramite un canale non sicuro, vengono codificate e rese inaccessibili a tutti tranne che alle controparti autorizzate. L'utilizzo più frequente di IPSec è per creare "Secure Virtual Private Network (S-VPN)", queste reti virtualmente private interconnettono sottoreti dislocate geograficamente per mezzo di una rete pubblica, come ad esempio la rete Internet. Da qui l'esigenza di proteggere il traffico già a livello IP. La diffusione di IPSec dipende direttamente dalla diffusione delle S-VPN

I.1 Storia dell' IPSec

L'IPsec è il risultato di uno dei gruppi di lavoro dell'Internet Engineering Task Force (IETF[1]) iniziato negli anni ottanta e culminato nella pubblicazione della prima Request For Comment (RFC-1825) nel 1995. Negli anni a seguire il lavoro del gruppo ha poi continuato a perfezionare l'IPSec fino ad una stesura e dunque pubblicazione di una nuova serie di RFC, in particolare le 2401, 2402 e 2406 nel novembre del 1998.

Da quel momento l'IPSec è stato considerato standard, e basandosi su queste ultime RFC, è iniziata l'opera di implementazione su i vari apparati di sicurezza.

Con il diffondersi delle reti intranet aziendali, e con la crescente sensibilità ai problemi di sicurezza, particolare interesse raccoglie oggi l'IPSec. Ad esempio queste reti virtualmente private mettono in comunicazione sedi diverse della stessa impresa, consentendo la comunicazione tra le sottoreti dei singoli edifici attraverso l'utilizzo della rete pubblica. Appare chiaro che una comunicazione che transiti su di una rete condivisa con altri utenti necessiti di un elevato grado di sicurezza.

Il grande diffondersi dell'IPSec ha portato l'IETF a continuare il proprio lavoro di ricerca, allo scopo di migliorare le prestazioni della suite di protocolli, aggiornando le funzionalità di sicurezza per elevarne il livello e soprattutto per gestire reti sempre più complesse.

Oggi il gruppo ha al suo attivo ben 21 RFC, e altrettante Internet-Draft. Parallelamente sono nati altri working group correlati, come l'IPSP e l'IPSRA. Il primo interessato allo studio degli strumenti necessari alla propagazione delle Policy, argomento di particolare interesse per rendere più dinamica e scalabile la attuale struttura delle S-VPN. Il secondo a individuare metodologie per il Remote Access, principalmente allo scopo di consentire l'accesso alla S-VPN anche per dispositivi portatili.

[1] www.IETF.org

I.2 Posizionamento all'interno dello stack TCP/IP

In IPSec i servizi di cifratura e di autenticazione si completano a livello IP, da ciò nasce l'esigenza e la possibilità di integrare la suite IPSec all'interno di apparati come router o gateway. Una rete che abbia come punto di accesso un security gateway potrà usufruire dell'IPSec per il traffico uscente indirizzato ad un'altra rete protetta anch'essa da un security gateway. Questo tipo di configurazione è tipica delle VPN, reti virtualmente private in cui il traffico viaggia su una rete IP pubblica allo scopo di mettere in comunicazioni due o più reti private, tipicamente due sezioni di una stessa intranet.

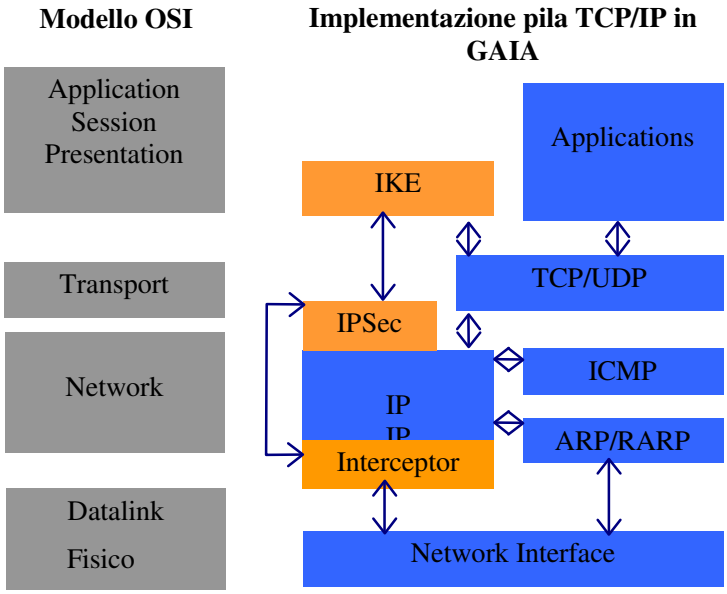

Figura I.1: Posizionamento nello stack TCP/IP

Rispetto allo stack TCP/IP, IPSec si colloca al di sopra di IP ed in parallelo ai protocolli superiori come il TCP o l'UDP. Il protocollo di scambio di chiavi "Internet Key Exchange (IKE)" si posiziona al di sopra del protocollo di trasporto UDP che utilizza per la proprie segnalazione, allo stesso tempo si posiziona logicamente in parallelo allo strato applicazione. Rispetto ad IPSec, IKE è un protocollo di tipo applicativo che viene utilizzato per la negoziazione delle sessioni sicure.

I.3 IPSec

I.3.1 I Servizi offerti

Lo standard IPSec definisce meccanismi di sicurezza specifici per i datagrammi IP sia nella versione IPv4 che IPv6. IPSec può essere implementato nei singoli host oppure in sistemi intermedi detti security gateway, fornisce essenzialmente servizi di autenticazione, integrità e riservatezza; dato che tali servizi sono forniti a livello IP possono essere utilizzati da qualunque protocollo di livello superiore in modo totalmente trasparente. Inoltre fornisce: controllo di accesso, integrità senza connessione, protezione contro le risposte false.

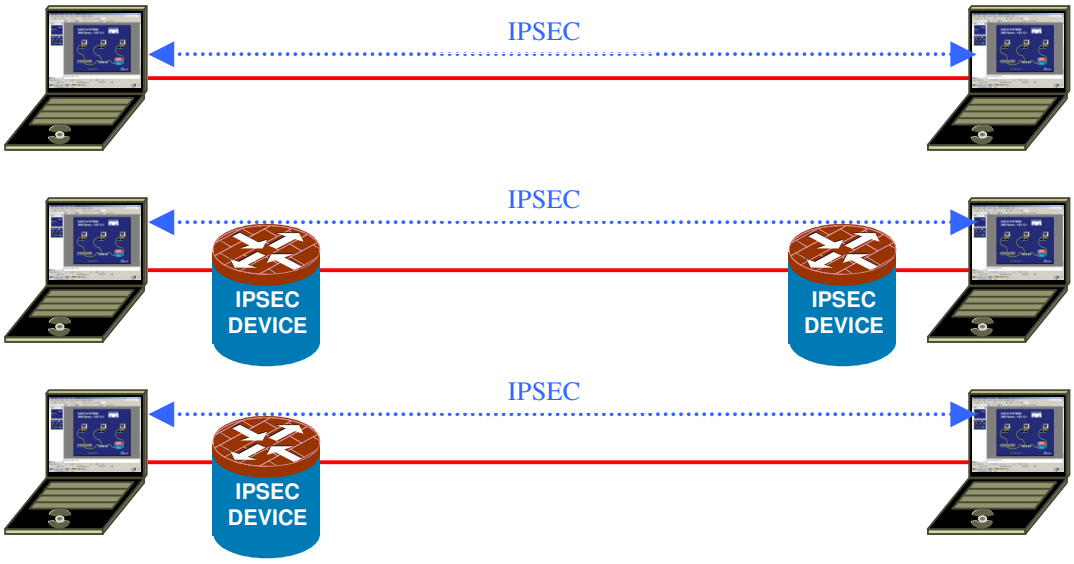

Figura I.2: diverse implementazioni: host to host, gateway to gateway, host to gateway

Questi servizi sono realizzati tramite l'instaurazione di tunnel sicuri fra due entità di pari livello, ovvero fra due host, fra due gateway di sicurezza o fra un host e un gateway di sicurezza (Figura I.2).

Una volta definite le caratteristiche di questi tunnel, risultano noti i criteri di selezione del traffico e le modalità (algoritmi) di protezione di tale traffico. In questo modo, quando un gateway di sicurezza riceve un pacchetto che soddisfa certi criteri di filtraggio (in genere fa parte di una certa lista di accesso), il gateway stabilisce un appropriato tunnel di sicurezza e invia il pacchetto ricevuto attraverso tale tunnel verso l'entità di pari livello destinataria (Figura I.3)

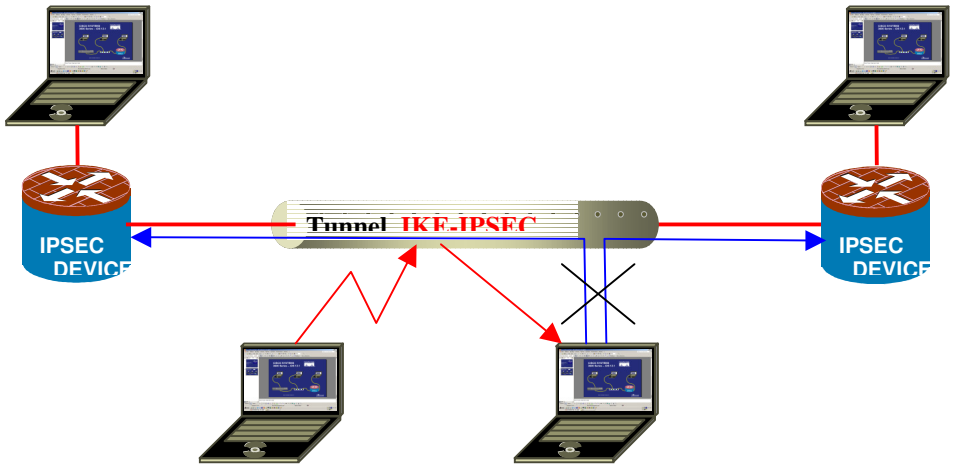

Figura I.3: instaurazione di un tunnel IPSec

In particolare i servizi offerti da IPSec si differenziano in due modalità: transport e tunnel. Come illustrato nella figura I.4, il transport mode si può instaurare esclusivamente tra due host: in questa modalità il pacchetto trasmesso mantiene l'header IP originale.

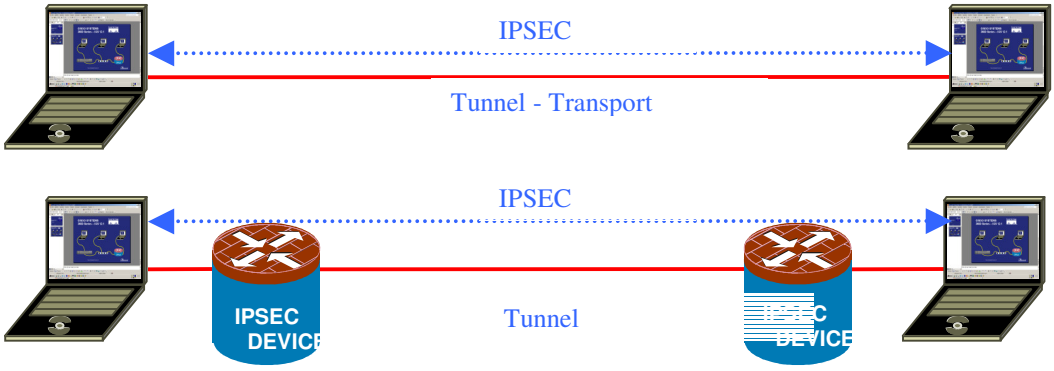

Figura I.4: : Transport mode e tunnel mode

Il tunnel mode può essere instaurato tra due security gateway allo scopo di securizzare il traffico che ha come mittente e destinatario lo stesso gateway o uno degli host protetti. In questa modalità l'header IP originale viene incapsulato da una nuova intestazione, che reca come mittente e destinatario i security gateway

I.3.2 I Database

Tutte le security association attive su un host o un security gateway sono contenute in un database detto Security Association Database (SAD), mentre esiste un altro database detto Security Policy Database (SPD) che contiene le politiche di sicurezza: tramite esse il sistema decide se un pacchetto IP debba essere scartato, lasciato passare in chiaro oppure elaborato tramite IPSec, basandosi su parametri come l'indirizzo IP sorgente o destinazione, la porta sorgente o destinazione, il protocollo di trasporto. Le security association possono essere combinate tra loro, sia nel caso che i nodi terminali siano gli stessi sia nel caso siano diversi: per esempio si potrebbero avere due SA in modalità trasporto tra due host (una per AH e una per ESP), oppure una SA in modalità trasporto tra due host a cui si aggiunge una SA in modalità tunnel tra due security gateway che stanno tra i due host.

I.3.3 I Protocolli

L'IPSec raccoglie un insieme di protocolli ed altri elementi che costituiscono l'intera architettura di sicurezza offerta. I protocolli principali che costituiscono IPSec sono due:

Authentication Header (AH) che fornisce servizi di autenticazione e integrità.

Encapsulating Security Payload (ESP) che fornisce servizi di riservatezza, autenticazione e integrità.

L'IPSec si appoggia al protocollo Internet Key Exchange (IKE) per gestire lo scambio delle chiavi. AH ed ESP non si preoccupano dello scambio delle chiavi e presumono che i due interlocutori si siano già accordati creando tra loro una security association (SA), ovvero un "contratto" che specifica quali meccanismi di sicurezza utilizzare e con quali chiavi. Il compito di negoziare e gestire le security association secondo delle politiche definite localmente è affidato appunto a IKE.

Un concetto fondamentale nell'ambito di IPSec è quello di Security Association. Una SA è una "connessione" tra due parti che comunicano tramite IPSec, e specifica quali meccanismi di sicurezza vengono utilizzati, quali algoritmi e quali chiavi, per proteggere il traffico che vi fluisce.

Una SA comprende:

Indirizzo IP di destinazione

Protocollo di sicurezza da utilizzare. La SA stabilisce se il traffico deve essere assistito per integrità e segretezza. Stabilisce inoltre la dimensione della chiave, la sua durata e gli algoritmi crittografici.

Chiavi segrete da utilizzare negli algoritmi crittografici

La definizione della modalità di incapsulamento, che stabilisce come sono create le intestazioni di incapsulamento e quali parti delle intestazioni e del traffico vanno protette durante il trasferimento.

L'Indice del Parametro di Sicurezza (SPI) è un numero casuale univoco che identifica la SA. Fornisce informazioni all'entità ricevente su come processare il traffico in arrivo.

Una SA è quindi identificata da uno SPI, un indirizzo IP di destinazione e da un identificatore del protocollo di sicurezza.

Le SA sono unidirezionali, in quanto definiscono le operazioni che si verificano nella trasmissione in un solo senso. Un tunnel può essere anche bidirezionale, in questo caso sono necessarie due SA per permettere a due host di comunicare tra loro: si tratta di utilizzare le stesse meta-caratteristiche per le due SA ma non le stesse chiavi. Inoltre una SA può essere associata ad AH o ad ESP, ma non ad entrambi. Per garantire al traffico le caratteristiche di sicurezza richieste, le SA possono essere raggruppate: ad esempio una può garantire la segretezza, mentre un'altra può assicurare l'integrità. I gruppi di SA sono identificati dallo stesso indirizzo IP di destinazione

I.3.4 Il protocollo AH

Il protocollo AH (Authentication Header) fornisce servizi di autenticazione, integrità e protezione da attacchi di tipo replay. AH autentica l'intero pacchetto IP, ad eccezione dei campi variabili dell'header IP (type of service, flags, fragment offset, time to live, header checksum, più alcune delle opzioni). Questi campi, essendo modificabili dai nodi intermedi, non possono essere autenticati. La posizione dell'header AH all'interno del pacchetto IPv4 e IPv6, nelle modalità trasporto e tunnel, è mostrata in Figura I.5 e Figura I.6

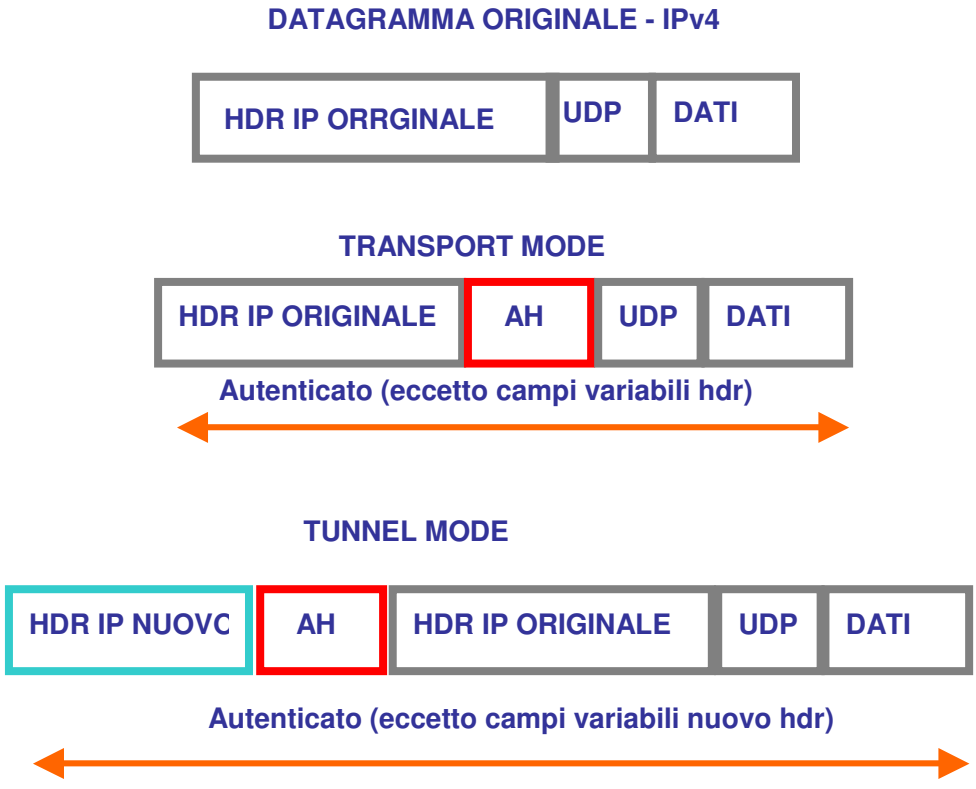

Figura I.5: Header AH all'interno del pacchetto ipv4

L'header del protocollo AH è mostrato in Fig. I.7 campi utilizzati sono:

Next header: contiene il codice identificativo del protocollo dell'header successivo, per esempio TCP o ESP.

Payload: contiene la lunghezza dell'header AH, espressa in parole di 32 bit, meno 2.

Reserved: riservato per usi futuri, deve essere posto a zero.

Security Parameters Index (SPI): contiene il valore numerico che, insieme con l'indirizzo IP di destinazione e il protocollo (ovvero AH, in questo caso) identifica la security association utilizzata. Viene stabilito dal destinatario quando la SA viene negoziata.

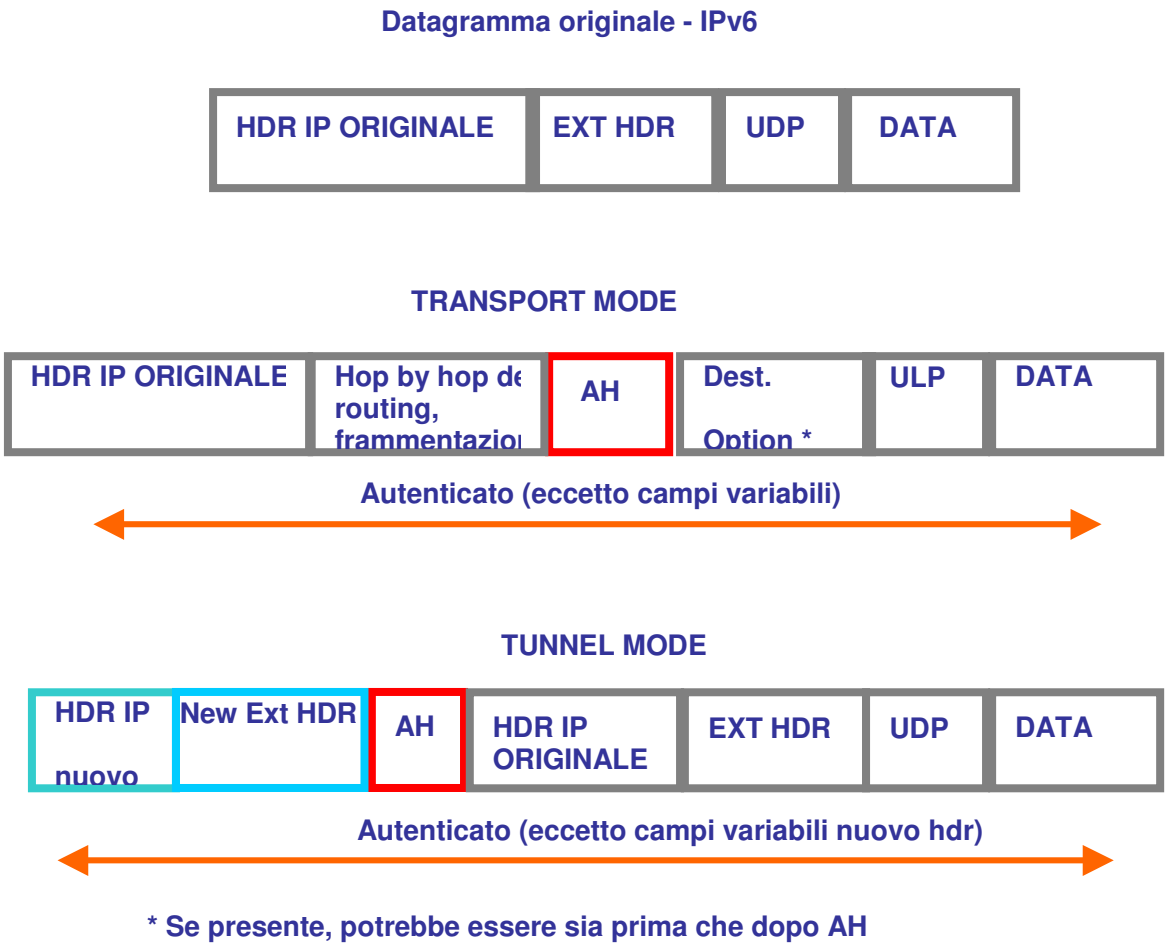

Figura I.6: Header AH all'interno del pacchetto ipv6

Sequence number: specifica il numero di sequenza del pacchetto all'interno della SA, per prevenire i replay attack. Il destinatario gestisce i numeri di sequenza (se ilservizio anti-replay è abilitato) tramite un meccanismo a finestra.

Authentication data: contiene il valore per il controllo dell'integrità (Integrity Check Value —ICV) del pacchetto. Per calcolare l'ICV e definire il MAC IPSec

prevede l'utilizzo di due algoritmi, MD5-HMAC-96 e HMAC-SHA-1-96. La lunghezza di questo campo è variabile ma deve essere un multiplo di 32 bit, per cui è possibile inserire un padding.

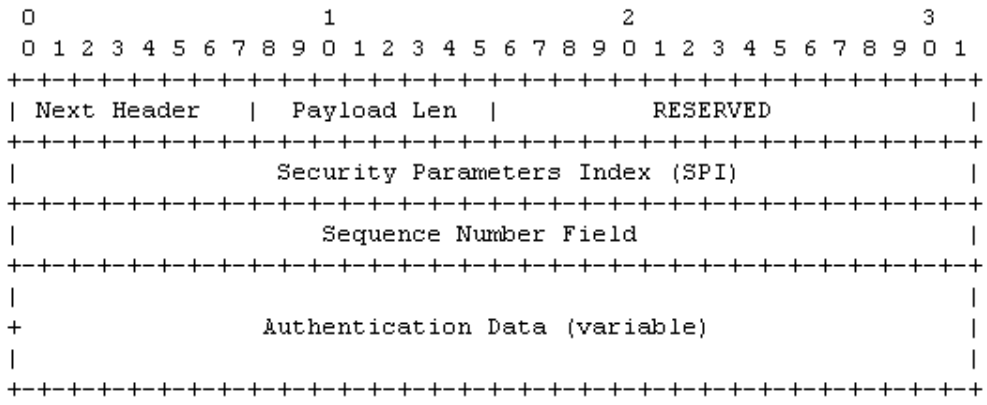

Figura I.7 formato header AH

L'ICV di AH è calcolato su i capi dell'intestazione IP non modificabili o deducibili dal nodo ricevente, sull'intestazione AH e su tutti i dati e le intestazioni dello strato superiore. Se un campo può essere modificato durante il transito, il suo valore deve essere impostato a zero ai fini del calcolo di ICV. Se il valore di un campo può essere modificato, ma il valore che assumerà a destinazione è predicibile, allora questo campo va inserito nel calcolo dell'ICV. Anche il campo di autenticazione è impostato a zero in preparazione a questa elaborazione.

- I campi dell'intestazione IPv4 sono classificati come segue:
- Non modificabili
- Versione
- Lunghezza dell'header
- Lunghezza totale
- Identificazione
- Protocollo
- Indirizzo di origine
- Indirizzo di destinazione (senza instradamento all'origine)
- Modificabile, ma deducibile:

- Indirizzo destinazione (con instradamento all'origine)
- Modificabili (impostati a zero prima dell'ICV)
- TOS (Type of Service): questo campo viene escluso in quanto è noto che alcuni router ne cambiano il valore anche se la specifica IP non lo considera un campo modificabile
- Flag: questo campo viene escluso in quanto un router intermedio può impostare il bit DF anche quando l'origine non lo ha selezionato.
- Offset di frammentazione: poiché AH viene applicato solo sui pacchetti IP non frammentati, questo campo deve essere sempre a zero, e quindi escluso anche se deducibile.
- TTL (Time to Live): questo campo viene modificato dai router durante la normale fase di elaborazione e il suo valore all'arrivo non è deducibile dal mittente.
- CheckSum dell'Header: questo campo viene modificato se cambia uno qualunque dei campi sopracitati.
- Opzioni: molte sono le regole associate a questo campo a seconda di quale opzione è selezionata.

Per Ipv6 si ha la classificazione seguente:
Non modificabili:

- Versione
- Lunghezza Payload (carico utile)
- Next Header
- Indirizzo origine (senza header dell'estensione di instradamento)
- Modificabile, ma deducibile
- Indirizzo di destinazione (con header estensione instradamento)
- Modificabili (impostati a zero prima dell'ICV)
- Classe
- Flow Label
- Limite di Hop

Per il traffico in uscita, se c'è frammentazione, essa avviene dopo l'elaborazione di AH. In questo modo la modalità di trasporto AH viene applicata solo per completare il datagramma IP, e non ai frammenti. Un pacchetto IP cui è stato applicato AH può essere frammentato dai router nel corso dell'elaborazione di AH al nodo ricevente. In modalità tunnel, AH viene applicata a un pacchetto IP il cui payload può essere un altro pacchetto IP frammentato.

Per il traffico entrante il riassemblamento dei frammenti viene effettuato prima dell'elaborazione di AH. La SA viene stabilita dall'esame dell'indirizzo IP di destinazione e dal tipo di AH e dallo SPI. Se non è presente nella configurazione dell'apparato IPSec, una SA valida, il pacchetto deve essere eliminato.

AH prevede inoltre l'utilizzo del numero di sequenza per ottenere la difesa da risposte false (anti-replay). Questo servizio può non essere attivo presso il destinatario, in questo caso il numero di sequenza viene ignorato.

Risolta la frammentazione, il nodo ricevente calcola l'ICV e confronta il risultato con l'ICV del campo autenticazione: se corrispondono il datagramma è valido.

I.3.5 Il protocollo ESP

Il protocollo ESP (Encapsulating Security Payload) fornisce servizi di riservatezza, autenticazione, integrità e protezione anti-replay. È possibile utilizzare solo il servizio di riservatezza, oppure solo i servizi di autenticazione e integrità (ed eventualmente anti-replay), oppure tutti i servizi insieme. Per quanto riguarda l'autenticazione, questa differisce da quella fornita dal protocollo AH in quanto non copre l'header IP esterno. La posizione di ESP all'interno del pacchetto IP nelle modalità tunnel e trasporto è mostrata in Figura 2.8. Come si vede, ESP aggiunge sia un header sia un trailer rappresentato dal gruppo coda ESP ed autenticazione ESP, perché incapsula tutti i dati che protegge. Nell'ambito IPv4 l'header di ESP è inserito dopo l'intestazione IP e prima del protocollo di strato superiore o prima di ogni altra intestazione IPSec precedentemente inserita, come si può osservare in figura I.3-7. In

IPv6 ESP è visto come payload end-to-end e quindi si posiziona dopo il campo hop-by-hop, routing e campi per la frammentazione come si può osservare in figura I.3-8

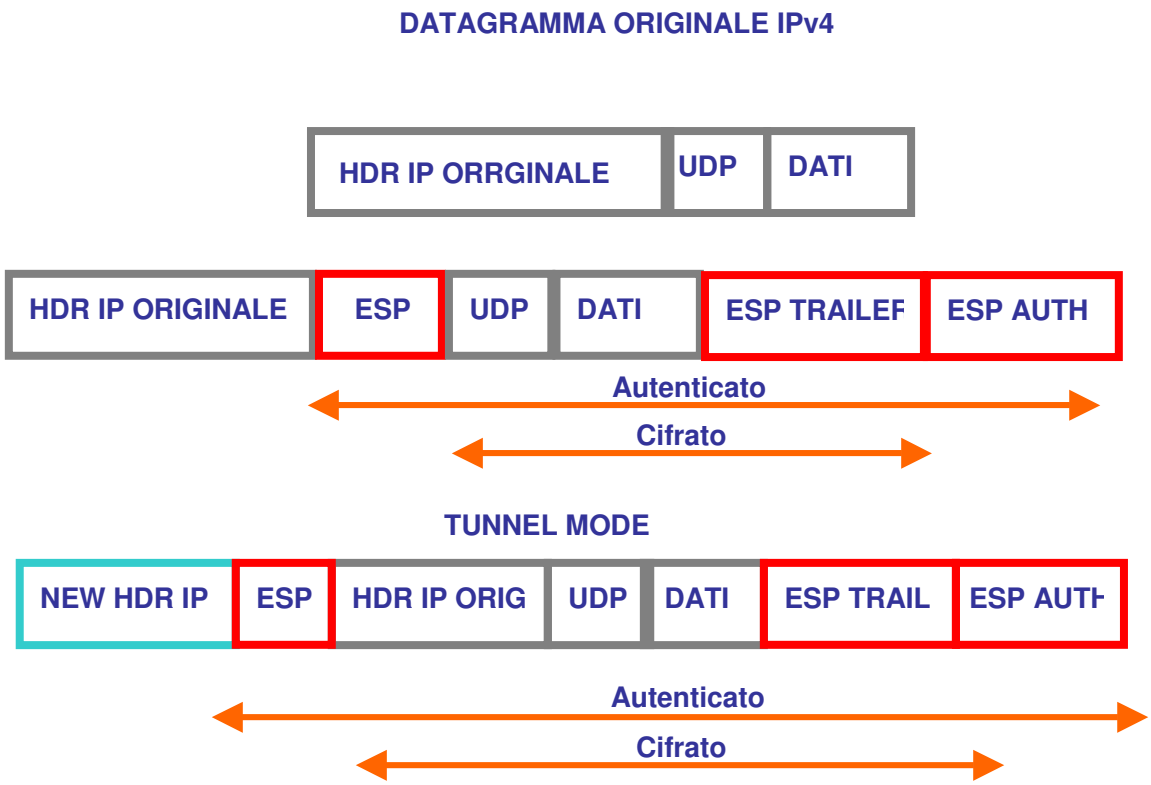

Figura I.8 Header ESP all'interno del pacchetto ipv4

Le intestazioni di estensione delle opzioni di destinazione potrebbero apparire prima o dopo l'intestazione ESP, a seconda della semantica. Poiché ESP protegge solo i campi dopo l'ESP header, in genere è meglio che le estensioni siano posizionate in modo da usufruire della cifratura.

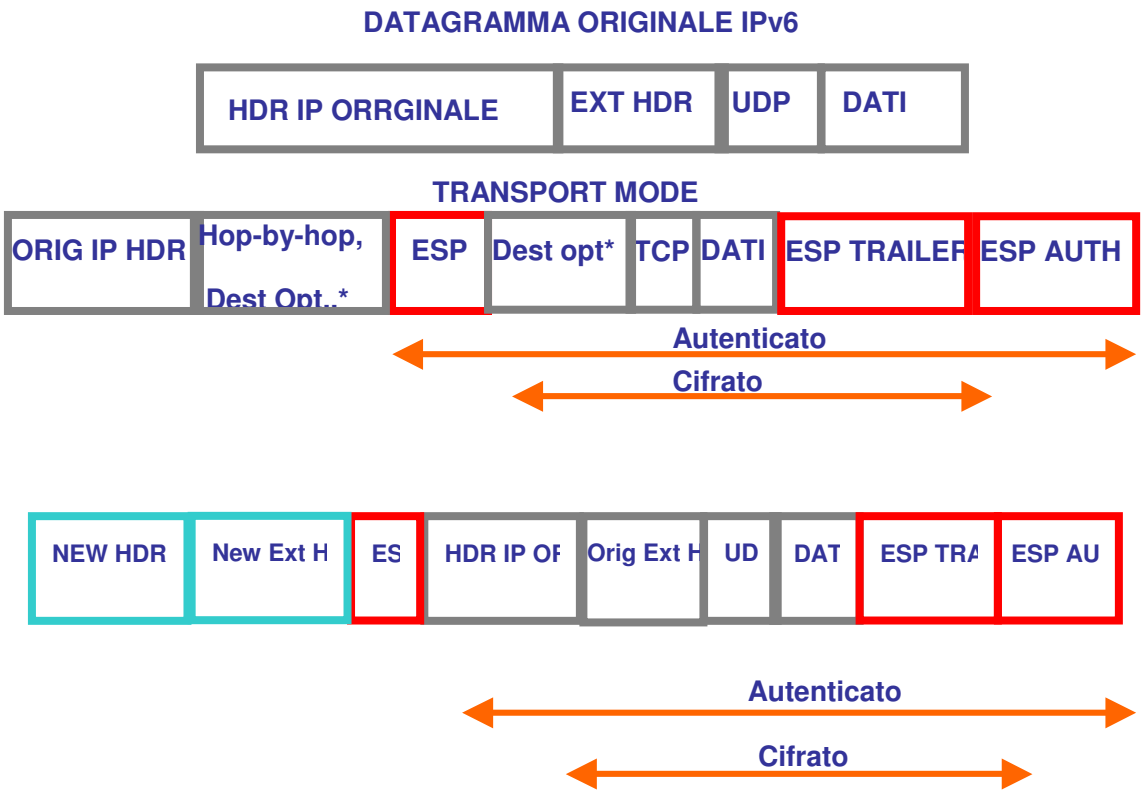

Figura I.9: header ESP all'interno del pacchetto ipv6

La struttura dell'header ESP secondo l'RFC-2406 è rappresentata nella Figura seguente

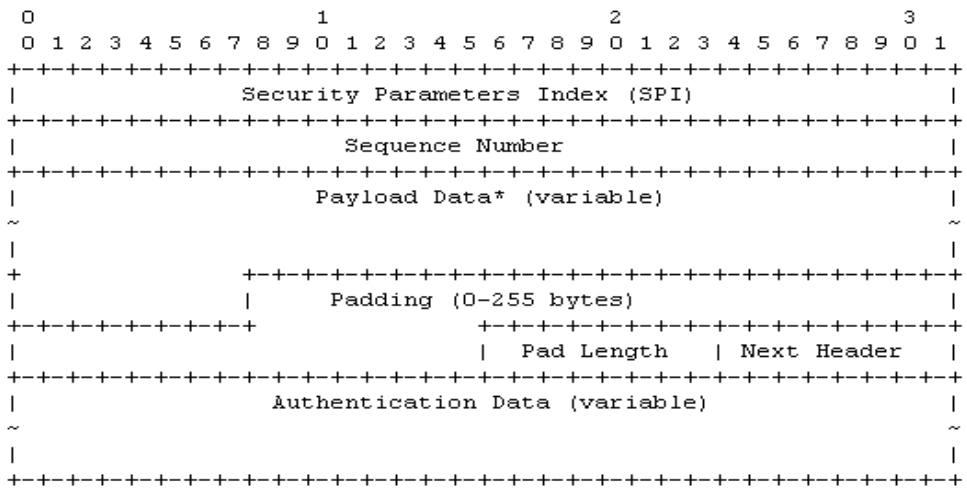

Figura I.10: Formato del Header ESP

Nell'header ESP sono presenti i seguenti campi:

Security Parameters Index (SPI): contiene un valore numerico che, insieme conl'indirizzo IP di destinazione e il protocollo (in questo caso ESP), permette di identificare la security association utilizzata. È analogo all'omonimo campo presente in AH.

Sequence number: contiene il numero di sequenza del pacchetto nell'ambito della security association, come in AH.

Payload data: contiene il payload del pacchetto IP originale (se in modalità trasporto) oppure l'intero pacchetto IP originale (se in modalità tunnel), cifrato se si utilizza il servizio di riservatezza. Nel caso l'algoritmo di cifratura utilizzato necessiti di un vettore di inizializzazione (Initialization Vector — IV), questo viene inserito all'inizio del payload.

Padding: il padding (variabile tra 0 e 255 byte) può essere necessario sia perché l'algoritmo di cifratura può richiedere che il testo in chiaro abbia una dimensione multipla di un certo valore, sia per assicurare il corretto allineamento dei campi successivi, come mostrato in figura 2.4. È anche possibile aggiungere un padding per limitare gli effetti di un'analisi del traffico basata sulla dimensione dei pacchetti.

Pad length: contiene la lunghezza del padding.

Next header: contiene il codice identificativo del protocollo per i dati contenuti nel payload, per esempio TCP o UDP. Si noti che, se si utilizza il servizio di riservatezza, questo campo è cifrato.

Authentication data: contiene il valore di controllo integrità (ICV), calcolato sull'intero pacchetto ESP escluso questo campo. È presente solo se si utilizza il servizio di autenticazione/integrità.

Anche ESP, come AH, presuppone che esista già una security association tra i due nodi e non si preoccupa di negoziarne i parametri.

Per la crittografia dei pacchetti in uscita, per prima cosa bisogna operare l'incapsulamento all'interno del campo payload di ESP, completando per mezzo del padding. Il risultato viene crittografato utilizzando la chiave, l'algoritmo crittografico e la modalità di applicazione dell'algoritmo, come da indicazioni della corretta SA. Se

l'autenticazione è richiesta, deve essere per prima effettuata la crittografia senza comprendere il campo Authentication data. Nel campo dell'ICV sono compresi: lo SPI, il sequence number, il payload, l'eventuale padding, la lunghezza del pad e il next header. Gli ultimi quattro campi saranno cifrati.

Per i pacchetti in entrata, qualora un elemento intermedio li abbia frammentati, l'eventuale riassemblamento viene effettuato prima dell'elaborazione di ESP. A questo punto il nodo ricevente determina la SA unidirezionale appropriata, basata sull'indirizzo IP di destinazione, ESP e lo SPI. La SA definisce se il campo sequence number deve essere controllato o meno, se c'è autenticazione e specifica gli algoritmi e le chiavi per la decodifica, e gli eventuali calcoli dell'ICV.

I.3.6 Collocazione

La collocazione specifica di IPSec a livello di hardware o software non viene definita come requisito nelle specifiche. Si possono identificare tre scenari plausibili, mostrati nella Fig. 2.10. Nel primo il software IPSec è situato direttamente nel codice IP di origine, per esempio a livello del software dell'host o del gateway. Questo approccio è chiamato "bump in the code" (BITC)

Un secondo scenario prevede la collocazione di IPSec al di sotto dello stack del protocollo IP. Questo approccio è in genere applicabile agli host, con IPSec operante sopra il MAC a livello di LAN. Questo approccio è detto "bump in the stack" (BITS).

Nel terzo si prevede di collegare all'host o al gateway un elemento separato, munito per esempio di processore crittografico. Si ha in questo caso un approccio "bump in the wire" (BITW), e in caso l'apparato supporti funzionalità di routing siamo di fronte ad un security gateway.

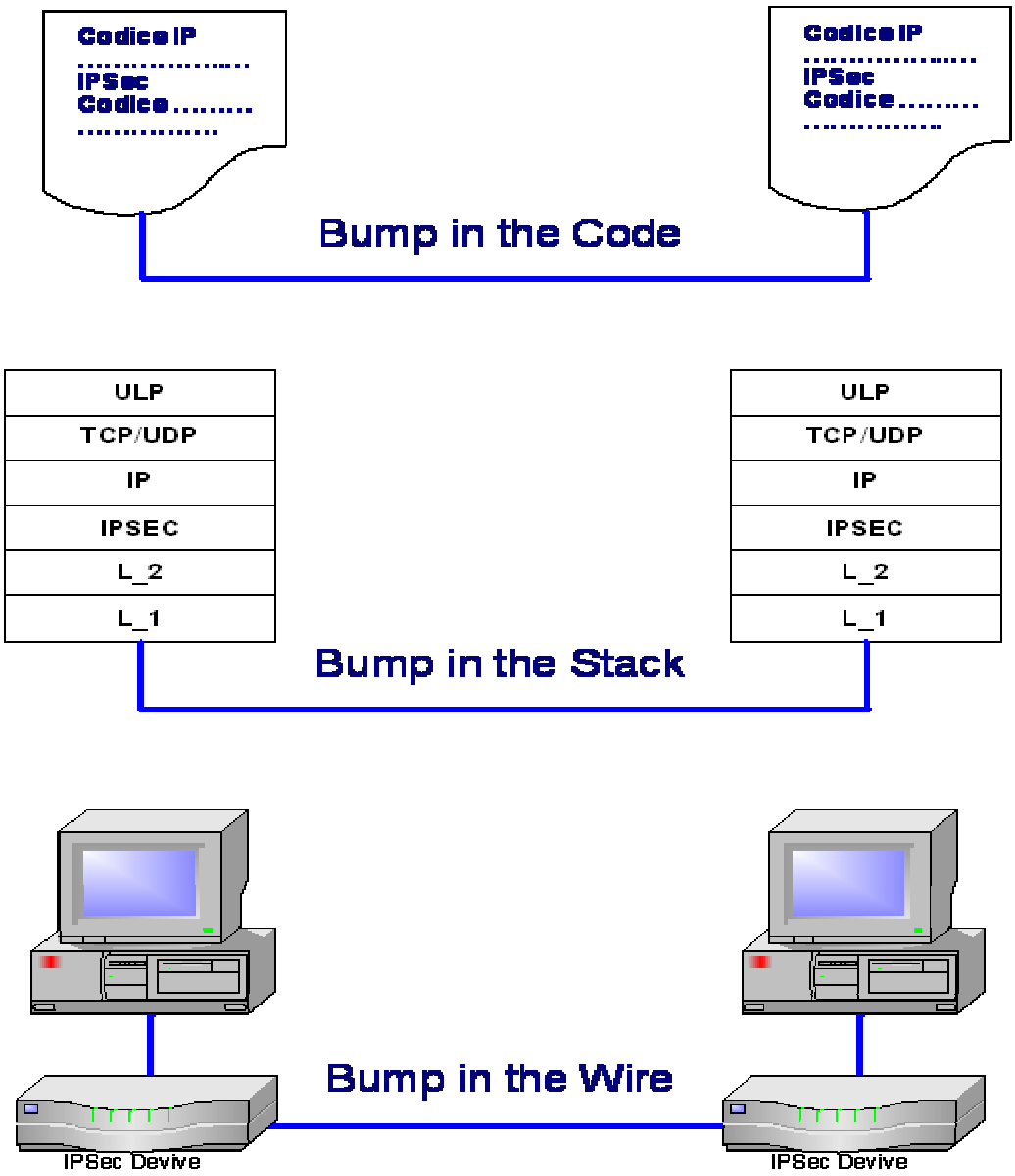

Figura I.11 Possibili scenari per l'implementazione di IPSec

I.4 IKE – Internet Key Exchange

I.4.1 Introduzione

IPSec gestisce il traffico ip per mezzo di SA, tramite le quali invia e riceve traffico protetto con determinate caratteristiche di sicurezza quali confidenzialità, autenticazione etc.; presuppone però che tali SA siano già state stabilite prima dello scambio di traffico IPSec. Risulta evidente che devono esserci delle procedure che stabiliscano associazioni di sicurezza tra i due peer. Tali operazioni rientrano nelle competenze del protocollo IKE. Tuttavia l'IKE non è di per sé un protocollo unico: esso infatti nasce dalla unione di due protocolli (ISAKMP e OAKLEY). Il primo definisce uno schema generico di scambi per poter garantire, attraverso un determinato numero di pacchetti costituito da un preciso numero di "payloads", lo scambio di informazioni con caratteristiche quali sicurezza, protezione di identità e autenticazione. Esso prevede la possibilità di definire chiavi generate in comune fra le due entità, ma non entra nei particolari delle finalità precise dello scambio, del modo di autenticazione né del meccanismo di generazione delle chiavi stesse. Il protocollo OAKLEY definisce un metodo attraverso cui generare quello che viene definito uno "shared secret", ovvero delle stringhe comuni ad entrambi i peer, generate all'atto dello scambio, ma mai inviate completamente in rete, quindi generabili solo da chi prende parte allo scambio.

L'IKE, prendendo il meccanismo di scambi definito da ISAKMP, la generazione delle chiavi di OAKLEY, utilizza alcuni degli scambi per le tre finalità viste in precedenza, ovvero autenticazione forte del peer, negoziazione delle SA e generazione delle chiavi per il protocollo client.

I.4.2 Shred secret

Avendo a disposizione un meccanismo di chiave asimmentrica come l'RSA[2], lo shared secret può essere realizzato in modo molto semplice: è infatti sufficiente che uno dei due peer generi una stringa casuale di byte e la invii all'altro cifrandola con la chiave pubblica del peer. Solo questi, quindi, per le note proprietà degli algoritmi di cifratura asimmetrici, potrà decodificare la stringa.

Questo meccanismo, nella sua semplicità, richiede la presenza di uno schema di generazione di chiavi pubbliche e private e della loro distribuzione, schema che vincolerebbe pesantemente la generalità dell'IKE. Si è quindi preferito l'algoritmo Diffie-Hellman[2] che non prevede architetture preesistenti, ma solo un meccanismo esterno di autenticazione forte.

Questo algoritmo fa uso di una proprietà per certi versi vicina a quella che rende affidabile l'RSA: la difficoltà, dato un valore x – parte pubblica - ottenuto come esponenziazione a modulo noto di un generatore noto con esponente casuale (parte privata) di risalire all'esponente stesso. Se, però, entrambi i peer eseguono una tale operazione partendo dallo stesso generatore e modulo, si può avere uno shared secret semplicemente inviando x all'altra entità ed elevando all'esponente casuale il valore ricevuto. Le due quantità sono identiche nonostante che in rete non sia passato mai nessuno dei valori segreti generati indipendentemente dai due peer, ma solo la loro esponenziazione che li rende, se il modulo ha opportune proprietà, difficilmente decrittabili.

Dal punto di vista logico, una rappresentazione del tipo di protezione offerta dal Diffie-Hellman è data in figura II.4.1. In questo schema il mittente protegge il messaggio con un proprio "lucchetto" e lo invia al destinatario, il quale a sua volta pone il proprio, e lo rispedisce al mittente. A questo punto il mittente toglie il lucchetto di cui possiede la chiave e rimanda il messaggio al destinatario, che è ora in grado di ottenere il messaggio. Questa rappresentazione non segue l'algoritmo del protocollo, che ha come obiettivo la generazione di uno shared secret, ma dimostra come sia possibile inviare dati protetti senza prima aver scambiato le chiavi.

[2] Per dettagli sui protocolli RSA e Diffie-Hellman, si veda l'appendice crittografica.

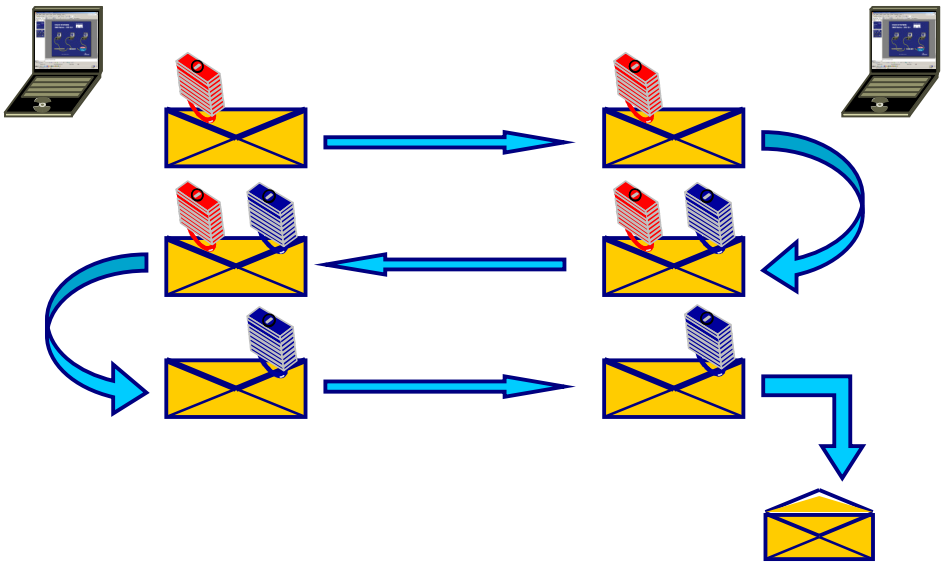

Figura II.4-1: scambio Diffide-Hellmann

Nello schema IKE il generatore vale 2, mentre si hanno moduli a 768, 1024 e 1536 bit (gruppi di Diffie-Hellman rispettivamente I, II e V). La grandezza del modulo ovviamente determina la forza dell'algoritmo stesso, ma anche la pesantezza del calcoli. La rumorosità invece dello shared secret è determinata dalla lunghezza della parte segreta. Vengono richiesti almeno 20 byte di esponente. Gli schemi logici degli algoritmi DH e RSA sono rappresentati nella figura II.4.2 e nella II.4.3

Il punto debole di questo algoritmo è il "men in the middle attack": infatti, lo shared secret viene generato con chiunque supporti lo scambio, anche con il nemico che, ponendosi nel mezzo, potrebbe spacciarsi rispetto ad ogni peer come se fosse l'altro e possedere entrambi i secrets, quindi le chiavi. Da qui l'assoluta necessità di un meccanismo di autenticazione che non usi materiale generato nello stesso scambio, ma qualcosa di noto per altre vie.

In Figura II.4-3 è mostrato l'algoritmo RSA, dove il lucchetto aperto rappresenta la chiave pubblica.

Il destinatario del messaggio invia la propria chiave pubblica al mittente che la utilizzerà per cifrare. Il messaggio cifrato potrà essere decifrato solo dal destinatario, essendo l'unico in possesso della chiave privata.

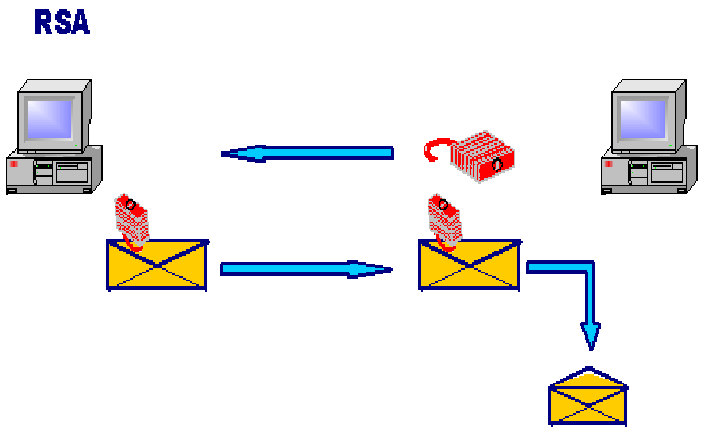

Figura II.4-3:schema dell' algoritmo RSA

Scopo particolare di IKE è di negoziare e fornire il materiale crittografico in maniera protetta. Inoltre IKE può essere utilizzato anche al di fuori di IPSec per esempio è attualmente associabile ai DOI di RIPv2 e OSPF.I.4.3

I.4.3 Le due fasi

IKE è strettamente legato a ISAKMP e ne utilizza le due fasi caratteristiche, adattandole al proprio scopo:

Fase uno: definire la SA IKE

Fase due: utilizzare la SA IKE per negoziare una SA per IPSec o un altro protocollo

Più in particolare IKE prevede differenti tipi di scambi per ogni fase più alcuni di tipo speciale:

Scambio di fase uno in Main Mode (Figura II.4-4): è costruito in modo da far sì che i payloads che portano informazione sulle identità dei peer viaggino protetti (ovvero cifrati). Questo rende lo scambio più macchinoso (sei pacchetti) e meno flessibile al riconoscimento del chiamante, in modo particolare se non si usano certificati. Esso rappresenta il metodo di default e deve essere supportato.

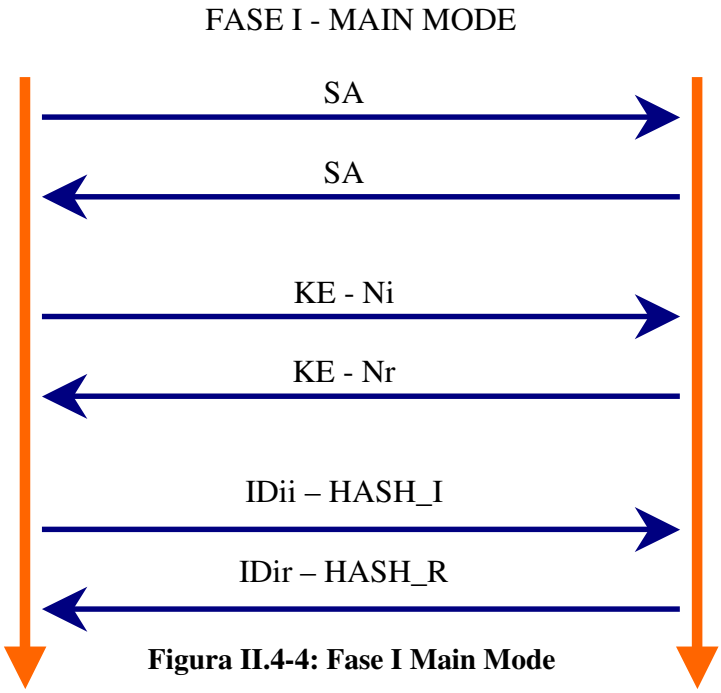

Figura II.4-4: Fase I Main Mode

Scambio di fase uno in Aggressive Mode(Figura II.4-5): L'aggressive ha meno sicurezza, ma maggiore flessibilità e snellezza, dal momento che definisce solo tre pacchetti. Questa modalità è opzionale.

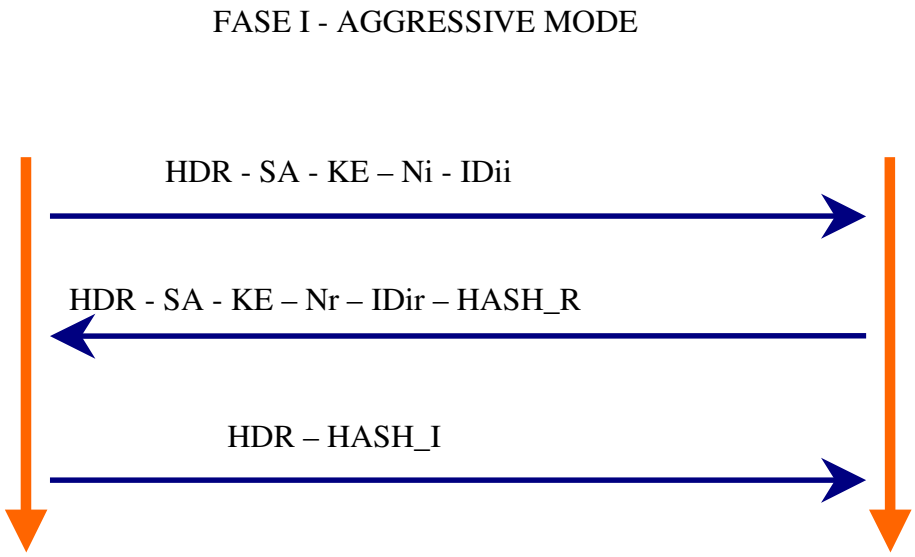

Figura II.4-5: Fase I Aggressive Mode

Scambio di fase due Quick Mode (Figura II.4-6): lo scambio in quick mode è costituito da tre pacchetti e ha come obiettivo la negoziazione della SA di IPSec.

Scambio “informational”.

Scambio “new group

FASE II - QUICK MODE

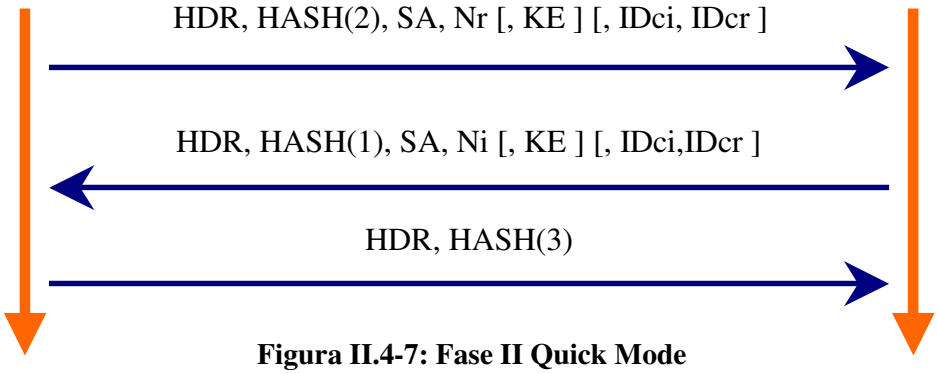

Figura II.4-7: Fase II Quick Mode

Fondamentale in IKE è il ruolo che ciascun security gateway assume, diverso è infatti il comportamento di chi inzia (initiator) la comunicazione rispetto a chi risponde (responder).

Nella Figura II.4-5 e nella Figura II.4-6 sono rappresentati gli scambi di fase I rispettivamente in main mode e in aggressive mode. Entrambe le rappresentazioni si riferiscono ad uno scambio che prevede autenticazione tramite preshared-key, e per semplicità non è stato indicato l’header. La Figura II.4-7 rappresenta invece uno scambio in quick mode. I pacchetti nelle figure su citate mancano di header per semplicità di rappresentazione, gli altri payload sono:

SA: rappresenta il payload contenente le proposal e le rispettive transform di cui parleremo in seguito.

KE: è il payload che contiene le informazioni pubbliche dello scambio Diffie-Hellman.

Ni, Nr: rappresentano il nonce payload initiator e responder. Il nonce contiene dati generati casualmente ed utilizzati peraltro anche per proteggere dagli attacchi di tipo “replay”.

IDii, IDir: rappresentano il payload di identificazione per l'initiator ed il responder.

HASH_I, HASH_R, HASH (n): tutti questi payload sono utilizzati dall'algoritmo di autenticazione, e sono il risultato di funzioni di hashing

I.4.4 La fase uno

Entrambi gli scambi della fase uno conducono comunque allo stesso risultato: lo stabilire una SA che permetta di inviare pacchetti cifrati e autenticati rispettivamente dagli algoritmi di cifratura e di hashing oggetto della negoziazione, di generare uno shared secret mediante l'algoritmo di Diffie-Hellman, da cui si traggono le chiavi per i precedenti due, e di autenticare, in modo forte, il peer. Con "forte" si intende mediante informazioni non comunicate attraverso i pacchetti scambiati.

Dei quattro metodi di autenticazione previsti, la prima versione dell'IKE prevede l'implementazione dei meccanismi di preshared key e di firma digitale.

Nel primo, l'autenticazione viene effettuata mediante un hashing su alcuni dei payloads precedentemente scambiati effettuato con la funzione negoziata e con una chiave che origina da una stringa precondivisa tra i due utenti. È, ovviamente, importante che essa non venga comunicata attraverso la stessa negoziazione, bensì "out of band", ovvero tramite un altro sistema di comunicazione.

Nel secondo modo, la stessa stringa di hashing, che funziona anche da autenticazione dell'intero scambio, viene firmata con la chiave privata del mittente. Il ricevente può così, oltre che autenticare lo scambio, anche verificare l'identità del mittente mediante l'uso dei certificati.

Parte della SA sono anche le informazioni circa la loro scadenza che può avvenire per timeout, per raggiunti limiti di traffico o anche per raggiunto massimo numero di generazioni di chiavi della seconda fase. Infatti, lo shared material, viene generato qui e usato tutte le volte che si richiede finché la SA rimane in piedi. Ad ogni utilizzo si ha ovviamente una degradazione della forza delle stringhe segrete.

Il primo scambio non ha di per sé nessuna informazione relativa al protocollo client. Il legame viene stabilito tramite il cosiddetto Domain Of Interpretation (DOI) che stabilisce come, dopo la prima fase, debbano essere interpretati i campi dei payloads successivi. In pratica, permette di mappare i campi IPSec (o di eventuali altri protocolli) in concetti IKE. Esso definisce anche i modi in cui i peer si possono identificare. A questo scopo, lo standard ISAKMP prevede solo l'utilizzo di indirizzo IP o di sottorete. Nello standard DOI, invece, si ha un ventaglio di possibilità molto più ampio comprendente, oltre ai due prima citati, anche range di indirizzi e soprattutto distinguish name del certificato, o stringhe FQDN o stringhe opache. Il prossimo paragrafo definisce in modo più dettagliato questi tipi di identificativi.

Uno dei principali scopi del primo scambio IKE è rappresentato dal riconoscimento del peer con cui si sta interagendo. Questo è ottenuto anche facendo riferimento all'indirizzo IP che compare nella testa UDP del pacchetto IKE (l'IKE infatti, utilizza il protocollo UDP su porta 500). Ma questa informazione può essere carente per vari motivi:

presenza di NAT nel percorso del pacchetto che maschera l'indirizzo originale nelle teste esterne, ma non ovviamente i valori contenuti nell'interno dei payload (che sono cifrati e autenticati);

utilizzo di peer "mobili", come PC portatili connessi a POP internet per i quali non è ovviamente possibile, né utile, prevedere quale indirizzo IP potrà esser loro assegnato;

possibilità da parte di utenti non graditi di mascherarsi data la facilità di cambiare questo parametro.

Per questo, nel pacchetto IKE, è previsto un apposito payload che permette di inviare un identificativo del peer. Esso può essere uno dei seguenti:

- Indirizzo IP (es.: 123.134.156.23);
- Sottorete IP (es.: 123.134.156.0 255.255.255.0);
- FQDN, name (es.:andrea.rossi@elsag.com);
- FQDN, dominio (es.: elsag.com);
- IP range (es.: 123.134.156.23 123.134.156.129);
- Distinguish Name (DN) del certificato in formato DER ASN1;
- General Name (GN) del certificato in formato DER ASN1;
- Stringa opaca (es.: identificativoRemoto).

Alcuni di questi sono vincolati al tipo di autenticazione scelta: per esempio, il DN o il GN richiedono l'uso dei certificati per essere utilizzati, la stringa opaca è disegnata per l'uso della preshared in aggressive mode. È buona norma che, in presenza dei certificati, il campo dell'identificativo sia presente nel certificato stesso. Vedremo come l'implementazione su SAS permette la definizione di questi parametri per effettuare il controllo di accesso.

I.4.5 La fase due

Finora abbiamo analizzato i passaggi che permettono la creazione di una SA propria dell'IKE che permette di avere un canale di comunicazione tra due peer ormai "riconosciuti". L'uso degli algoritmi di cifratura e hashing con le chiavi prima generate permette di poter effettuare molte negoziazioni per conto di protocolli client tra gli stessi peer senza dover effettuare nuove autenticazioni, operazione computazionalmente pesante.

L'obiettivo finale di una negoziazione IKE, oltre a quello ormai già ottenuto nella prima fase di riconoscere il peer remoto, e fare quindi controllo di accessi, è quello di negoziare uno o più protocolli con relativi algoritmi e chiavi per poter permettere

traffico IP con i servizi scelti tra i due peer. Questa negoziazione con relativa generazione di materiale per chiavi, viene ottenuta nel corso della seconda fase.

Lo scambio utilizzato è in questo caso unico e denotato Quick Mode. Esso non è definito nella specifica ISAKMP, ma utilizza payload e strutture logiche di questo schema. Esso, venendo dopo l'autenticazione della prima fase, non prevede l'autenticazione del peer, ma solo la negoziazione dei protocolli per il modulo cliente e degli identificativi degli host client. Se, infatti, nella prima fase l'identificativo era quello della macchina che funge da end point di cifratura, in questo secondo momento esso è quello delle macchine che generano effettivamente i pacchetti di traffico. Naturalmente nel caso in cui IKE sia utilizzato in un PC, oppure nel caso di traffico verso il gateway cifrante, i due soggetti coincidono. Nel caso di un apparato di accesso, invece, si tratta di concetti alquanto diversi: l'elemento cifrante nella prima fase, gli host che attraverso esso possono uscire nella seconda. Qui in genere come identificativi si utilizzano range o sottoreti o indirizzi IP singoli.

Va anche notato che, mentre nella prima fase ognuno dei due peer invia il proprio identificativo, in questo caso chi inizia invia l'identificativo di chi vuole parlare e quello di chi vuole raggiungere. La stessa informazione viene replicata da chi risponde.

Al termine di questa fase, si hanno le SA per il protocollo client munite di tutte le chiavi necessarie per il loro funzionamento. Tra le opzioni, vi è la possibilità di effettuare un nuovo Diffie-Hellman in modo da non degradare il materiale di chiave generato nella prima fase.

I.4.6 Altri scambi

Oltre agli scambi che "costruiscono" le SA, vi sono quelli che ne permettono la cancellazione o che trasportano informazioni circa errori eventualmente verificatesi durante una negoziazione.

Trattandosi di un protocollo di sicurezza non sono ammessi errori, quindi ogni condizione critica per errore di protocollo, per fallita negoziazione o fallito controllo di accesso provoca l'immediata cancellazione della fase che era in corso di negoziazione e la notifica dell'errore al peer interessato. Queste notifiche viaggiano protette dalla SA IKE, se si verificano quando questa SA è già in piedi ma devono essere, o meglio possono essere inviate, solo in chiaro se l'errore investe la prima fase. La politica di sicurezza locale deve indicare se prendere in considerazione o meno tali messaggi essendo facile generarli anche da parte di apparati non facenti parte dello scambio; questo potrebbe portare ad una truffaldina cancellazione di SA in progress, quindi ad attacchi di Denial of Service (DoS).

Gli scambi che portano informazioni circa l'avvenuta cancellazione delle SA (sia IKE che del protocollo cliente) possono invece essere inviati sotto protezione di SA (almeno di quella che sta per essere cancellata). Questi scambi sono molto semplici essendo costituiti da un unico pacchetto, infatti tutte le informazioni sono ormai presenti nelle SA. Sono chiamati "Informational Mode" e, come per il quick, non hanno riferimento nella specifica di ISAKMP, anche se ne utilizzano logica e payloads.

I.5 ISAKMP

I.5.1 Introduzione

Una fase molto importante nella instaurazione di un canale sicuro è la fase di autenticazione nella quale un'entità si autentica ad un'altra. Esistono molti meccanismi di autenticazione che possono essere suddivisi in due grandi categorie: forti e deboli.

Un meccanismo di autenticazione debole si ha, ad esempio, quando si inviano le chiavi in chiaro oppure quando si inviano in chiaro altre informazioni caratterizzanti la fase di autenticazione, che in questo modo sono facilmente recuperabili per mezzo di uno sniffer. Sono considerati meccanismi di autenticazione forte, invece, la Digital Signature Standard (DSS) e la RSA Signature. Quando si usano questi meccanismi di Digital Signature a chiave pubblica ciascuna entità deve possedere una chiave pubblica e una chiave privata. Inoltre, i certificati hanno un'importanza fondamentale nel meccanismo di autenticazione con digital signature: legano l'identità di un'entità alle sue chiavi pubbliche ed eventualmente ad altre informazione di sicurezza.

L'autenticazione basata sulle digital signatures può richiedere la presenza di una terza parte, chiamata "Certificate Authority" (CA), il cui compito è quello di creare, firmare e distribuire i certificati.

I.5.2 Fase di Autenticazione

Una fase molto importante nella instaurazione di un canale sicuro è la fase di autenticazione nella quale un'entità si autentica ad un'altra. Esistono molti meccanismi di autenticazione che possono essere suddivisi in due grandi categorie: forti e deboli.

Un meccanismo di autenticazione debole si ha, ad esempio, quando si inviano le chiavi in chiaro oppure quando si inviano in chiaro altre informazioni caratterizzanti la fase di autenticazione, che in questo modo sono facilmente recuperabili per mezzo di uno sniffer.

Sono considerati meccanismi di autenticazione forte, invece, la Digital Signature Standard (DSS) e la RSA Signature. Quando si usano questi meccanismi di Digital Signature a chiave pubblica ciascuna entità deve possedere una chiave pubblica e una chiave privata. Inoltre, i certificati hanno un'importanza fondamentale nel meccanismo di autenticazione con digital signature: legano l'identità di un'entità alle sue chiavi pubbliche ed eventualmente ad altre informazione di sicurezza.

L'autenticazione basata sulle digital signatures può richiedere la presenza di una terza parte, chiamata "Certificate Authority" (CA), il cui compito è quello di creare, firmare e distribuire i certificati.

I.5.3 Requisiti di ISAKMP

Negli scambi ISAKMP si deve garantire un metodo di autenticazione forte. Infatti, se questo non avviene è possibile che la SA e le chiavi siano state scambiate con un intruso che sta attuando un man-in-the-middle attack e sta intercettando i nostri dati personali.

ISAKMP richiede l'utilizzo di un algoritmo di digital signature tuttavia non stabilisce quale di essi debba essere utilizzato né quale CA debba essere impiegata. Infatti, ISAKMP consente all'entità Initiator della comunicazione di indicare quale CA utilizzare. Una volta stabilita la CA utilizzata, il protocollo fornisce i messaggi richiesti per supportare la fase di autenticazione e dà la possibilità di supportate oltre a varie CA anche diversi formati del certificato, quali ad esempio il formato X.509, PKCS #7, PGP, DNS SIG, etc.

I.5.4 ISAKMP Header

Un messaggio ISAKMP ha un header di lunghezza fissa, la cui struttura è mostrata in figura II.5-1:

I campi dell'ISAKMP Header sono definiti come segue:

Initiator Cookie (4 ottetti): Cookie dell'entità che richiede l'instaurazione o la distruzione di una SA.

Responder Cookie (4 ottetti): Cookie dell'entità che risponde alla richiesta di instaurazione o di distruzione di una SA.

Next Payload (1 ottetto): indica il tipo del primo payload nel messaggio. Si possono avere diversi tipi di payload:

Next Payload Type	Value
NONE	0
Security Association (SA)	1
Proposal (P)	2
Transform (T)	3
Key Exchange (KE)	4
Identification (ID)	5
Certificate (CERT)	6
Certificate Request (CR)	7
Hash (HASH)	8
Signature (SIG)	9
Nonce (NONCE)	10
Notification (N)	11
Delete (D)	12
Vendor ID (VID)	13
RESERVED	14 - 127
Private USE	128 – 255

Major Version (4 bits): indica la versione maggiore del protocollo ISAKMP in uso.

Minor Version (4 bits): indica la versione minore del protocollo ISAKMP in uso.

Exchange Type (1 ottetto): indica il tipo di scambio che viene usato. Questo campo indica l'ordine dei messaggi e dei payloads negli scambi ISAKMP.

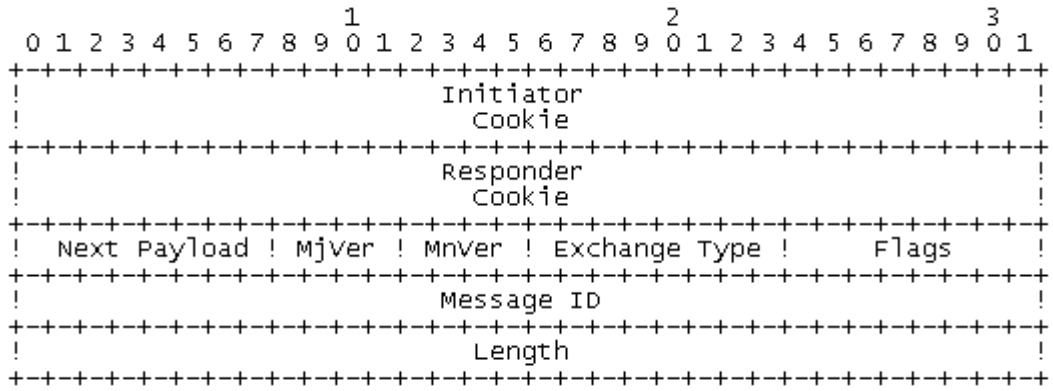

Figura II.5-1:Header Isakmp

Flags (1 ottetto): indica le opzioni specifiche che sono settate per lo scambio ISAKMP.

Message ID (4 ottetti): serve ad identificare lo stato del protocollo durante le negoziazioni della fase due. Questo valore viene generato in modo random dall'Initiator della negoziazione della fase due.

Length (1 ottetto): Lunghezza del messaggio totale (header + payloads) in ottetti.

I.5.4 ISAKMP Generic Payload Header

Esistono diversi tipi di payload ISAKMP e ciascuno di essi è costituito da un header generico la cui struttura è riportata in Figura II.5-2

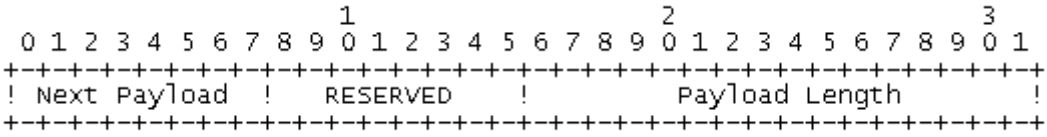

Figura I.5-2: Generic payload Header

I suoi campi sono definiti come segue:

Next Payload (1 ottetto): identificativo per il tipo di payload del prossimo payload nel messaggio. Se il payload corrente è l'ultimo nel messaggio, questo campo sarà settato a 0.

Reserved (1 ottetto):non è usato. Viene settato a 0.

Payload Length (2 ottetti): lunghezza in ottetti del payload corrente, incluso il generic payload header.

I.5.5 Identification Payloads

Tra i diversi payload presenti nel protocollo ISAKMP risulta di particolare importanza, nell' autenticazione basata su PKI, l'Identification Payload.

L'Identification Payload contiene dati DOI-specific usati per scambiare informazioni di identificazione. Queste informazioni sono usate per determinare le identità dei peers comunicanti e possono essere usate per determinare l'autenticità dell'informazione. Il formato di tale payload è mostrato in Figura II.5-3:

I suoi campi sono definiti come segue (i campi dell'intestazione sono stati definiti nel paragrafo precedente):

ID Type (1 ottetto): indica il tipo di identificazione che viene utilizzata. I valori assumibili da questo campo sono i seguenti:

ID Type	Value
RESERVED	0
ID_IPV4_ADDR	1
ID_FQDN	2
ID_USER_FQDN	3
ID_IPV4_ADDR_SUBNET	4
ID_IPV6_ADDR	5
ID_IPV6_ADDR_SUBNET	6
ID_IPV4_ADDR_RANGE	7
ID_IPV6_ADDR_RANGE	8
ID_DER_ASN1_DN	9
ID_DER_ASN1_GN	10
ID_KEY_ID	11

Figura I.5-3: Valori del campo ID Type.

Se l'autenticazione nella fase uno del protocollo IKE viene effettuata usando i certificati digitali (di ciascun formato), ciascun ID usato come input nelle decisioni di policy locale dovrebbe (SHOULD) essere contenuto nel certificato usato nell'autenticazione dello scambio.

In particolare:

- ID_IPV4_ADDR: indica l'indirizzo IPv4.
- ID_FQDN:specifica una stringa "Fully-Qualified Domain Name". Un esempio di ID_FQDN è "foo.bar .com".
- ID_USER_FQDN: specifica una stringa "Fully Qualified Username". Un esempio di ID_USER_FQDN è "piper@foo.bar.com".
- ID_DER_ASN1_DN: specifica la codifica binaria DER di un Distinguished Name in formato ASN.1 X.500 dell'entità i cui certificati sono stati scambiati per stabilire la SA.

DOI specific ID Data (3 ottetti): contiene dati di identificazione DOI specific. Se non viene usato va settato a 0.

Identification Data (lunghezza variabile): contiene informazioni di identità. I valori per questo campo sono DOI specific e il formato è specificato dal campo ID Type.

Il Payload Type per per l'Identification Payload è cinque (5).

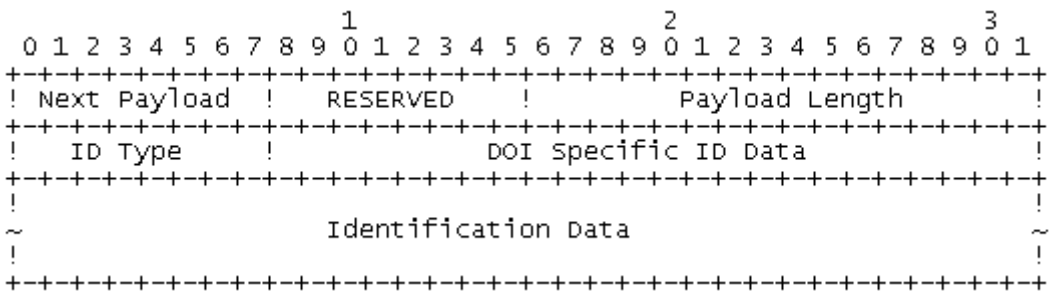

Figura I.5-4 Identification Payload Format.

Nella creazione di un Identification Payload l'entità trasmittente deve (MUST) eseguire i seguenti passi:

Determinare l'informazione di identificazione da utilizzare come definito dal DOI.

Determinare l'utilizzo del campo Identification Data come definito dal DOI.

Costruire l'Identification Payload.

Trasmettere il messaggio all'entità ricevente.

L'entità che, invece, riceve un Identification Payload deve (MUST) eseguire i seguenti passi:

- Determinare se l'Identification Type è supportato. Se ciò non avviene, si eseguono i seguenti passi:

- L'evento INVALID ID INFORMATION dovrebbe (SHOULD) essere segnalato nei log.

- Si invia all'entità trasmittente un Informational Exchange con un Notification Payload contenente il messaggio INVALID_ID_INFORMATION.

CAP II: PUBLIC KEY INFRASTRACTURE

La crittografia a chiave pubblica è un sistema matematico che viene impiegato per cifrare informazioni servendosi di due chiavi: una segreta, chiamata chiave privata, e una che può essere distribuita pubblicamente, ovvero la chiave pubblica. Entrambe non sono altro che sequenze numeriche, lunghe tipicamente diverse centinaia di caratteri. Aspetto molto importante è invece che vengono create a coppie e che qualsiasi cosa protetta con la chiave pubblica può essere sprotetta solo con quella privata.

Andando indietro nel tempo fino al 1970, gli ideatori di questo tipo di crittografia ritenevano che gli utenti avrebbero un giorno pubblicato le proprie chiavi pubbliche in un elenco simile a quello telefonico. Poco dopo la creazione della crittografia a chiave pubblica sorse però una questione: chi avrebbe assicurato che la chiave stampata nell'elenco pubblico vicino al nome "Tizio" appartenesse veramente a quest'ultimo?

Per garantire la provenienza delle informazioni ci si affida oggi a un aspetto della crittografia a chiave pubblica definito firma digitale[3]. In sostanza, anziché criptare un messaggio con la chiave pubblica di Tizio e mandarglielo, il mittente lo firma con la propria chiave privata. Essendo il messaggio leggibile solo con la sua chiave pubblica, il destinatario sarà certo della sua provenienza. L'azienda che si occupa degli elenchi con le chiavi pubbliche li firmerà quindi anch'essa digitalmente con la propria chiave privata. In tal modo prima di utilizzare la chiave del destinatario per mandare un messaggio, ne si verificherà prima l'autenticità usando la copia della chiave pubblica di chi rilascia l'elenco. Nella pratica, chi pubblica tali elenchi firma l'intera ogni singola voce (i certificati) composta dai dati della persona e dalla sua chiave pubblica. Il tutto sfruttando formati quali X.509v3. Chi rilascia i certificati è invece chiamato "certification authority" (CA) e attualmente l'intero sistema di directory, certificati e CA viene raccolto sotto la denominazione di PKI (Public Key Infrastructure).

[3] Per ulteriori riferimenti sulla Firma digitale vedere l'APPENDICE A

II.1 Certificati Digitali

I certificati digitali svolgono una funzione essenziale nella crittografia a chiave pubblica. Infatti, in tale tecnologia, sia in fase di cifratura sia in fase di verifica di una firma digitale occorre conoscere la chiave pubblica o del destinatario di un messaggio oppure del firmatario del messaggio firmato. In entrambi i casi, il valore delle chiavi pubbliche non è confidenziale; la criticità del reperimento delle chiavi sta nel garantire non la confidenzialità, ma l'autenticità delle chiavi pubbliche, ossia sta nell'assicurare che una certa chiave pubblica appartenga effettivamente all'interlocutore per cui si vuole cifrare o di cui si deve verificare la firma. Se, infatti, una terza parte prelevasse la chiave pubblica del destinatario sostituendola con la propria, il contenuto dei messaggi cifrati sarebbe disvelato e le firme digitali potrebbero essere falsificate. In altre parole, un certificato digitale permette di autenticare un individuo, certificando che la chiave pubblica, in esso contenuta, appartenga realmente al soggetto per il quale è stato rilasciato.

Ogni certificato ha una struttura dati costituita almeno dalle seguenti informazioni:

- informazioni che identificano univocamente il possessore di una chiave pubblica (ad esempio il nome);
- il valore della chiave pubblica;
- il periodo di validità temporale del certificato;
- la firma digitale dell' autorità di certificazione con cui si assicura autenticità della chiave ed integrità delle informazioni contenute nel certificato;

La semplicità del meccanismo di distribuzione delle chiavi è diretta conseguenza delle caratteristiche stesse dei certificati: i certificati, infatti, possono essere distribuiti senza dover necessariamente ricorrere ai tipici servizi di sicurezza di confidenzialità, integrità, e autenticazione delle comunicazioni.

Per le proprietà della crittografia a chiave pubblica non c'è infatti alcun bisogno di garantire la riservatezza del valore della chiave pubblica; durante il processo di distribuzione, poi, non ci sono requisiti di autenticazione ed integrità dal momento che il certificato è per sua costituzione una struttura già protetta (la firma digitale dell'autorità di certificazione sul certificato fornisce, infatti, sia autenticazione sia integrità).

Se, quindi, un intrusore tentasse, durante la pubblicazione del certificato, di alterarne il contenuto, la manomissione sarebbe immediatamente rilevata in fase di verifica della firma sul certificato; il processo di verifica fallirebbe e l'utente finale sarebbe avvertito della non integrità della chiave pubblica contenuta nel certificato.

Le caratteristiche stesse, quindi, del certificato permettono di distribuire i certificati a chiave pubblica anche mediante canali non sicuri (file server insicuri o sistemi di directory o protocolli di comunicazione intrinsecamente insicuri).

Un certificato è, a tutti gli effetti, un documento d'identità digitale e come tale richiede la necessità di una autorità di certificazione ufficialmente riconosciuta dalla società, che garantisce l'autenticità delle informazioni in esso contenute. Tale entità prende il nome di *Certification Authority* (CA). Il processo attraverso il quale un certificato viene generato è il seguente:

l'utente compila una richiesta di certificato con i suoi dati e la sua chiave pubblica. La richiesta di certificato può essere effettuata, ad esempio, usando lo standard PKCS #10 (Appendice **Errore. L'origine riferimento non è stata trovata.**).

inviare all'autorità di certificazione (CA) la richiesta di certificazione. Quest'ultima verifica l'autenticità dei dati e, se il responso è positivo, produce un certificato e lo rilascia al richiedente firmandolo con la propria chiave privata.

Il richiedente può ora inviare il proprio certificato ad un altro individuo per farsi autenticare e per consegnargli la propria chiave pubblica. La verifica dell'identità dell'individuo avviene mediante il controllo della firma apposta sul certificato dall'autorità di certificazione che, pertanto, mette a disposizione di tutti la propria

chiave pubblica. Più precisamente, mette a disposizione il proprio certificato autofirmato, o firmato da un'altra autorità di certificazione.

II.1.1 Lo Standard X.509 per i certificati

Lo standard ormai diffusamente riconosciuto di definizione del formato dei certificati è quello descritto nello standard X.509 ISO/IEC/ITU [RFC2459, RFC 3280]. Tale standard è giunto alla terza versione (X.509v3), dopo che le precedenti due versioni si erano dimostrate insufficienti a risolvere certe problematiche. Vediamo da quali campi è composto un certificato:

1. **TbsCertificate** *field***,** che contiene le seguenti informazioni:
   - *Version*: indica la versione del formato del certificato (1, 2 o 3).
   - *Serial number*: è un intero assegnato dalla CA, che identifica univocamente il certificato tra tutti i certificati emessi dall'Autorità di Certificazione.
   - *Signature*: specifica l'algoritmo utilizzato dalla CA per firmare il certificato. Tale campo deve contenere lo stesso "Algorithm Identifier", presente nel campo **SignatureAlgorithm** *field* del certificato.
   - *Issuer*:identifica l'entità che ha firmato ed emesso il certificato. Deve (MUST) contenere un *Distinguished Name* (DN) non vuoto.
   - *Validity*: specifica la data e l'ora di inizio e la data e l'ora di fine della validità del certificato.
   - *Subject*: identifica l'entità che ha richiesto il certificato e a cui è associata una chiave pubblica specificata nel campo *SubjectPublicKeyInfo*. Quando non è vuoto tale campo deve (MUST) contenere un Distinguished Name (DN).

- *SubjectPublicKeyInfo*: contiene il valore della chiave pubblica del possessore del certificato e l'algoritmo con cui tale chiave viene usata.
- *IssuerUniqueID* e *SubjectUniqueID*: questi campi, opzionali,sono presenti nel certificato per avere la possibilità di riusare il *subject* e/o lo *issuer name* nel tempo.
- *Extensions:* in tale campo sono presenti le estensioni che, se presenti, forniscono informazioni aggiuntive relative all'utente o alle chiavi pubbliche e che permettono la gestione della gerarchia di certificazione.

2. **SignatureAlgorithm** *field*: contiene l'identificativo per l'algoritmo crittografico usato dalla CA per firmare il certificato.

3. **SignatureValue** *field*: contiene la firma digitale (*digital signature*) generata dall'elaborazione del **TbsCertificate** *field.* Generando tale firma, una CA certifica il legame tra la chiave pubblica e il soggetto del certificato.

Nella versione 3, rispetto alle precedenti, è stato aggiunto un ulteriore campo, *Extensions* (nel **TbsCertificate** *field*); che serve ad associare attributi addizionali agli utenti o alle chiavi pubbliche e per la gestione della gerarchia di certificazione. Inoltre, il formato del certificato X.509 v3 permette alle comunità di definire anche delle estensioni private che forniscono informazioni uniche a quelle comunità. Tale campo è suddiviso in tre sottoinsiemi:

- l'identificatore del tipo di estensione.
- un indicatore di criticità.
- il valore effettivo dell'estensione.

L'indicatore di criticità facilita l'interoperabilità tra sistemi che non utilizzano certificati con determinate estensioni e sistemi che, invece, interpretano tutte le

estensioni definite a livello di standard. L'indicatore di criticità è semplicemente un flag che stabilisce se l'estensione è critica o non critica; se l'estensione non è critica, significa che il sistema che deve elaborare il certificato può eventualmente ignorare l'estensione in questione se non è in grado di interpretarla.

Le estensioni standard definite nei certificati X.509v3 si suddividono in quattro gruppi:

- Estensioni contenenti informazioni sulla chiave pubblica: specificano il tipo di utilizzo per cui è stata emessa una certa chiave.
- Estensioni contenenti informazioni aggiuntive relative all'Autorità di Certificazione e all'utente possessore del certificato (ad esempio nomi alternativi per la CA e per l'utente).
- Estensioni contenenti informazioni sulle politiche di emissione e sulle finalità di utilizzo dei certificati (le estensioni *Certificate Policy* e *Policy Mapping*).
- Estensioni contenenti informazioni sui vincoli di spazio dei nomi o di politica da imporre durante il ritrovamento o la verifica di un certificato appartenente ad un dominio di fiducia esterno.

Tra le possibili estensioni previste, la RFC 3280, prevede che la CA debba (MUST) supportare le seguenti estensioni:

1. *Key Identifiers*: le estensioni di questo tipo sono due:
   - *Authority key Identifier*: fornisce un modo per identificare la chiave pubblica corrispondente alla chiave privata usata per firmare un certificato. Questa estensione viene usata quando l'entità mittente del certificato ha chiavi di firma multiple. L'identificazione si può (MAY) basare sia sul *key identifier* oppure sullo *issuer name* e sul *serial number*. Tale campo deve (MUST) essere contenuto in tutti i certificati generati in modo da rendere più semplice la costruzione del cammino di certificazione (C*ertification Path*).

- *Subject Key Identifier*: rappresenta un mezzo per identificare i certificati che contengono una particolare chiave pubblica. Come la precedente, anche tale estensione deve (MUST) essere contenuta in tutti i certificati generati in modo da rendere più semplice la costruzione del cammino di certificazione (*certification path*).

1. *Basic Constraints: identifica se il soggetto del certificato è la CA e individua la massima profondità dei certification paths validi che includono il certificato in questione.*

Key Usage: definisce lo scopo della chiave contenuta nel certificato. Tale campo può assumere diversi valori tra cui ricordiamo i seguenti:

Certificate Policies: contiene una sequenza di due o più informazioni relative alla policy. Nel caso del certificato di una end entity (soggetto che richiede un certificato), queste informazioni indicano la policy sotto la quale il certificato è stato emesso e lo scopo per cui il certificato deve essere usato. Nel caso del certificato della CA, invece, queste informazioni limitano il set di policies per i *certification paths* che includono questo certificato.

Se la CA emette certificati con un *Subject* field vuoto, allora la CA deve (MUST) supportare anche l'estensione:

Subject Alternative Name: tale estensione permette di legare al soggetto del certificato identità addizionali. Le opzioni definite includono:

- *Internet mail address*: l'indirizzo deve (MUST) essere incluso come un rfc822name.
- *IpAddress*: l'indirizzo deve (MUST) essere inserito sotto forma di un "octet string" in "network byte order".
- *Domain Name System (DNS)*: il nome del dominio deve (MUST) essere inserito nel dNSName (è una IA5String).

- *URI*: il nome deve (MUST) essere inserito nello UniformResourceIdentifier (una IA5String).

  Esistono anche altre opzioni che includono definizioni completamente locali.
  Inoltre, poiché questo campo è considerato essere definitivamente legato alla chiave pubblica, tutte le sue parti devono (MUST) essere verificate dalla CA. In aggiunta, se l'unica identità del soggetto inclusa nel soggetto è un *Alternative Name Form* (per esempio, un indirizzo e-mail), allora il DN del *subject* deve (MUST) essere vuoto e l'estensione *SubjectAltName* deve (MUST) essere presente.
  Quando il *SubjectAltName* contiene un DN, questo deve (MUST) essere unico per ciascun soggetto certificato da una CA come definito dallo *issuer name* field. Una CA potrebbe (MAY) emettere più di un certificato con lo stesso DN allo stesso *subject*.
  Infine, tale campo può (MAY) contenere altri tipi addizionali attraverso l'uso del campo *otherName*. Il formato e la semantica del tipo sono indicati da un OID nel campo *type-ID*.

Il supporto delle altre estensioni da parte della CA è opzionale.

Analizziamo adesso le possibili estensioni che un'Implementazione deve (MUST) supportare (RFC 3280):

1. *Key Usage.*
2. *Certificate Policies.*
3. *SubjectAlternativeName*
4. *Basic Constraints.*
5. *Name Constraints*: deve (MUST) essere usata solo nel certificato della CA ed indica lo spazio in cui devono (MUST) essere collocati tutti i nomi del soggetto nei certificati sottoseguenti in un *certification path*.

6. *Policy Constraints*: può essere usata nei certificati emessi dalla CA. Si può utilizzare per proibire la *policy mapping* o per richiedere che ciascun certificato in un *path* contenga un *policy identifier* accettabile.
7. *Extended Key Usage*: indica uno o più scopi per cui la chiave pubblica certificata può essere usata, in aggiunta o al posto degli scopi di base indicati nel campo *Key Usage*. Solitamente, tale campo compare solo nei certificati delle entità richiedenti il certificato e, se presente, il certificato deve (MUST) essere usato solo per gli scopi indicati.

Inoltre, tale profilo raccomanda alle implementazioni di usare :

1. *Authority Key Identifier.*
2. *Subject Key Identifier.*

Un esempio della visualizzazione che Microsoft Windows fornisce per analizzare un certificato X.509, da cui possiamo vedere i campi e le estensioni che lo compongono, è fornito dalle seguenti tre figure:

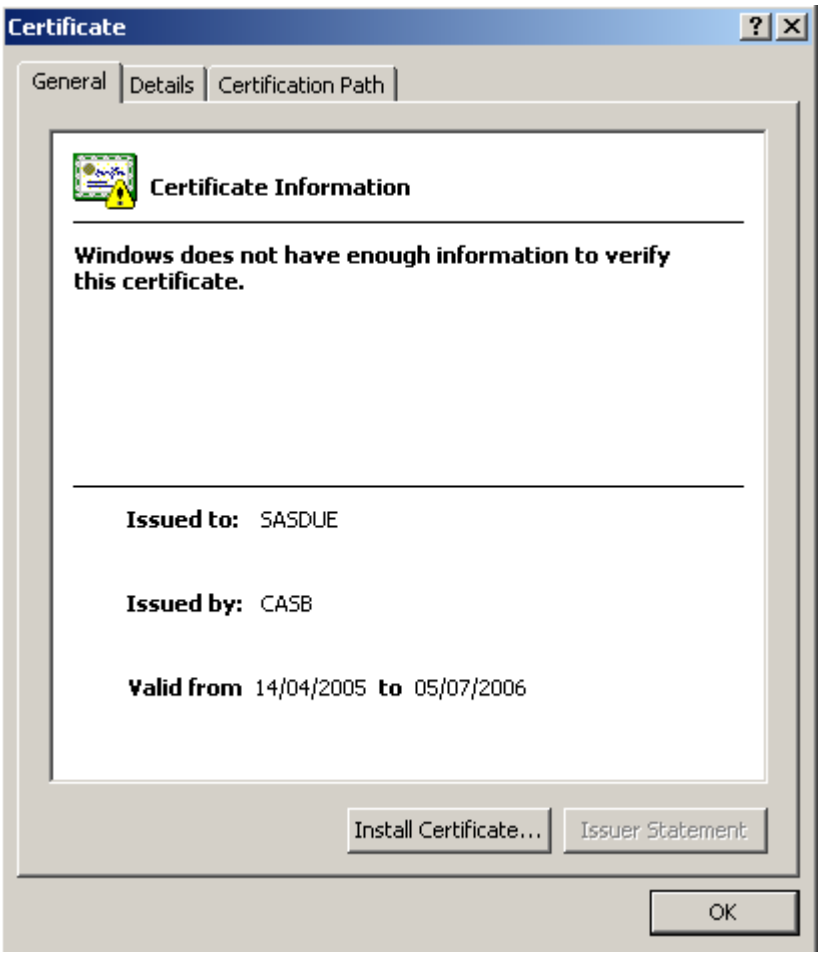

Figure II.1: Esempio di certificato X.509v3

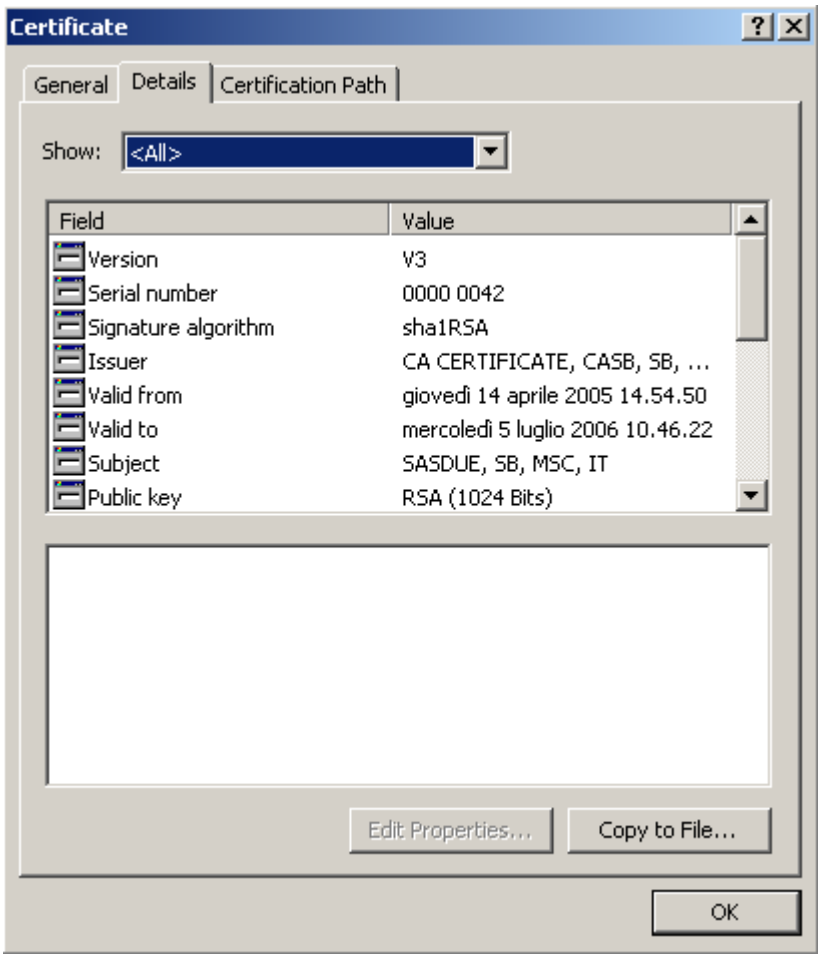

Figure II.2: Esempio di certificato X.509v3

II.2 PKI – Infrastrutture a Chiave Pubblica

Le infrastrutture a chiave pubblica (*Public Key Infrastructure*, *PKI*) forniscono il supporto necessario affinchè la tecnologia di crittografia a chiave pubblica sia utilizzabile su larga scala. Le infrastrutture offrono servizi relativi alla gestione delle chiavi e dei certificati e delle politiche di sicurezza. Le autorità di certificazione e la problematica di gestione dei certificati elettronici costituiscono, infatti, il cuore delle infrastrutture a chiave pubblica.

Una infrastruttura a chiave pubblica introduce il concetto di *third-party trust*, ossia di quella situazione che si verifica quando due generiche entità si fidano implicitamente l'una dell'altra senza che abbiano precedentemente stabilito una personale relazione di fiducia. Questo è possibile perchè entrambe le entità condividono una relazione di fiducia con una terza parte comune.

Una PKI è una rete di comunicazione in cui la sicurezza è gestita ricorrendo all'uso di CHIAVI PUBBLICHE per abilitare servizi quali:

- AUTENTICAZIONE: consente ad una entità di dimostrare la propria identità e/o di verificare quella dell'interlocutore;
- CONTROLLO ACCESSI: finalizzato ad ostacolare il tentativo di utilizzo di una risorsa senza autorizzazione;
- NON RIPUDIO: impossibilità di rifiuto da parte di una entità di essere la sorgente dei dati;
- RISERVATEZZA DEI DATI: cifratura;

In questi sistemi la singola entità è identificata dalla Chiave Pubblica e la garanzia che una data Chiave Pubblica appartenga effettivamente all'entità presunta è fornita dai Certificati Digitali. Il Certificato Digitale è un documento informatico che identifica univocamente la singola entità della PKI cui appartiene, ne contiene la Chiave Pubblica, non è falsificabile e puó quindi essere distribuito senza creare limitazioni al sistema di sicurezza.

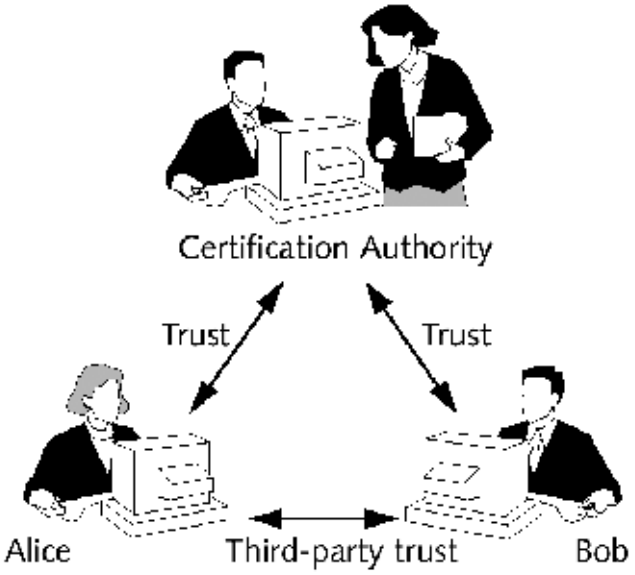

Figure II.3: Third-party trust.

Affinché due entità possano comunicare con i servizi di sicurezza sopra indicati è sufficiente che ognuna possegga il Certificato Digitale dell'altra.

La libera distribuzione dei Certificati Digitali permette la diffusione della Chiave Pubblica mettendo quindi i possessori in grado di comunicare con sicurezza con l'entità certificata.

Il funzionamento di una PKI si basa sull'uso dei Certificati Digitali, la cui gestione è affidata ad una specifica autorità: la *Certification Authority* (CA) che è quindi un SERVER con il compito di garantire l'identità di ogni entità appartenente alla PKI fornendo, pubblicando, revocando e sospendendo i Certificati Digitali.

Ogni CA ha poi l'obbligo di mantenere un DataBase (*Repository*) dove sia possibile rintracciare, da parte delle altre entità della PKI oppure di altre CA subordinate, tutti i certificati da lei rilasciati.

Per garantire i certificati, la CA deve conoscere tutte le entità della PKI e viceversa, ogni entità della PKI, per essere sicura che il proprio certificato sia rilasciato dalla CA, deve conoscere la CA.

Il compito di accertare l'identità delle entità di una PKI è gestito da una ulteriore autorità: la *Registration Authority* (RA) che, tramite apposite procedure, si fa carico ed informa la CA e le singole entità della PKI sulle rispettive identità.

La struttura semplificata di un modello architetturale per una PKI risulta pertanto composta dai seguenti elementi:

- **END-ENTITY**: singole entità connesse nella PKI. Gli utenti finali (*end-entity*) sono dotati del software in grado di interagire con la PKI in tutte le fasi in cui sia richiesta un'interazione tra le applicazioni client e la CA o la directory (ad esempio un'interazione fondamentale interviene nella fase di inizializzazione dell'utente, fase nella quale vengono creati i relativi certificati di cifratura e di firma).

- **CA**: Autorità che certifica le Chiavi Pubbliche delle singole End-Entity. In particolare, una ***Certification Authority*** (CA) è un ente pubblico o privato che, come terza parte fidata (*trusted third party*) emette certificati in cui attesta il legame tra l'identità di un utente e la sua chiave pubblica. Le principali funzioni svolte da una CA sono:

  - Autenticazione dell'identità dei richiedenti
  - Validazione delle richieste di certificati
  - Emissione dei certificati
  - Gestione di un archivio dei certificati
  - Pubblicazione dei certificati emessi
  - Pubblicazione dell'elenco dei certificati revocati o sospesi (black-list)
  - Riemissione dei certificati alla scadenza

Come accennato in precedenza, una CA può avvalersi di organizzazioni esterne chiamate *Registration Authority* (RA) per l'identificazione degli utenti che richiedono l'emissione di un certificato.
Il ricorso ad una CA è conveniente per tutte le applicazioni che richiedano l'utilizzo di un certificato. Non basta infatti ottenere la generazione di un certificato (che del resto potrebbe essere fatta in proprio dal titolare) ma è necessario poter distribuire a tutti gli interessati (pubblicare) il certificato, garantendone contemporaneamente la validità. In assenza di una terza parte fidata, quale è una CA, ogni utilizzatore di certificati dovrebbe scambiare il proprio certificato con tutte le persone con cui, ad esempio, desidera inviare messaggi sicuri.

- **RA**: Autorità che garantisce l'identità della CA verso le End-Entity e viceversa. In particolare, La ***Registration Authority*** è preposta alla registrazione delle end-entity per metterle in grado di interagire con la CA. Una RA assolve generalmente alle seguenti funzionalità:

  - autenticazione e registrazione delle end-entity
  - assegnazione di nomi e password alle end-entity
  - inizializzazione dei token per le end-entity (generazione delle chiavi, richiesta di certificato, memorizzazione del certificato, scrittura sul token)
  - key archive e recovery
  - Validazione delle Richieste di Certificato effettuate dalle end-entity e da inviare alla CA e invio alle end-entity delle relative risposte
  - Supporto di protocolli per comunicare con le end-entity
  - Quando riceve certificati rilasciati dalla CA, li inoltra sia tramite la RA GATEWAY che tramite Operatore
  - richiesta di revoca e/o sospensione di certificati
  - Mantiene un audit log, firmato su ogni entry, di tutte le operazioni compiute dalla RA.

- **REPOSITORY**: sistema di distribuzione di certificati verso le End-Entity.E' un DataBase dove sia possibile rintracciare, da parte delle altre entità della PKI oppure di altre CA subordinate, tutti i certificati da lei rilasciati.

I legami fra i quattro elementi suddetti sono illustrati in figura II.4
Tali legami individuano le seguenti funzionalità:

a) REGISTRAZIONE: processo per cui una end-entity fa conoscere se stessa (tramite la RA) alla CA e viceversa.
INIZIALIZZAZIONE: prima che una END-ENTITY o una CA possa operare nella PKI, è necessario che possegga una propria chiave pubblica.
CERTIFICAZIONE: processo tramite cui una END-ENTITY richiede un certificato e la CA lo rilascia e lo ritorna al richiedente e/o lo memorizza nella

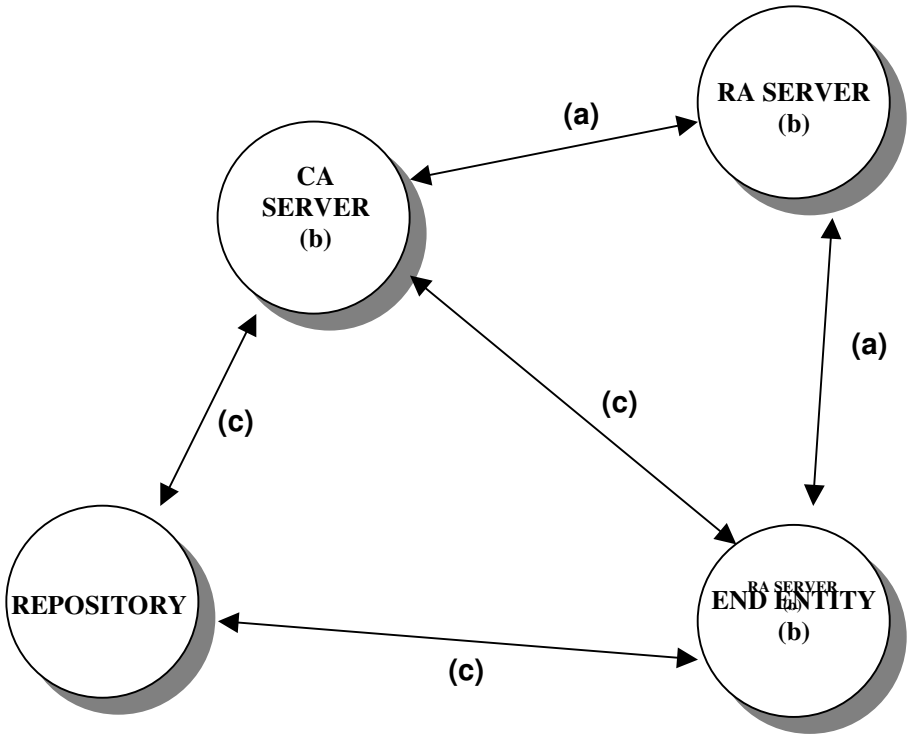

Figure II.4: Struttura base di una PKI.

II.2.1 Fase di Registrazione

La Registrazione è il processo che permette di inserire nuove entità nella PKI.

Nel processo di Registrazione la nuova entità deve fornire i dati che ne permettano il controllo di identità e, quale controparte, le vengono forniti i dati per il collegamento alla CA ed alla Repository.

A partire dai dati di registrazione, la nuova entità potrà successivamente inizializzarsi con la CA e diventare cosí membro della PKI.

Tutto il processo è gestito da una specifica autorità: la Registration Authority (RA).

Le funzioni supportate da una RA sono, generalmente: Autenticazione Personale, distribuzione di Token, Revoca di Certificati.

Il principio base tipico su entità di natura umana è il seguente:

- il richiedente si reca all'ufficio delegato della RA e riempie i moduli che gli vengono proposti;
- quindi l'addetto dell'ufficio RA invia la documentazione all'ufficio centrale della RA;
- la RA accerta i dati ricevuti e genera l'Authentication Key (o Challenge Password) con cui la nuova entità potrà comunicare con la CA;
- il richiedente riceverà i codici di accesso con cui potrà instaurare il collegamento con la CA (ad esempio Authentication key e reference value, indirizzo della CA e indirizzo della Repository), procedere alla inizializzazione del dispositivo di firma e ad effettuare la richiesta del Certificato Digitale;
- la RA comunica alla CA i dati del nuova entità (Challenge password);
- ulteriore compito della RA sarà richiedere la revoca dei certificati emessi dalla CA ogniqualvolta vengano meno i requisiti in base ai quali i certificati sono stati rilasciati.

In tutti i casi la CA rimane la sola autorità responsabile dell'identificazione del richiedente il certificato.

II.2.2 Fase di Inizializzazione

La fase di *Inizializzazione* di una entità, provvede alla generazione dei dati necessari affinché tale entità sia totalmente riconosciuta all'interno della PKI.

Il possesso del Certificato Digitale generato dalla CA della PKI è la condizione necessaria e sufficiente, pertanto, dal momento che il Certificato Digitale è costruito sulle chiavi RSA, l'operazione di Inizializzazione si compone delle seguenti fasi:

- generazione, all'interno della nuova entità, di una chiave RSA;
- compilazone, da parte della nuova entità, della richiesta del certificato digitale utilizzando la parte pubblica della chiave RSA;
- Composizione della richiesta del certificato.

L'operazione di inizializzazione puó essere compiuta sia internamente alla RA, in tal caso la end-entity riceve dalla RA direttamente il token già inizializzato, oppure esternamente alla RA ed in tal caso la RA deve fornire la end-entity delle informazioni necessarie per compilare la richiesta di certificato.

II.2.3 Fase di Certificazione

Una entità di una PKI ha necessità di una chiave RSA per comunicare con sicurezza con le altre entità. La parte pubblica della chiave RSA deve essere certificata dalla CA per assicurare chi la usa sull'effettiva identità del proprietario della chiave.

L'operazione di validazione che la CA effettua sulle chiavi pubbliche è detta *Certificazione* ed è finalizzata alla emissione di un *Certificato Digitale*.

L'operazione di Certificazione è composta dalle eseguenti fasi:

- collegamento, da parte della end-entity (utilizzando i dati forniti dalla RA per autenticarsi), con la CA ed invio della Richiesta di Certificato composta nella fase di inizializzazione;
- Controllo, da parte della CA, della validità della end-entity richiedente;

- Emissione, da parte della CA, del Certificato Digitale e sua pubblicazione nella Repository;
- Collegamento, da parte della end-entity, con la CA o con la Repository della CA per il reperimento del Certificato e del Certificato della CA che lo ha firmato.

Anche il Certificato stesso della CA non sfugge a questa regola.

Una CA puó AUTOCERTIFICARE il proprio certificato digitale, oppure appoggiarsi ad un'altra CA che lo certifica, creando in tal modo una catena di Certificati che termina necessariamente con un Certificato Autofirmato.

II.3 Ciclo di vita dei Certificati

L'emissione effettiva di un certificato elettronico da parte di un'Autorità di Certificazione a favore di un utente finale deve essere preceduta come accennato già in precedenza da una fase di registrazione dell'utente richiedente il certificato elettronico. Attraverso il processo di registrazione l'utente richiedente un servizio di certificazione si identifica presso l'autorità preposta al servizio di registrazione; le credenziali che in questa fase l'utente deve sottoporre all'Autorità di Registrazione dipendono fortemente dalle modalità e procedure di registrazione definite nell'ambito di una politica di sicurezza.

Il processo di registrazione stabilisce una relazione iniziale tra utente finale ed Autorità di Certificazione; l'utente finale, una volta attestata l'autenticità della sua identità, viene registrato nel dominio di fiducia gestito dalla CA. L'obiettivo, di primaria importanza del processo di registrazione è, quindi, quello di garantire che la chiave pubblica, di cui un certo utente finale richiede la certificazione, appartenga effettivamente al nome del richiedente.

Terminata la fase di registrazione, l'utente puó richiedere l'emissione di un certificato elettronico. La procedura di generazione di un certificato elettronico consiste dei seguenti passi:

- l'utente finale sottopone all'autorità di certificazione le informazioni da certificare.
- l'Autorità di Certificazione puó verificare l'accuratezza delle informazioni presentate in accordo a politiche e standard applicabili.

L'Autorità di Certificazione firma le informazioni generando il certificato e lo pubblica sul sistema di Directory; di solito, la CA archivia una copia del certificato sul suo database privato; ogni operazione di generazione di certificati elettronici viene registrata su un archivio di registrazione dati.

Ogni certificato elettronico generato ha una validità temporale limitata al cui termine va sostituito; il periodo di validità di un certificato, in assenza di compromissioni o usi illeciti, garantisce l'utente che deve utilizzare tale certificato che la chiave pubblica possa essere utilizzata per lo scopo per cui è stata generata e che l'associazione tra la chiave pubblica e le altre informazioni contenute nel certificato sia ancora valida.

II.6.2 Rrevoca e Sospensione di un certificato

In ogni momento, all'interno del periodo di validità di un Certificato, questo puó essere REVOCATO, ovvero reso non idoneo all'uso.

L'operazione di revoca di un certificato costituisce una fase della gestione del ciclo di vita dei certificati di alta criticità. Il certificato elettronico deve essere revocato in presenza delle seguenti condizioni:

- compromissione rilevata o semplicemente sospettata della chiave privata corrispondente alla chiave pubblica contenuta nel certificato.
- cambiamento di una qualsiasi delle informazioni contenute nel certificato elettronico o delle condizioni iniziali di registrazione.

La revoca del certificato elettronico è effettuata dall'Autorità di Certificazione e generalmente viene avviata su richiesta dello stesso utente finale.

Tale operazione provoca l'inserimento del Certificato (tramite il SERIAL NUMBER) in un particolare elenco denominato CRL (*Certificate Revocation List*) che la CA pubblica nella REPOSITORY e che tutte le end-entity possono quindi richiedere e consultare.la gestione di tali liste è delegata alla CA; nell'ambito del dominio amministrato, ogni CA pubblica periodicamente una struttura dati contenente l'elenco dei certificati revocati, ossia una lista, firmata digitalmente dall'Autorità di Certificazione, che riporta i certificati revocati, la data temporale in cui è avvenuta la revoca ed eventualmente il motivo della revoca.

Come già detto la revoca avviene riportando nella CRL il SERIAL NUMBER del Certificato.

Il Serial Number è quindi un campo molto importante del Certificato Digitale. Ogni Serial Number è unico all'interno della stessa CA ed individua univocamente il dato Certificato.

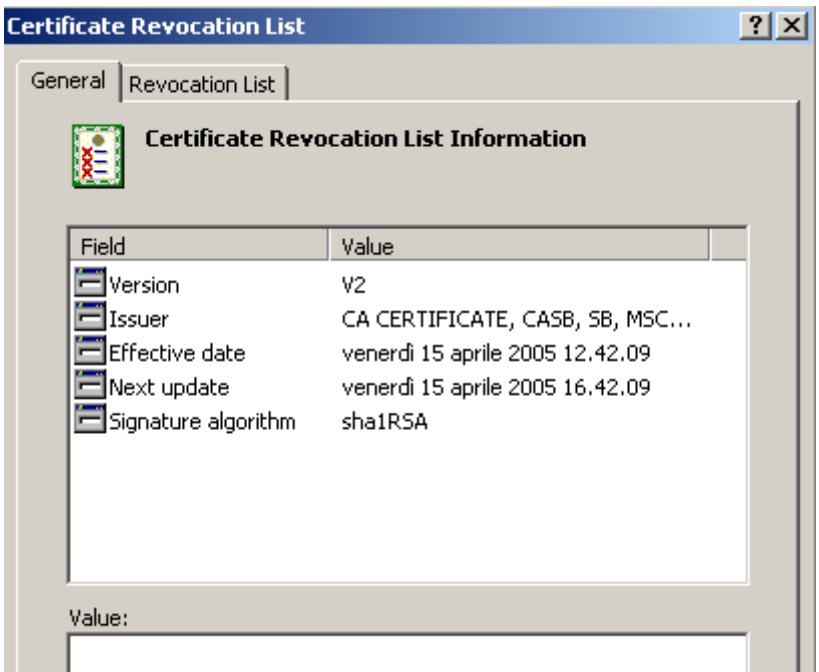

Figure II.5: Esempio di CRL

La CA mette a disposizione delle end-entity una funzione che ritorna la validità del dato Certificato. Questo puó essere utile perché permette di controllare la validità senza scaricare ed archiviare tutta la CRL.

Il processo di Revoca di un Certificato può esserequindi riassunto nelle seguenti fasi:

- la RA richiede alla CA la REVOCA del Certificato (tramite la Challenge Password)
- la CA effettua la revoca ed emette una nuova CRL
- la CA memorizza la CRL nella Repository
- successivamente le end-entity si fanno carico di scaricare periodicamente le CRL dalla Repository per controllare la validità delle entità con cui entrano in contatto

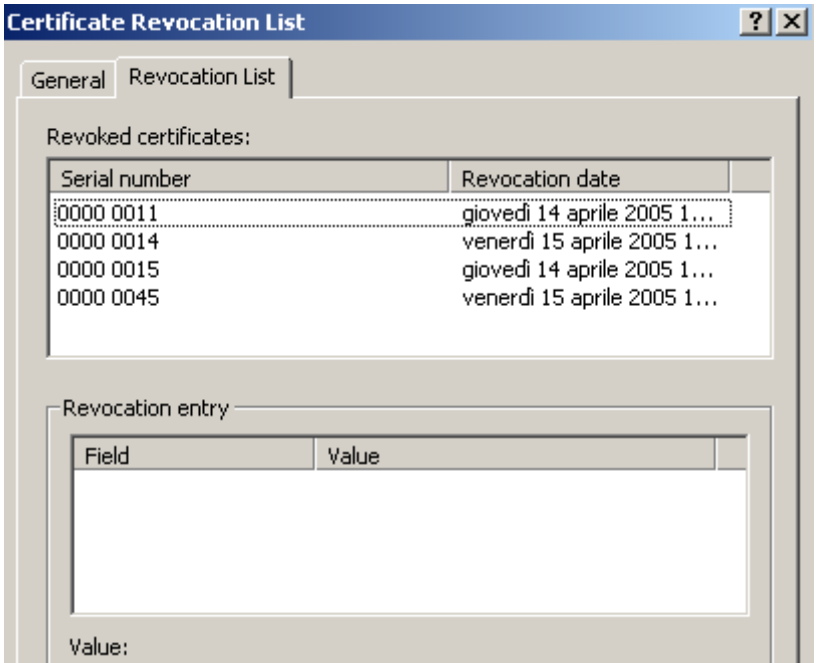

Figure II.6: Esempio di CRL

Insieme alla funzionalità di REVOCA viene supportata quella di SOSPENSIONE.

Un certificato sospeso è TEMPORANEAMENTE non idoneo all'uso.

Il processo di Sospensione si compone delle stesse fasi di quello di Revoca e l'operazione provoca l'inserimento del certificato in un elenco denominato CSL (Certificate Suspension List) che la CA pubblica nella REPOSITORY e che tutte le end-entity possono quindi richiedere e consultare.

La CSL riporta quindi le date di inizio e fine sospensione e lo stato di sospeso termina automaticamente senza la necessità di una nuova emissione di CSL.

II.4 Certification Paths

Se si potesse disporre di un'unica Autorità di Certificazione su scala globale, il problema della distribuzione, del reperimento e della verifica della validità delle chiavi pubbliche non sussisterebbe; tuttavia una tale soluzione non è praticabile per motivi di scalabilità, flessibilità e sicurezza. Diventa, quindi, inevitabile ricorrere ad un modello

costituito da Autorità di Certificazione multiple tra loro concatenate secondo differenti modelli organizzativi, detti anche modelli di fiducia.

In uno scenario costituito da una molteplicità di Autorità di Certificazione su larga scala, strutturate secondo un certo modello organizzativo, non è pensabile che ogni utente abbia diretta conoscenza delle chiavi pubbliche di ogni potenziale interlocutore, sotto forma di certificato elettronico, o delle chiavi pubbliche delle corrispondenti autorità di certificazione competenti. Occorre, quindi, disporre di un meccanismo corretto di ritrovamento dei certificati elettronici degli interlocutori appartenenti a domini di sicurezza esterni. Il modello generale su cui si basano tutti i sistemi di distribuzione su larga scala delle chiavi pubbliche sotto forma di certificati elettronici, utilizza le cosiddette catene di certificazione, altrimenti conosciute come cammini di certificazione (Certification Paths).

Il problema del ritrovamento di un cammino di certificazione consiste sostanzialmente nel trovare, se esiste, un cammino di certificazione che permetta di verificare l'autenticità del certificato elettronico di uno specifico utente remoto a partire da un insieme di chiavi pubbliche, assunte come radici del cammino, di cui si ha diretta e sicura conoscenza; la risoluzione del problema, quindi, assume per date certe condizioni iniziali: tali condizioni iniziali si identificano nelle chiavi di specifiche autorità di certificazione.

In **Errore. L'origine riferimento non è stata trovata.** viene riportato un esempio di catena di certificati: la presenza del certificato 1 garantisce Alice dell'autenticità della chiave pubblica dell'autorità di certificazione B; Alice può, quindi, utilizzare la chiave pubblica di B per verificare l'autenticità del certificato 2; la presenza del certificato 2 garantisce ora Alice dell'autenticità della chiave pubblica dell'autorità di certificazione C; ora Alice può utilizzare la chiave pubblica dell'autorità di certificazione C per verificare l'autenticità della chiave pubblica di Bob. Se fosse mancato il certificato 2, Alice non avrebbe potuto verificare l'autenticità del certificato elettronico di Bob e conseguentemente avviare con Bob una comunicazione sicura.

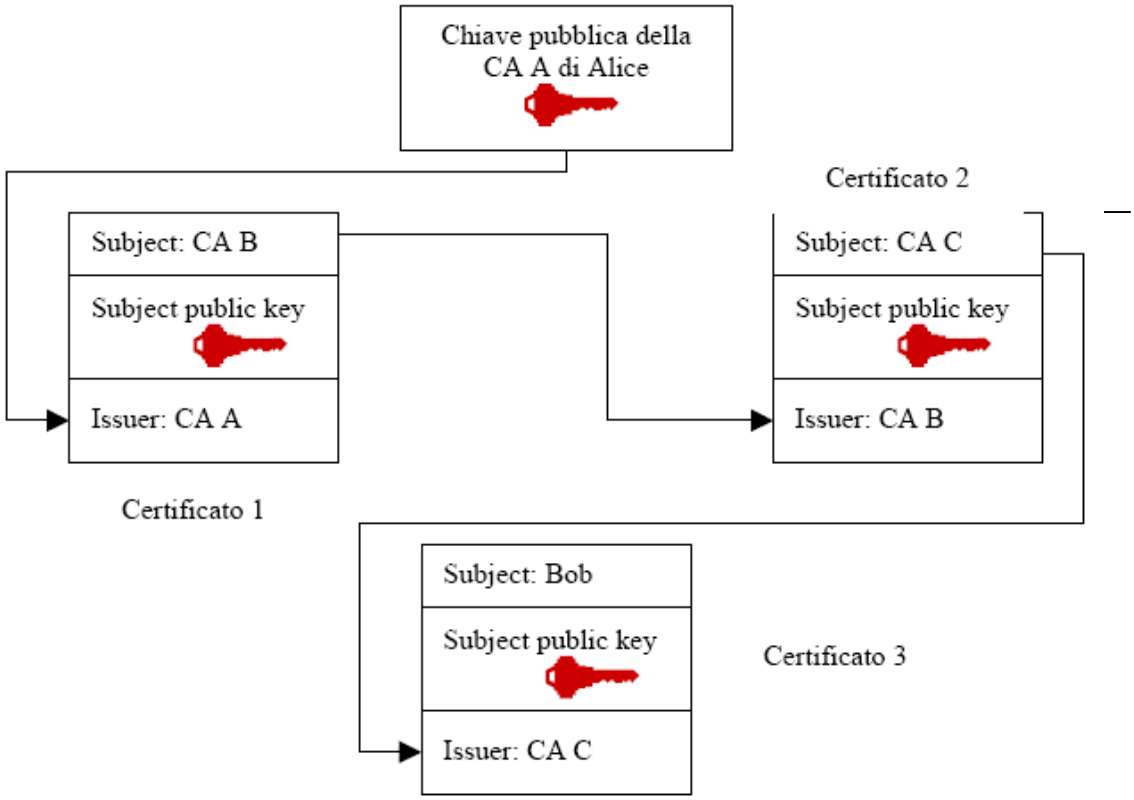

Figure II.7: Esempio di catena di certificati.

Naturalmente i cammini di certificazione dipendono dal modello di fiducia, ossia dal modo in cui le diverse CA sono concatenate. Un tipico modello è quello a struttura gerarchica in cui le CA sono collegate tra loro in una struttura ad albero in cui esiste una CA radice (*Root CA*) che crea canali affidabili tra le varie CA subordinate. Questi canali permettono agli utenti che non sono certificati dalla stessa CA di trovare una CA comune, in modo da stabilire una relazione di fiducia. Nella seguente figura è mostrato un modello gerarchico in cui la CA A è la root CA mentre le B e C sono le CA subordinate. Membri delle CA B e CA C possono fidarsi gli uni degli altri grazie alla CA A. In tal caso si parla di C*ross-Certification*.

Figure II.8: Gerarchia tra CA

CAP III: INTEROPERABILITÀ - I TEST

Il servizio IPSec è per definizione dedicato alla comunicazione e come tale coinvolge necessariamente due o più apparati. Un prodotto IPSec sarà probabilmente in grado di instaurare un canale per lo scambio di dati in maniera sicura con un altro prodotto IPSec a sé stesso identico, poiché identica sarà la procedura di negoziazione del protocollo IKE così come dell'IPSec stesso. Tuttavia gli standard pubblicati non sempre definiscono con precisione tutti i punti di un protocollo, lasciando all'implementatore un certo margine di interpretazione. Questo implica possibili differenze di implementazione dello stesso standard, il che può impedire che un apparato possa comunicare con un prodotto realizzato nei laboratori di un'azienda concorrente.

Il capitolo seguente illustra i test di interoperabilità effettuati sui dispositivi dell' AMTEC S.p.A. in ogni loro aspetto, verranno descritte le problematiche che si sono presentate e le soluzioni a cui si è giunti.

III.1 Scopi

Obiettivo primario di un'azienda che realizza prodotti conformi allo standard IKE/IPSec è quello di ottenere un livello di interoperabilità tra i propri apparati e quelli concorrenti, tale da permettergli un più semplice inserimento dei propri sul mercato.

I test descritti nel seguito del capitolo hanno avuto l'obiettivo di verificare la capacita del router SAS 862/B dell'AMTEC S.p.A di interoperare con altri apparati già presenti sul mercato quali i router della CISCO System.

In particolare, lo scopo del lavoro è stato di indicare al reparto Sviluppo dell'azienda le diverse modalità in cui i due dispositivi risultano interoperabili e le modalità in cui ciò non avviene, portando così alla luce le differenze implementative tra i due apparati.

Per far ciò sono stati studiati approfonditamente tutti gli argomenti relativi all'oggetto del test e il modo in cui l'oggetto del test è attualmente realizzato dal prodotto.

In seguito, sono stati completati tutti i test sul prodotto, è stato necessario modificare volta per volta le configurazioni dei gateway appartenenti alla S-VPN in base alle esigenze del test, utilizzare gli strumenti di analisi opportuni per analizzare i pacchetti che attraversavano la S-VPN

III.2 Descrizione dell' ambiente di Test

In questo paragrafo viene introdotta la descrizione del protagonista dei test, cioè il SAS 862 B, e le descrizioni di tutti gli strumenti utilizzati per il procedimento relativo ai test: le piattaforme, le configurazioni, modificate a seconda dell'esigenza, i log dei due sistemi.

III.2.1 I Protagonisti

I protagonisti principali dei test svolti e descritti di seguito sono il SAS 862/B e il CISCO 2600 series

Il SAS 862 B, prodotto dall'AMTEC S.p.A. è un dispositivo che si comporta da security gateway e da Firewall. Nella sua versione 'B' mette a disposizione due porte ethernet di cui una 10/100 Mbit/s e l'altra a 10 Mbit/s e una interfaccia ISDN.

Figure III.1: SAS 862 B

Il processore Motorola PowerPC MPC866 @ 100MHz con 32 Mbyte di RAM insieme al coprocessore crittografico HIFN, rappresentano le unità di calcolo di cui è

dotato questo apparato. Il sistema operativo che lo gestisce, è il GAIA4 realizzato anch'esso nei laboratori di sviluppo dell'AMTEC S.p.A..

Il router 2600 series della CISCO System nella sua versione base, è disponibile con due slot per le interfacce di rete. Il router utilizzato è il 2621 assemblato con due porte ethernet 10/100 Mbit/s dotato di un processore MPC860 @ 50MHz con 64 MByte di ram e 16 Mbyte di memoria flash

III.2.2 Obiettivi

Prima di iniziare un processo di test è indispensabile schematizzare gli obiettivi ed in base a questi le procedure necessarie per conseguirli. L'interoperabilità IKE/IPSec si può dividere innanzitutto in due punti:

1. la negoziazione e l'instaurazione dei tunnel
2. l'effettiva capacità di trasportare traffico in maniera sicura.

I test effettuati riguardano esclusivamente il primo punto che dipende principalmente dall' implementazione di IKE. In particolare sono stati testati per la fase uno:

- *I diversi ID_TYPE:* i dispositivi sono configurabili in modo da inviare e accettare diversi tipi di identificativi. Tale informazione viene scambiata nel quinto e sesto pacchetto della fase uno per il MAIN MODE e nel primo e secondo pacchetto per l' AGGRSSIVE MODE all' interno dell' IDENTIFICATION PAYLOAD di cui si è parlato nel paragrafo I.5.5 .

- *Il numero dei pacchetti della negoziazione*: è importante in una negoziazione tra apparati diversi che supportano lo stesso protocollo, che ognuno riceva esattamente i pacchetti che si aspetta, ciò è particolarmente vero se entrano in campo politiche di sicurezza.

- *Il rispetto dei tempi di vita (TOL) delle connessioni:* ogni SA ha una sua durata in termini di traffico o di Kbytes trasmessi, è fondamentale per motivi di sicurezza che la durata configurata sia rispettata.

- *Le politiche effettivamente disponibili ed interoperabili*: le RFC non sono sempre imperative riguardo le politiche che un apparato deve supportare. E' importante controllare quali sono state implementate, e se lo sono state in maniera conforme alle specifiche.

- *Eventuali messaggi di warning segnalati dai log degli apparati*: i servizi di log tengono nota anche di eventuali problemi non bloccanti, nel processo di negoziazione, dei quali va tenuto comunque conto.

- *Il rispetto delle priorità nel caso di proposal multiple tutte accettabili e la negoziazione complessa nel caso in cui di più proposal, una sola sia accettabile*: la costruzione di un pacchetto IKE è molto complicato dalla presenza eventuale di più proposal contemporanee e questo può essere utile per evidenziare problemi di implementazione.

La modalità di autenticazione con al quale sono stati svolti i test è la RSA-signature, ovvero l'autenticazione tramite firma digitale con l'algoritmo RSA per la cifratura asimmetrica di cui si è ampliamente parlato nel corso del Capitolo II.

III.2.3 Strumenti di analisi

Una volta stabiliti gli obiettivi, si stabiliscono gli strumenti necessari per effettuare i test.

Tutti gli apparati analizzati, il SAS a il router della CISCO, sono dotati di strumenti per tenere traccia delle operazioni effettuate, dello stato delle connessioni, del processo di cifratura/decifratura, delle connessioni rigettate. Questi strumenti di log sono proprietari di ciascun sistema, ma comunque sono fondamentali per accertarsi della corretta operatività di tutte le funzioni utilizzate, nonché essenziali per la diagnostica di eventuali problemi e per la loro risoluzione. La loro funzione principale, non è certo quella di aiutare chi si occupa di interoperabilità, queste funzioni nascono infatti come strumenti fondamentali del processo di messa in sicurezza di una rete. Chi si occupa di controllare la sicurezza delle comunicazioni deve poter avere traccia di eventuali attacchi, di eventuali errate configurazioni, nonché poter tenere una statistica del traffico per un corretto dimensionamento, il tutto allo scopo di garantire con continuità il servizio di sicurezza.

Un analizzatore di pacchetti comeEthereal è uno strumento software fondamentale in qualunque laboratorio di sviluppo di apparati di telecomunicazioni. La sua funzione principale consente di registrare tutto il traffico che transita in una determinata rete. E' possibile infatti catturare tutti i pacchetti in transito in un ramo di una rete ethernet sfruttando nel caso specifico, la proprietà di queste reti per la quale tutto il traffico transita per le schede di rete di tutti gli host connessi. Inoltre ogni pacchetto catturato viene analizzato, suddividendo i dati in base ai protocolli della pila TCP/IP. E' così possibile controllare la costruzione dei pacchetti IKE e IPSec, in particolare quando non è richiesta cifratura, altrimenti sono intelligibili solo i campi rimasti in chiaro.

III.2.4 Piattaforme e configurazioni base

I test di interoperabilità sono stati effettuati presso i laboratori dell'AMTEC S.p.A, nelle sedi di Roma e di Abbadia S. Salvatore (SI). Gli apparati o sistemi sottoposti a test sono stati i seguenti:

- Marconi SAS 860/B.
- CISCO Systems router 2600 e 3600

La piattaforma è stata completata da PC che ricoprivano la funzione di host non IPSec, per lo studio dei pacchetti è stato utilizzato un software analizzatore di traffico, l'Ethereal.

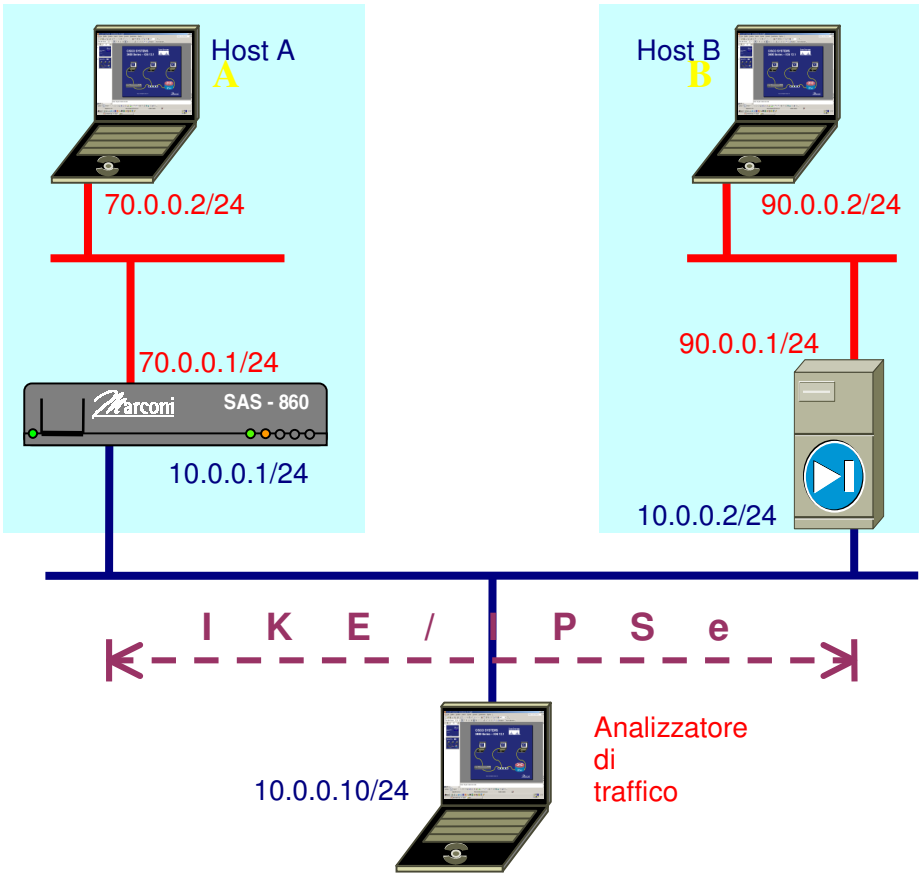

Figure III.2: Piattaforma di Test

In Figura III.2 è rappresentato uno schema generale di piattaforma, dove si individuano tre LAN: la 10.0.0.0/24, la 15.0.0.0/24 e la 20.0.0.0/24.

Le ultime due sono le reti cosiddette "protette" ovvero che usufruiscono per comunicare tra loro di un tunnel IPSec instaurato dai rispettivi security gateway. La rete 70.0.0.0 è protetta dal Marconi SAS860/B, che tramite la rete 10.0.0.0 è connesso al security gateway della rete 90.0.0.0. Quest'ultimo security gateway rappresenta il CISCO 2621, con il quale testare l'interoperabilità.

Il SAS 862 B dell'AMTEC S.p.A. è stato configurato direttamente da console, ovvero grazie a un PC connesso tramite porta seriale alla porta service per mezzo di un applicativo, per esempio l'Hyperterminal della Microsoft, disponendo di una shell testuale per l'intera configurazione dell'apparato. Quella che segue è una tipica configurazione:

ETHERNET1 IP ON 70.0.0.1 255.255.255.0

ETHERNET2 IP ON 10.0.0.1 255.255.255.0

IP DEFGTW 10.0.0.2 ETHERNET2

SYS NAME SAS

IPSEC ON

IPSEC ADD MAP 10000 MANUAL clear

IPSEC ADD MAP 2 ISAKMP

IPSEC TRANSFORM ipsec1 esp/des esp/sha/hmac MODE tunnel LIFETIME SECONDS 200

IPSEC MAP 2 SET PEER 10.0.0.2

IPSEC MAP 2 SET TRANSFORM ipsec1

IPSEC SECUR ADDRESS ETHERNET2

IPSEC ACCESS 10000 PERMIT 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 ike_ipsec

IPSEC ACCESS 10000 PERMIT 70.0.0.0 255.255.255.0 70.0.0.1 255.255.255.255 BIDIRECTIONAL

IPSEC ACCESS 2 PERMIT 70.0.0.0 255.255.255.0 90.0.0.0 255.255.255.0

IKE ON

IKE ADD POLICY ike1

IKE POLICY ike1 ENCRYPTION tdescbc

IKE POLICY ike1 HASH md5

IKE POLICY ike1 AUTHENTICATION rsasign

IKE POLICY ike1 DHGROUP 1024m

IKE POLICY ike1 LIFETIME SECONDS 150

IKE POLICY ike1 PRIORITY 1

IKE ADD PEER cisco

IKE PEER cisco POLICY ike1

IKE PEER cisco MAP 2

ADD SCERT "Sergio Ruspantini" CN:CISCO;O:ELSAG;OU:AMTEC;C:IT; MAP 2

Si possono individuare alcuni semplici comandi:

- Gli indirizzi IP delle interfacce ethernet1, ethernet2 e del gateway di default sono rispettivamente 70.0.0.1, 10.0.0.1 e 10.0.0.2.

- La politica IPSec è una ESP/DES per la cifratura e ESP/SHA/HMAC per l'autenticazione dei dati .

- La mappa 2 è collegata al PEER 10.0.0.2 che risulta essere l'indirizzo del CISCO, all'interfaccia di rete ETHERNET2, alla politica IPSec descritta precedentemente e alla entry 2 della ACCESS LIST che permette il traffico tra la rete 70.0.0.0 \ 24 e la rete 90.0.0.0 \ 24.

- La entry 10000 ha la priorità più bassa ed è necessaria per permettere il passaggio del traffico non protetto tra la rete protetta e il proprio

SECURITY GATEWAY; questa infatti è collegata alla mappa 10000 "MANUAL CLEAR" cioè ad una mappa "in chiaro".

- La politica IKE denominata ike1 usa l' algoritmo TRIPLE DES per la cifratura e l'MD5 per l'autenticazione dei dati (HASH), mentre l'autenticazione dei PEER in comunicazione viene effettuata, come detto in precedenza, tramite firma digitale con l'algoritmo RSA.

- Infine la politica ike1 è collegata al PEER denominato CISCO che è a sua volta legato alla mappa2.

- L'ultimo comando aggiunge una entry nella tabella dei certificati i cui parametri corrispondono a quelli del certificato installato nel CISCO[4]

Il CISCO 2621 è stato configurato nella stessa modalità del SAS quando è stato disponibile all'interno del laboratorio, e tramite TELNET quando non è stato disponibile all' interno del laboratorio ma era comunque collegato alla rete aziendale; in questa circostanza lo schema generale della piatta forma è stato modificato: con riferimento alla figura III.2 la rete 10.0.0.0 è stata sostituita dall' Intrnet aziendale e le rispettive configurazioni sono state modificate. In particolare sono stati cambiati gli indirizzi IP delle interfacce di rete e dei default gateway dei due apparati. Quella che segue è una tipica configurazione del CISCO 2621:

```
hostname cisco1
!
ip domain name amtec.it
!
crypto ca trustpoint ca_amtec
 enrollment terminal
 fqdn cisco1.amtec.it
```

[4] per la gestione dei certificati nei due apparati si rimanda al paragrafo III.4

```
 ip-address 10.36.3.148
 subject-name cn=cisco,o=elsag,ou=amtec,c=it
 crl optional
!
!
crypto isakmp policy 10
 encr 3des
 hash md5
 group 2
 lifetime 200
 authentication rsa-sign
!
!
crypto ipsec transform-set angelo esp-des esp-sha-hmac
!
!
crypto map angelo 10 ipsec-isakmp
 set peer 10.33.0.80
 set security-association lifetime seconds 250
 set transform-set angelo
 match address 110
!
!
crypto key pubkey-chain rsa
 addressed-key 10.36.0.80
  address 10.36.0.80
  key-string
   30819F30      0D06092A   864886F7   0D010101   05000381      8D003081
   89028181      00E6E101   0402067F   C60227AA   2E409577   7643F8D0
   4ED4FCC8   AEBDD75C FFF981A0   B384A5A8   E61D1A34   89EAF4BB
   617C11B8     0076356C    9E5F1B3D   CFC4049C   027223C7   5ADD28A8
   A67EC2FA   03EBE479    3E16D0A4    BB18491D   ACC5BCFA   24115901
```

```
  3E093779    E710D534    E33A99E3  0E295595  DA36B16B  E664A294
  B7E3E42E    81500ED0    91D77FC9  F7020301  0001
 quit
!
interface FastEthernet0/0
 ip address 10.36.3.148 255.255.252.0
 speed auto
 half-duplex
 crypto map angelo
!
interface FastEthernet0/1
 ip address 90.0.0.2 255.255.255.0
 speed auto
 half-duplex
!
ip http server
no ip http secure-server
ip classless
ip default-gateway 10.36.3.254
!
access-list 110 permit ip 90.0.0.0 0.0.0.255  70.0.0.0 0.0.0.255
end
```

Come per il SAS 862, anche per il CISCO 2621 sono individuabili alcuni semplici comandi che riguardano le politiche IKE e IPSec le quali risultano naturalmente identiche a quelle configurate per il SAS, gli indirizzi delle interfacce ETHERNET e del gateway di default, la entry nella ACCESS LIST che permette il traffico tra le due LAN appartenenti alla S-VPN.

Inoltre sono individuabili alcuni comandi che non hanno un corrispettivo nella configurazione del SAS:

- *crypto ca trustpoint ca_amtec …..*

  …….

  …..*crl optional:* riguarda la Certification Autority
- *crypto key pubkey-chain rsa……..*

  ……..

  …..*quit:* riguarda la chiave pubblica del PEER remoto

tali comandi si riferiscono alla gestione dei certificati nel CISCO e verranno illustrati più approfonditamente nel paragrafo III.4.2

III.3 Procedure di Test

Dopo uno studio approfondito della configurazione degli apparati sottoposti a test, per ognuno degli obiettivi precedentemente definiti, si deve ora definire una procedura per individuare i risultati cercati.

Per scatenare la negoziazione dei tunnel è stato utilizzato un pacchetto di echo-request del protocollo ICMP, in sintesi un "ping". Con un'analisi incrociata dei log dei due apparati coinvolti nel test e dell'insieme di pacchetti raccolti dall'analizzatore di traffico, si può valutare l'eventuale raggiungimento del risultato cercato.

Risulta particolarmente utile preparare una serie di tabelle ove raccogliere i risultati ottenuti durante i test, queste tabelle rispettano nei dettagli le procedure da seguire nel processo di test e saranno presentate nei paragrafi seguenti

III.3.1 ID Type

Il primo obiettivo dei Test è volto a verificare che i due apparati possono interoperare con diversi tipi di identificativi. Questi sono contenuti nell' IDENTIFICATIOIN PAYLOAD di cui si è parlato nel paragrafo I.5.5.

Gli ID_TYPE configurabili nel SAS sono i seguenti:

- ID_IPV4_ADDR
- ID_FQDN
- ID_USER_FQDN
- ID_DER_ASN1_DN

L'Identification Payload viene scambiato nel quinto e sesto pacchetto della prima fase IKE e risulta avere la seguente struttura:

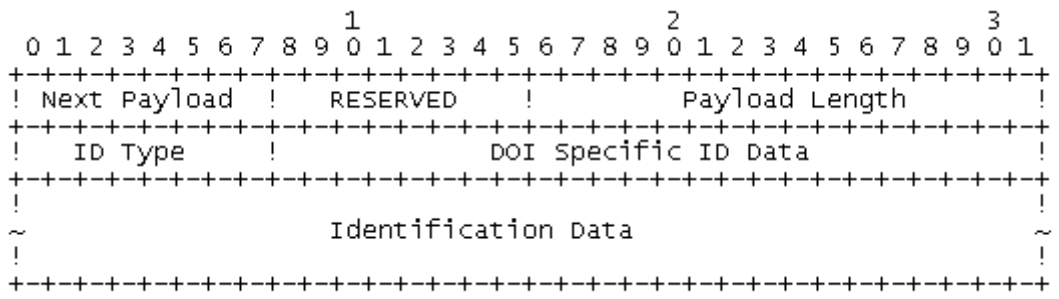

Figure III.3:Formato dell'IDENTIFICATION PAYLOAD

dove il campo "**ID Type**", che indica il tipo di identificativo utilizzato, puó assumere i seguenti valori:

ID Type	Value
RESERVED	0
ID_IPV4_ADDR	1
ID_FQDN	2
ID_USER_FQDN	3
ID_IPV4_ADDR_SUBNET	4
ID_IPV6_ADDR	5
ID_IPV6_ADDR_SUBNET	6
ID_IPV4_ADDR_RANGE	7
ID_IPV6_ADDR_RANGE	8
ID_DER_ASN1_DN	9
ID_DER_ASN1_GN	10
ID_KEY_ID	11

Figure III.4:Valori del campo ID_TYPE

In particolare:

- ID_IPV4_ADDR: indica l'indirizzo IPv4.
- ID_FQDN:specifica una stringa "Fully-Qualified Domain Name". Un esempio di ID_FQDN è "foo.bar .com".
- ID_USER_FQDN: specifica una stringa "Fully Qualified Username". Un esempio di ID_USER_FQDN è "piper@foo.bar.com".

- ID_DER_ASN1_DN: specifica la codifica binaria DER di un Distinguished Name in formato ASN.1 X.500 dell'entità i cui certificati sono stati scambiati per stabilire la SA.

Se l'autenticazione nella fase uno del protocollo IKE viene effettuata usando i certificati digitali (di ciascun formato), ciascun ID usato come input nelle decisioni di policy locale dovrebbe (SHOULD) essere contenuto nel certificato usato nell'autenticazione dello scambio.

Nel caso di autenticazione con i certificati digitali il SAS 862 B utilizza di default l'identificativo DN per l'ID Type e questa scelta è stata effettuata dagli implementatori per semplicità. Gli altri ID sono configurabili per mezzo dei comandi:

- ***ike peer CISCO LOCAL_ID IPV4_ADDR 10.10.5.2***
- ***ike peer CISCO LOCAL_ID FQDN elsag.it***
- ***ike peer CISCO LOCAL_ID USER_FQDN federica@elsag.it***

Con questi comandi il SAS si presenta al peer remoto con l'ID specificato e per configurarlo in modo da fargli accettare un ID diverso da quello di default (che è sempre il DN), si usano gli stessi comandi suddetti sostituendo alla parola LOCAL_ID la parola REMOTE_ID.

Da RFC si ha che ciascun ID usato come input nelle decisioni di policy locale dovrebbe (SHOULD) essere contenuto nel certificato usato nell'autenticazione dello scambio. Il SAS gestisce questo SHOULD nel modo seguente: di default, si controlla che il valore dell'ID configurato corrisponda al corrispondente contenuto nel "Subject Alternative Name" (si veda paragrafo II.1.1) del certificato. Per non effettuare tale controllo si utilizza il comando:

- ***ike peer CISCO flags setflag (NoIDonCert)***

In tal modo è possibile accettare un ID che non è contenuto all'interno del certificato del peer.

La tabella[5] seguente è stata utilizzata per raccogliere i risultati dello studio dell' ID_TYPE senza settare il flag che evita il controllo dell' ID nel certificato

Interoperabilità SAS -CISCO			CISCO							
			Local_ID				*Remote_ID*			
			IPV4ADDR	FQDN	USERFQDN	DN	IPV4_ADDR	FQDN	USERFQDN	DN
SAS	*Local_ID*	IPV4 ADDR								
		FQDN								
		USER FQDN								
		DN								
	Remote_ID	IPV4 ADDR								
		FQDN								
		USER FQDN								
		DN								

Tabella III.1: ID_TYPE

[5] Le tabelle che seguono sono prive di dati poiché mostrate al solo scopo di esplicitare i test effettuati. I risultati ottenuti sono riportati nei paragrafi successivi.

III.3.2 Rispetto dei TOL

Altro obiettivo cruciale dei test effettuati riguarda il rispetto dei tempi di vita delle SA. I test sono stati effettuati configurando un TOL campione di 300 secondi e si è verificato che la durata rispettasse effettivamente il valore configurato. Per far ciò, oltre ad avvalersi di un semplice cronometro, si è potuto controllare il corretto valore tramite la lettura dei file di log di ciascun apparato. Con lo stesso test si controlla che ciascun apparato sia in grado di rinegoziare le SA, quando il TOL è scaduto. Allo stesso tempo ogni apparato può comportarsi in maniera differente quando si dovesse trovare a negoziare una SA con un apparato che ha un TOL metà o doppio rispetto al proprio.

Time Of Life 100%= 300 secondi			CISCO			
			initiator		*Responder*	
			50%	100%	50%	100%
SAS	*initiator*	50%				
		100%				
	responder	50%				
		100%				

Tabella III.2: Rispetto dei TOL

III.3.3 Politiche e rispetto delle priorità

Affinché i due apparati risultino interoperabili essi dovranno supportare e negoziare le politiche da usare per la protezione del traffico. Nella tabella III.3 sono rappresentate tutte le politiche disponibili nel SAS862. L'unico limite alle politiche configurabili per il SAS862 riguarda la possibilità di settare il gruppo Diffie Hellman, che come per la maggior parte degli apparati è limitata ai gruppi I e II; Il CISCO 2621 prevede anche la possibilità di utilizzare il gruppo V.

Questo test prevede la configurazione dell'apparato sostituendo successivamente tutte le politiche elencate in tabella III.3

Politiche supportate e loro interoperabilità				CISCO							
				Diffie-Hellman Group 1				*Diffie-Hellman Group 2*			
				DES		3DES		DES		3DES	
				SHA	*Md5*	*SHA*	*Md5*	*SHA*	*Md5*	*SHA*	*Md5*
SAS	*DH Group 1*	DES	*SHA*								
			Md5								
		3DES	*SHA*								
			Md5								
	DH Group 2	DES	*SHA*								
			Md5								
		3DES	*SHA*								
			Md5								

Tabella III.3:Politiche supportate

I campi in grigio corrispondono all'associazione di politiche tra loro diverse per cui, salvo errori di implementazione, due apparati non possono essere interoperabili con due politiche diverse.

Il payload delle SA può contenere differenti proposal ciascuna con associate una o più trasformate, diremo più genericamente che contiene differenti politiche. L'initiator invia il suo payload SA, il responder seleziona tra le politiche propostegli quella che

corrisponde ad almeno una di quelle presenti nella propria configurazione. Nella tabella che segue la politica 'A' è presente nella configurazione di entrambi gli apparati, insieme alle politiche 'X' e 'Y' che invece sono presenti esclusivamente in uno o nell'altro apparato.

Il responder deve essere in grado di individuare, scorrendo l'insieme delle politiche ricevute da parte dell'initiator, indipendentemente dall'ordine in cui si presentano, quella utilizzabile. Per essere tale una politica deve corrispondere in maniera esatta ad una di quelle presenti nella configurazione del responder, devono coincidere: il protocollo (AH o ESP), il tipo dell'eventuale cifratura (DES o 3DES), il tipo di autenticazione (SHA1 o Md5), il gruppo Diffie Hellman.

Negoziazione Complessa			**CISCO**			
			initiator		*responder*	
			1.politica A 2.politca Y	*1.politica Y 2.politica A*	*1.politica A 2.politica Y*	*1.politica Y 2.politica A*
SAS	*init.*	*1.politica A 2.politica X*				
		1.politica X 2.politica A				
	resp.	*1.politica A 2.politica X*				
		1.politica X 2.politica A				

Tabella III.4:Negoziazione complessa con una sola politica valida

Le configurazioni di due apparati possono anche prevedere le stesse politiche. Nella tabella che segue le politiche sono entrambe valide ma con assegnate priorità diverse. Con questo test si verifica quale sia l'algoritmo di selezione della politica da parte del responder, ovvero se questi privilegia la propria scala di priorità o quella dell'initiator.

Rispetto delle priorità			**CISCO**		
			initiator	*responder*	
			1. politica A *2. politica B*	*1. politica A* *2. politica B*	*1. politica B* *2. politica A*
SAS	*initiator*	*1. politica A* *2. politica B*			
	responder	*1. politica A* *2. politica B*			
		1. politica B *2. politica A*			

Tabella III.5:Negoziazione complessa con politiche entrambe valide

III.4 Certicate Enrollment

In questo paragrafo verranno affrontate le problematiche relative alla gestione dei certificati digitali nei due apparati sottoposti a test, la loro iscrizione e il loro caricamento all' interno dei dispositivi.

Relativamente al SAS 862 e al sistema operativo GAIA4 che lo gestisce, è stato affrontato uno studio sui software proprietari della AMTEC S.p.A. che si occupano della gestione dei certificati e delle CRL per il SAS.

Tale analisi risulta indispensabile per comprendere a pieno le scelte effettuate durante lo svolgimento dei test, le problematiche che di volta in volta si sono presentate e i risultati raggiunti.

III.4.1 La gestione dei certificati nel GAIA4

Il sistema operativo GAIA4 non prevede l'implementazione di protocolli standard per la gestione dei certificati e delle CRL quali il protocollo SCEP "Simple Certificate Enrollment Protocol " e il protocollo CMP "Certificate Management Protocol"; tale gestione è affidata invece ad un suo modulo interno chiamato CRT_MNG che svolge il ruolo di manager dei certificati. Il suddetto modulo interagisce con un software proprietario dell'AMTEC S.p.A. chiamato CSSMAN, che oltre a permettere l'inizializzazione del SAS, si occupa della gestione delle richieste di certificato in formato standard PKCS #10, dei certificati digitali X.509 v3 e delle CRL. Il modo in cui il CSSMAN e il modulo CRT_MNG interagiscono è illustrato nella figura III.6, in cui:

- ATCM: Applicazione che realizza la connessione TCP fornendo servizi di autenticazione e rimpacchettamento degli stream TCP.

- Directory X.500: directory in cui la CA pone i certificati emessi, revocati o sospesi.

- CA: Certification Authority. I test sono stati effettuati usando una CA di laboratorio, ed esattamente la "CryptoCert" AMTEC S.p.A..

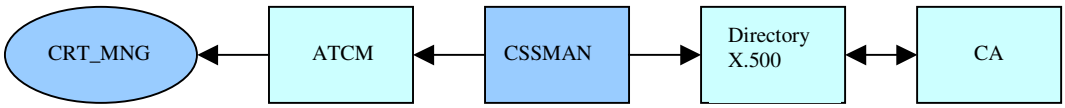

Figure III.5: Interazione tra CRT_MNG e CSSMAN.

Prima di entrare nel dettaglio di come il CRT_MNG e il CSSMAN interagiscono fra loro, va detto che quando il dispositivo non possiede alcun certificato occorre inizializzarlo, ossia bisogna inviare allo stesso il certificato della CA e del CSSMAN. Per far questo, occorre inizializzare a sua volta un dispositivo crittografico, che può essere una smart card, una crypto card o una disk card, e questo può essere fatto con un programma proprietario dell'AMTEC S.p.A., chiamato "CryptoIniDev", che fra le altre cose consente la generazione della richiesta di certificato in formato PKCS #10 per il CSSMAN. Tale richiesta viene firmata dalla CA che rilascia il certificato corrispondente, il quale a sua volta viene memorizzato nel dispositivo crittografico insieme al certificato della CA. Questi certificati possono essere inviati al SAS per mezzo del programma CSSMAN e in tal modo, il possessore del dispositivo crittografico diventa l'amministratore del SAS. Infatti, l'amministratore possedendo il dispositivo crittografico è l'unico che può richiedere la generazione di certificati e l'invio di questi ultimi al SAS. Lo schema a cui si fa riferimento è illustrato nella figura III.7.

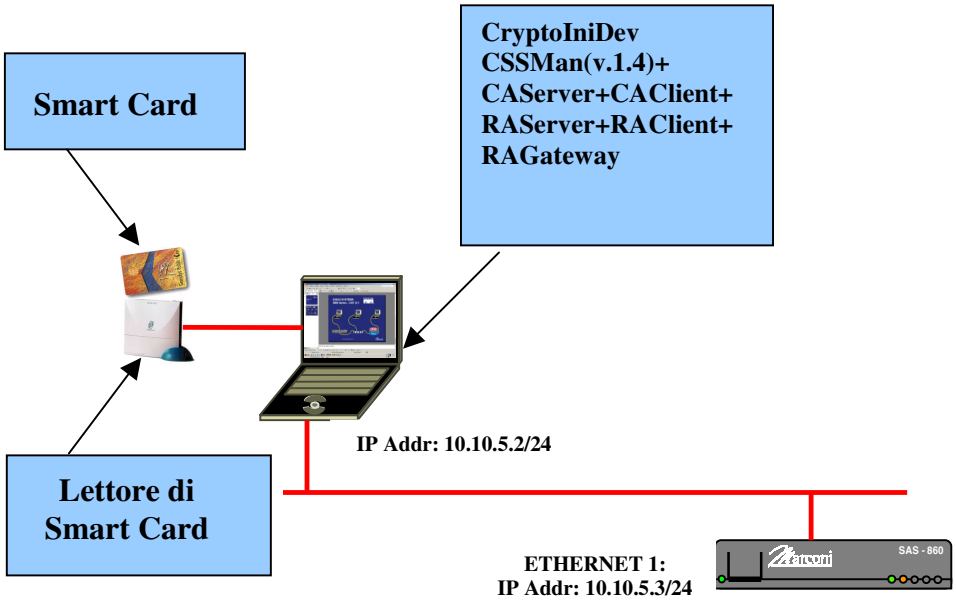

Figure III.6: Schema della gestione dei certificati nel SAS

Una volta inviati i certificati della CA e del CSSMAN al SAS con il CSSMAN, è possibile richiedere la generazione di una richiesta di certificato e inviare il certificato generato al SAS. Per far questo, seguendo lo schema di **Errore. L'origine riferimento non è stata trovata.**, CSSMAN genera la richiesta di certificato in formato PKCS #10 inserendo le informazioni che si vogliono avere nel certificato finale. A tal punto, il CSSMAN invia la richiesta al SAS, più esattamente al modulo CRT_MNG, che richiede al modulo RSA (modulo interno del S.O. GAIA4), di generargli una coppia di chiavi. Il CRT_MNG, avute le chiavi termina l'operazione riempendo i campi del PKCS #10 che erano stati lasciati vuoti dal CSSMAN. Questi campi sono la *chiave pubblica* e la *firma digitale*, campi che necessariamente vanno gestiti all'interno del S.O. GAIA4 per motivi di sicurezza.

Il CSSMAN, preleva il PKCS #10 dal CRT_MNG e lo invia alla CA, la quale rilascia il certificato che viene inviato al SAS dal CSSMAN.

III.4.2 La gestione dei certificati nel CISCO

La gestione dei certificati nei router CISCO equipaggiati di IOS 12.3[6] prevede l'utilizzo del protocollo SCEP per l'iscrizione (enrollment) e il caricamento dei certificati nel dispositivo. Lo stesso protocollo viene utilizzato per scaricare le CRL dal CDP e verificare in tal modo la validità dei certificati digitali durante l'autenticazione del peer remoto che avviene durante la prima fase di una negoziazione IKE.

Oltre l'utilizzo del protocollo SCEP è previsto, però, anche un enrollment manuale dei certificati attraverso una procedura di "cut and paste"; tale metodologia è risultata essere indispensabile per lo svolgimento dei test ed è stata utilizzata per i motivi descritti di seguito.

La CA utilizzata per generare e firmare i certificati digitali è una CA di laboratorio dell'AMTEC S.p.A.; durante la trasmissione dei certificati dalla CA Server al dispositivo tramite protocollo SCEP si sono verificati dei problemi che hanno impedito tale operazione. In pratica il CISCO non accetta il formato di trasmissione del certificato, rifiutandolo. I motivi per cui ciò avviene vanno ricercati analizzando l'implementazione del protocollo SCEP nella CA e nella IOS 12.3. Sotto suggerimento del reparto sviluppo di Abbadia S. Salvatore (SI) si è deciso perciò di utilizzare l'enrollment manuale dei certificati non avendo a disposizione nessuno strumento per risolvere tale problematica in breve tempo.

Di seguito verranno analizzate le righe di configurazione relative alla gestione dei certificati nel CISCO che sono state presentate nel paragrafo III.2.4 :

```
crypto ca trustpoint ca_amtec
  enrollment terminal
  fqdn cisco1.amtec.it
  ip-address 10.36.3.148
  subject-name cn=cisco,o=elsag,ou=amtec,c=it
  crl optional
```

[6] Interworking Operatine System versione 12.3

La prima riga definisce un entità di certificazione a cui viene dato il nome fittizio di ca_amtec; le righe a seguire definiscono i vari parametri attribuiti a questa entità e in particolare:

- *enrollment terminal* stabilisce che l'enrollment dei certificati avverrà manualmente tramite una procedura di "cut and paste". Nel caso si fosse voluto utilizzare il protocollo SCEP si sarebbe dovuto usare il comando *enrollment url http://<CA ip-address>:<tcp-port>*

- *fqdn cisco1.amtec.it* e *ip-address 10.36.3.148* definiscono i parametri FQDN e IPV4_ADDR che verranno inseriti nel campo "Subject Alternative Name"

- *subject-name cn=cisco,o=elsag,ou=amtec,c=it* definisce I valori del campo "Subject"

- *crl optional* stabilisce la gestione delle CRL che per semplicità è stata scelta opzionale

Un secondo insieme di comandi verrà presentato di seguito:

crypto key pubkey-chain rsa
addressed-key 10.36.0.80
address 10.36.0.80
key-string
30819F30……
…… F7020301 0001
quit

Questo secondo insieme di righe di configurazione gestisce la chiave pubblica del peer remoto. Tali comandi devono essere inseriti quando la gestione dei certificati viene effettuata manualmente e, quindi, quando non è possibile utilizzare il protocollo

SCEP per verificare la chiave pubblica del peer direttamente dalla CA. In particolare i precedenti comandi definiscono:

- *crypto key pubkey-chain rsa* definisce una catena di chiavi pubbliche per l'algoritmo RSA.

- *addressed-key 10.36.0.80* definisce una chiave pubblica legata ad un indirizzo usato come identificativo; è possibile anche definire una *named-key* legata ad un nome invece che ad un indirizzo.

- *address 10.36.0.80* definisce l'indirizzo ip del peer a cui appartiene la chiave pubblica.

- *key string 30819F3…… F7020301 0001…… quit* inserisce la chiave pubblica presente nell' omonimo campo del certificato digitale.

Questa modalità di gestione risulta evidentemente meno scalabile; l'inserimento della chiave pubblica di ogni certificato che si vuole accettare durante l'autenticazione di un peer risulterebbe laborioso in uno schema generale di S-VPN in cui il numero dei peer, e quindi dei certificati, potrebbe risultare non trascurabile. Tuttavia questa operazione è risultata indispensabile durante lo svolgimento dei test come si può evincere dal debug log del dispositivo riportato qui di seguito:

Jul 7 16:35:03.627: ISAKMP (0:1): received packet from 10.36.3.133 dport 500 sport 500 Global (R) MM_KEY_EXCH

Jul 7 16:35:03.635: ISAKMP (0:1): Input = IKE_MESG_FROM_PEER, IKE_MM_EXCH

Jul 7 16:35:03.635: ISAKMP (0:1): Old State = IKE_R_MM4 New State = IKE_R_MM5

Jul 7 16:35:03.639: ISAKMP (0:1): processing ID payload. message ID = 0

Jul 7 16:35:03.639: ISAKMP (0:1): peer matches *none* of the profiles

Jul 7 16:35:03.639: ISAKMP (0:1): processing CERT payload. message ID = 0

Jul 7 16:35:03.639: ISAKMP (0:1): processing a CT_X509_SIGNATURE cert

Jul 7 16:35:03.667: ISAKMP (0:1): peer's pubkey isn't cached

Jul 7 16:35:03.703: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 10.36.3.133 is bad: CA request failed!

Jul 7 16:35:03.703: ISAKMP (0:1): Input = IKE_MESG_INTERNAL, IKE_PROCESS_MAIN_MODE

Jul 7 16:35:03.703: ISAKMP (0:1): Old State = IKE_R_MM5 New State = IKE_R_MM5

Jul 7 16:35:03.907: ISAKMP (0:1): sending packet to 10.36.3.133 my_port 500 peer_port 500 (R) MM_KEY_EXCH

Jul 7 16:35:03.907: ISAKMP (0:1): Input = IKE_MESG_INTERNAL, IKE_PROCESS_ERROR

Jul 7 16:35:03.907: ISAKMP (0:1): Old State = IKE_R_MM5 New State = IKE_R_MM4

Analizzando il debug log del dispositivo si può notare come il CISCO durante l'analisi del CERT payload rifiuti al negoziazione dopo il controllo sulla chiave pubblica del peer che risulta non memorizzata (peer's pubkey isn't cached).

La procedura di enrollment e caricamento dei certificati deve essere completata, dopo aver definito l'entità di certificazione e i parametri ad essa relativi, attraverso i seguenti comandi:

- *crypto ca authenticate ca_amtec* installa il certificto della CA nel dispositivo.

- *crypto key generate rsa* genera una coppia di chiavi pubblica e privata.

- *crypto ca enroll ca_amtec* genera una richiesta di certificato in formato pkcs10 associato alla chiave pubblica appena generata.

- *crypto ca import ca_amtec certificate* installa il certificato nel dispositivo.

III.5 Risultati

I risultati sono stati suddivisi in due sezioni riguardanti il MAIN e l' AGGRESSIVE mode. La negoziazione delle SA, per quanto riguarda la prima fase del protocollo IKE, avviene infatti con caratteristihe e numero di pacchetti differenti nelle due modalità come descritto nel I.4.3. I test precedentemente descritti sono stati infatti ripetuti per entrambi il MAIN e l'AGGRESSIVE mode.

Prima di affrontare lo studio dei risultati specifici delle due modalità di negoziazione sarà affrontato, però, un problema verificatosi nella prima fase di test. Tale problema ha riguardato la seconda fase del protocollo IKE quando l'iniziator della negoziazione risultava essere il CISCO. La negoziazione della SA falliva sistematicamente durante il primo scambio del quick mode in quanto il SAS rifiutava gli attributi della SA IPSec inviati dal CISCO all'interno del SA payload. Nello specifico veniva inviato un NOTIFICATION payload del tipo "attributi non supportati". Gli attributi in questione riguardano il tempo di vita della SA IPSec; questo attributo può assumere valori in secondi e in Kbyte e può essere configurato in modo da assumere valori a piacere all' interno di un determinato range. Il problema sussisteva in quanto il SAS non riusciva a gestire istanze multiple di questo stesso attributo: come si può evincere dal debug log del SAS riportato di seguito, infatti, negli attributi inviati dal CISCO compare sia un TOL espresso in secondi che un TOL espresso in Kbyte; quest' ultimo, infatti, viene settato ad un valore di default quando non specificato nella configurazione dell'apparato.

IKE UDP 0: rcv pkt for 10.36.3.148:500 -> 10.33.0.80:500

IKE 1: start QUICK MODE EXCHANGE from peer 10.10.10.2:500 -> 10.10.10.1:500 responder

SA rcv

PROPOSAL rcv

IKE 1: read HASH payload

DATA rcv

IKE 1: HASH(1) verification OK

SA rcv

PROPOSAL rcv

TRANSFORM rcv
ATTRIBUTE rcv
80 04 00 01
IKE: read attribute ENC mode 1
ATTRIBUTE rcv
80 01 00 01
ATTRIBUTE rcv
80 02 00 96
IKE: read attribute LIFETIME seconds 150
ATTRIBUTE rcv
80 01 00 02
ATTRIBUTE rcv
80 02 27 10
IKE: read attribute LIFETIME kByte 10000
IKE: found error in analisys of msg:
Origin... 10.36.3.148
Type..... Attributes not supported
State.... 4
Payload.. transform
value.... 1
Descr: On the same proposal, I found differents timelife kByte value!!!
IKE 1: informational sent to IKE
IKE 1: error in verifing 1th msg
IKE 2: write NOTIFIC payload

Il log presentato sopra descrive chiaramente quanto detto finora. Il problema era dovuto ad un errore di implementazione della versione 4.2.2 del GAIA4 che era attualmente caricato sul SAS ed è stato risolto nella nuova versione del software la 4.2.3. I test sono stati quindi effettuati caricando sul dispositivo la nuova versione del software ed hanno dato i risultati esposti nel seguito del paragrafo.

III.5.1 Fase I MAIN MODE

Prima di analizzare i risultati relativi ai test sull' ID_TYPE bisogna fare alcune considerazioni.

Come detto nel III.3.1, il SAS 862 utilizza di default l'identificativo DN quando l'autenticazione viene effettuata con i certificati digitali sia per quanto riguarda l' ID locale, cioè l'identificativo con il quale si presenta al peer remoto, sia per quanto riguarda l' ID remoto, cioè quello con cui ci si aspetta che il peer remoto si presenti. Anche il CISCO utilizza di default l'identificativo DN, ma solo relativamente all' ID locale; nel CISCO non è possibile infatti settare un ID remoto, o meglio quest'ultimo accetta indifferentemente qualsiasi ID_TYPE dal peer remoto. Inoltre mentre il SAS effettua un controllo dell'identificativo all'interno del certificato, il CISCO non effettua tale controllo accettando indipendentemente ogni valore presente all' interno del campo IDENTIFICATION DATA.

In tabella III.6 sono inseriti i risultati dei test; la tabella è stata modificata per i motivi appena esposti. I campi in grigio corrispondono a punti in cui l'ID_TYPE con cui il CISCO si presenta è diverso da quello settato nel SAS per cui la negoziazione fallisce come ci si aspettava. In tutti gli altri casi i test hanno dato esito positivo.

I test sono stati effettuati senza settare il flag che evita il controllo dell' identificativo nel certificato digitale. Il settaggio di tale flag non è comunque risultato necessario poiché si è verificato che da parte del CISCO risulta impossibile settare un valore di identificativo locale che non sia contenuto all' interno del certificato. Questo infatti effettua un controllo sul proprio ID prima di scrivere quest'ultimo nell' IDENTIFICATION PAYLOAD e assume l'identificativo di default nel caso questo controllo fallisca come si può evincere dal debug log riportato di seguito:

Jul 26 12:55:12.829: ISAKMP (1): My ID configured as IPv4 Addr,but Addr not in Cert!

Jul 26 12:55:12.829: ISAKMP (1): Using FQDN as My ID

Jul 26 12:55:12.829: ISAKMP (0:1): SA is doing RSA signature authentication using id type ID_FQDN

Jul 26 12:55:12.829: ISAKMP (1): ID payload

next-payload : 6

type : 2

FQDN name : cisco1.amtec.it

protocol : 17

port : 500

length : 19

Jul 26 12:55:12.829: ISAKMP (1): Total payload length: 23

Jul 26 12:55:12.833: ISAKMP (0:1): using the ca_amtec trustpoint's keypair to sign

Interoperabilità SAS -CISCO			CISCO				
			Local_ID				*Remote_ID*
			IP V4 ADDR	FQDN	USER FQDN	DN	PARAMETRO NON CONFIGURABILE
SAS	*Local_ID*	IPV4 ADDR					✓
		FQDN					
		USER FQDN					
		DN					
	Remote_ID	IPV4 ADDR	✓				
		FQDN		✓			
		USER FQDN			✓		
		DN				✓	

Tabella III.6: Risultati ID_TYPE

Si potrebbe concludere quindi dicendo che i due dispositivi gestiscono in maniera differente lo SHOULD espletato nel III.3.1, uno effettuando un controllo

sull'identificativo del peer remoto, l'altro effettuando un controllo, senza però possibilità di evitarlo, sul proprio identificativo.

I test sul rispetto del TOL hanno dato tutti esito positivo come illustrato in tabella III.7.

La negoziazione è stata completata in ogni caso; inoltre si è verificato che il tempo di vita delle SA fosse effettivamente pari a quello configurato negli apparati e che questi ultimi fossero in grado di rinegoziare la SA una volta che il TOL era scaduto.

Time Of Life 100%= 300 secondi			CISCO			
			initiator		*Responder*	
			50%	100%	50%	100%
SAS	*initiator*	50%			✓	✓
		100%			✓	✓
	responder	50%	✓	✓		
		100%	✓	✓		

Tabella III.7: Risultati TOL

Analizziamo ora i risultati dei test riguardanti le politiche di protezione del traffico. Per quanto riguarda il supporto delle diverse politiche configurabili tutti itest, effettuati modificando di volta in volta le configurazioni, hanno dato esito positivo.

Dalla tabella III.8 si può notare come la negoziazione sia andata a buon fine soltanto nei casi in cui le politiche configurate nei due apparati risultano uguali. Nei restanti casi (campi in grigio) la negoziazione non viene completata così come ci si aspettava.

Politiche supportate e loro interoperabilità				CISCO							
				Diffie-Hellman Group 1				*Diffie-Hellman Group 2*			
				DES		3DES		DES		3DES	
				SHA	*Md5*	*SHA*	*Md5*	*SHA*	*Md5*	*SHA*	*Md5*
SAS	*DH Group 1*	DES	*SHA*	✓							
			Md5		✓						
		3DES	*SHA*			✓					
			Md5				✓				
	DH Group 2	DES	*SHA*					✓			
			Md5						✓		
		3DES	*SHA*							✓	
			Md5								✓

Tabella III.8: Risultati politiche supportate

Anche nel caso di negoziazione con politiche multiple, di cui una sola accettabile, i test hanno dato tutti esiti positivi. Si è verificato che i due apparati nel ruolo di responder selezionassero l'unica politica effettivamente accettabile. Si ricorda che i due apparati sono stati configurati, per l'esecuzione di questo test, entrambi con due transform di cui solamente uno in comune. I risultati sono elencati in tabella III.9.

Di maggiore interesse sono invece i test sulla negoziazione complessa con politiche entrambe valide. In questo caso, infatti, l'aspetto da tenere più in considerazione è il rispetto delle priorità: il SAS e il CISCO sono stati infatti configurati entrambi con le stesse politiche, ma sono state assegnate loro un ordine di priorità differente.

I risultati mostrati in tabella III.10 mostrano alcune differenze implementative:

Il SAS nel ruolo di responder seleziona tra le politiche accettabili quella che rispetta la propria scala di priorità al contrario di quello che avviene per il CISCO il quale sceglie la politica accettabile rispettando la scala di priorità dell'iniziator.

Negoziazione Complessa	CISCO

			initiator		*responder*	
			1.politica A 2.politca Y	*1.politica Y 2.politica A*	*1.politica A 2.politica Y*	*1.politica Y 2.politica A*
SAS	*init.*	*1.politica A 2.politica X*			**A**	**A**
		1.politica X 2.politica A			**A**	**A**
	resp.	*1.politica A 2.politica X*	**A**	**A**		
		1.politica X 2.politica A	**A**	**A**		

Tabella III.9: Risultati negoziazione complessa con una sola politica accettabile

Rispetto delle priorità			**CISCO**		
			initiator	*responder*	
			1. politica A 2. politica B	*1. politica A 2. politica B*	*1. politica B 2. politica A*
SAS	*initiator*	*1. politica A 2. politica B*		**A**	**A**
	responder	*1. politica A 2. politica B*	**A**		
		1. politica B 2. politica A	**B**		

Tabella III.10: Risultati negoziazione complessa con entrambe le politiche valide (rispetto delle priorità)

Quanto detto può essere dedotto anche analizzando I debug dei due apparati. In particolare si riporta di seguito il debug log del router CISCO in cui le righe sottolineate evidenziano come questo confronti la transform con priorità 1 inviata dal SAS con tutte le proprie politiche prima di passare all'analisi delle transform successive, rispettando in questo modo la scala di priorità dell' iniziator.

Jul 7 16:35:00.175: ISAKMP (0:0): received packet from 10.36.3.133 dport 500 sport 500 Global (N) NEW SA

Jul 7 16:35:00.187: ISAKMP (0:1): processing SA payload. message ID = 0

Jul 7 16:35:00.191: ISAKMP (0:1): Checking ISAKMP transform 1 against priority1 policy

Jul 7 16:35:00.191: ISAKMP: encryption 3DES-CBC

Jul 7 16:35:00.191: ISAKMP: hash MD5

Jul 7 16:35:00.191: ISAKMP: auth RSA sig

Jul 7 16:35:00.191: ISAKMP: default group 2

Jul 7 16:35:00.191: ISAKMP: life type in seconds

Jul 7 16:35:00.191: ISAKMP: life duration (VPI) of 0x0 0x0 0x0 0x96

Jul 7 16:35:00.195: ISAKMP (0:1): Diffie-Hellman group offered does not match policy!

Jul 7 16:35:00.195: ISAKMP (0:1): atts are not acceptable. Next payload is 0

Jul 7 16:35:00.195: ISAKMP (0:1): Checking ISAKMP transform 1 against priority2 policy

Jul 7 16:35:00.195: ISAKMP: encryption 3DES-CBC

Jul 7 16:35:00.195: ISAKMP: hash MD5

Jul 7 16:35:00.195: ISAKMP: auth RSA sig

Jul 7 16:35:00.195: ISAKMP: default group 2

Jul 7 16:35:00.195: ISAKMP: life type in seconds

Jul 7 16:35:00.195: ISAKMP: life duration (VPI) of 0x0 0x0 0x0 0x96

Jul 7 16:35:00.195: ISAKMP (0:1): atts are acceptable. Next payload is 0

III.5.2 Fase I AGGRESSIVE MODE

La modalità di negoziazione AGGRESSIVE per la fase uno di ike è caratterizzata, come detto, dallo scambio di quattro pacchetti rispetto ai sei di cui necessita la modalità MAIN. Questo è fatto per rendere l'architettura IPSec meno pesante dal punto di vista elaborativo, ma rende più critiche le operazioni di negoziazione delle SA. Da quanto detto si comprende bene che l'AGGRESSIVE MODE necessita anch'esso di una batteria di test, come quelli effettuati per il MAIN MODE, che ne verifichi l'interoperabilità con altre implementazioni. Avendo ben chiaro questo concetto si sono ripetuti tutti i test effettuati fino ad ora cambiando la modalità di negoziazione sul SAS per mezzo del comando:

- ***ike peer CISCO mode AGGRESSIVE***

Per mezzo di questo comando si forza il SAS ad inziare una negoziazione in modalità AGGRESSIVE.

Nel router CISCO 2621, invece, non è stato possibile inserire un comando analogo e cioè non è stato possibile configurarlo in modo che iniziasse una negoziazione in AGGRESSIVE MODE. Tuttavia il CISCO nel ruolo di responder riconosce ed accetta una negoziazione in AGGRESSIVE MODE e ciò ci ha permesso di effettuare i test anche se solamente con il SAS nel ruolo di iniziator della comunicazione.

I test effettuati hanno evidenziato un grosso problema di interoperabilià che causa il fallimento sistematico della negoziazione in AGGRESSIVE MODE.

Prima di descrivere in dettaglio il problema riscontrato è importante richiamare alcune nozioni specifiche del protocollo IKE.

Da RFC 2409 si ha che il protocollo IKE utilizza per il trasporto il protocollo UDP su porta 500. Questa specifica del protocollo rende complicato il passaggio di traffico IPSec in presenza di NAT (Network Address Translation), in presenza, cioè, di un dispositivo che transla gli indirizzi privati della rete in indirizzi pubblici, mantenendone, per esempio, una discriminazione su base porta. In questo scenario

risulta chiaro che la negoziazione ike potrebbe non avvenire su porta 500 dovendo il NAT dedicare due porte diverse a due diversi host che usano lo stesso indirizzo pubblico. Altri problemi sussistono nell' utilizzo contemporaneo di IPSec e NAT che riguardano l'autenticazione e la cifratura del pacchetto. In questa situazione viene utilizzato il NAT-T[7] ovvero il NAT traversal. Non entreremo nel dettaglio di come questo funzioni, tuttavia ciò che è importante dire è che in presenza di NAT traversal deve essere scambiato un NAT-T VENDOR payload durante il primo scambio di pacchetti ed un NAT-D payload cioè un pacchetto di cui l'altro peer necessita per tenere traccia della comunicazione (NAT-Discovery).

Si ricorda inoltre che all'interno dell' IDENTIFICATION PAYLOAD i campi PROTOCOL e PORT indicano per l'appunto i valori assegnati al protocollo di trasporto e alla porta. Da RFC 2407 si ha che:

During Phase I negotiations, the ID port and protocol fields MUST be set to zero or to UDP port 500. If an implementation receives any other values, this MUST be treated as an error and the security association setup MUST be aborted.

Anche se in nessuno dei due Security Gataway utilizzati nei test è stata configurata la funzionalità di NAT, i test hanno reso necessaria l'introduzione di questi concetti.

Dall' analisi incrociata dei debug dei due apparati, presentati di seguito, si è potuto tenere traccia della negoziazione e capire quando e perché questa falliva.

Jul 19 09:42:22.961: ISAKMP (0:1): SA is doing RSA signature authentication using id type ID_DER_ASN1_DN

Jul 19 09:42:22.961: ISAKMP (1): ID payload

next-payload : 10

type : 9

addr : 10.36.3.148

protocol : 17

port : 0

[7] RFC 3947e 3948

length : 122

Jul 19 09:42:22.965: ISAKMP (1): Total payload length: 126

Jul 19 09:42:24.077: ISAKMP (0:1): sending packet to 10.36.3.133 my_port 500 peer_port 500 (R) AG_INIT_EXCH

Durante la negoziazione della fase I i campi identificativi di protocollo e porta devono (MUST da RFC) essere settati a zero o con valori UDP porta 500; si nota invece che il router CISCO setta tali campi con valori pari a UDP porta 0.

Questo prevedrebbe la presenza di un NAT-T che necessita per quanto detto dello scambio di uno specifico VENDOR ID payload. Questo è ciò che si aspetta anche il SAS che rispettando le specifiche del protocollo rigetta la connessione quando verifica che i suddetti campi assumono valori diversi da quelli previsti.

IKE MNG cisco: rcv pkt on AGGRESSIVE mode

IKE MNG cisco: analyse II pkt

IKE MNG cisco: RCV Remote ID:IPv4 address: 10.36.3.148 - Prot UDP port 0 -

IKE MNG cisco: rcv ID for NAT-T scenario (prot 17 port 0)

IKE: found error in analisys of msg:

Origin... 10.36.3.148

Type..... Invalid transform-id

State.... 17

Payload.. identifier

value.... 0

Descr: With port NULL I need for NAT-T vendor payload

IKE MNG cisco: informational sent to peer 10.36.3.148

IKE MNG cisco: pkt not valid from peer 10.36.3.148 st 17 Release...

ERROR CONNECTION FAILED

Il debug log dell'apparato descrive chiaramente i motivi del fallimento della negoziazione.

Le cause del problema riscontrato e descritto sopra sembrerebbero essere attribuite ad un errore di implementazione presente nel software del router CISCO che ricordiamo essere la IOS 12.3 (1a)

III.5.3 Commenti

CONCLUSIONI

Questa tesi raccoglie i risultati di un lavoro di ricerca, di studio e di test, completati nel corso di uno stage della durata di tre mesi, presso l'ELSAG S.p.A..

Per la realizzazione dei test è stata utilizzata la struttura dei laboratori *Security System Integration* e *Sviluppo* dell'AMTEC di Roma e Abbadia San Salvatore (SI), società appartenente alla divisione Sistemi e Servizi di Sicurezza di ELSAG S.p.A..

Grazie ad un percorso formativo di studio e di applicazione degli argomenti approfonditi sono stati raggiunti i seguenti risultati:

- conoscenza approfondita della suite di protocolli di sicurezza IKE/IPSec;
- acquisizione di competenze relative a S-VPN e security gateway;
- apprendimento delle metodologie di testing

Oggetto di studio è stata la suite di protocolli IKE/IPSec, strumento fondamentale per la sicurezza delle VPN, in una sua particolare implementazione: il router SAS 862 con il sistema operativo Gaia4.

L'obiettivo principale è stato dimostrare l'interoperabilità a livello IKE / IPSec, del router SAS 862 con il router CISCO 2600 series già ampliamente presente sul mercato.

Le esigenze che hanno spinto ELSAG S.p.A. a sottoporre a tali test il proprio prodotto sono molteplici: un prodotto che sia competitivo nel mercato moderno deve essere in grado di fornire sia la garanzia di massima sicurezza possibile e sia di interoperabilitá con i prodotti piú diffusi nel panorama internazionale

I risultati ottenuti hanno permesso a ELSAG di dimostrare l'interoperabilità dei propri prodotti con CISCO pur evidenziando differenze implementative tra i due apparati.

In particolare i test effettuati hanno mostrato una diversa interpretazione della specifica riguardante la presenza dell'identificativo all'interno del certificato ed una

seconda differenza riguardante il rispetto delle priorità nella scelta della politica in una negoziazione complessa.

I casi di mancata interoperabilità sono stati due:

Il primo riguardante il non supporto degli attributi della SA da parte del SAS era dovuto ad un errore implementativo ed è stato risolto nella nuova versione del software (GAIA4 4.2.3).

Il secondo riguardante il fallimento sistematico della negoziazione in AGGRESSIVE MODE …….

Infine i test hanno mostrato l'esigenza di implementare nel GAIA4 i protocolli di gestione dei certificati digitali quali SCEP e LDAPv2, implementazione che dovrebbe avvenire nella prossima versione del software, al fine di rendere più semplice e automatica la gestione delle PKI

APPENDICE A: CRITTOGRAFIA

La crittografia è lo studio della codifica e della decodifica dei dati. Il termine crittografia deriva dalle parole kryptos che significa nascosto, e graphia, che significa scrittura. La crittografia riveste un ruolo preminente nell'ambito dei meccanismi di sicurezza, essendo in grado di fornire la maggior parte dei servizi contemplati nell'architettura di sicurezza stabilita dall'ISO (International Standard for Standardization); pertanto i sistemi crittografici rivestono un'importanza fondamentale nella soluzione di molte problematiche di sicurezza connesse con la protezione di informazioni.

Per inquadrare il problema è opportuno rifarsi al modello di riferimento proposto dall'ISO che individua cinque classi di funzionalità necessarie per garantire il desiderato livello di sicurezza. Queste sono:

- **Confidenzialità**: con tale funzionalità si vuole impedire di rilevare informazioni riservate ad entità non autorizzate alla conoscenza di tali informazioni. La confidenzialità è ottenuta tramite tecniche crittografiche di cifratura dei dati.

- **Integrità dei dati**: i servizi di integrità dei dati proteggono contro gli attacchi attivi finalizzati ad alterare illegittimamente il valore di un dato. L'alterazione di un messaggio può comprendere la cancellazione, la modifica, o il cambiamento dell'ordine dei dati.

- **Autenticazione**: i servizi di autenticazione permettono di accertare l'identità delle entità.

- **Controllo degli accessi**: una volta che l'autenticazione è avvenuta, è possibile eseguire un controllo degli accessi in modo tale da verificare che vengano utilizzate solo quelle risorse o servizi ai quali si è autorizzati.

- **Non ripudio**: tali servizi hanno la finalità di impedire che un'entità possa rinnegare la partecipazione ad una transazione o comunicazione a cui in realtà ha preso parte. Si noti che tale servizio non previene il ripudio di una comunicazione o di una transazione ma fornisce gli elementi per dimostrare in caso di contenzioso l'evidenza dei fatti.

Le principali tecniche crittografiche, cifratura e firma digitale, costituiscono i componenti basilari nell'implementazione di tutti i servizi di sicurezza sopra descritti. Alla base delle tecniche crittografiche si colloca l'algoritmo di cifratura che fornisce le funzioni di trasformazione di un testo da testo in chiaro a testo cifrato.

Gli algoritmi di cifratura si suddividono in due classi: algoritmi a chiave simmetrica e algoritmi a chiave asimmetrica. Fino a poco più di venti anni fa erano conosciuti solo gli algoritmi a chiave simmetrica, poi nel 1976 Diffie ed Hellman presentarono un protocollo per lo scambio di una chiave segreta attraverso un canale insicuro. Tale meccanismo era stato inteso essenzialmente per risolvere il problema dell'avvio di un normale sistema di cifratura a chiavi simmetriche ma in realtà ha posto le basi della crittografia a chiave pubblica.

La trasformazione inversa, da testo cifrato a testo in chiaro, si dice decifratura. Per decifrare il messaggio non basta conoscere l'algoritmo di cifratura utilizzato, ma è necessario conoscere anche la chiave. In genere nello studio degli algoritmi crittografici e della loro sicurezza si ipotizza che l'algoritmo sia noto a tutti (pubblico) e che solo la chiave sia segreta: questo perché oggi si considera inaffidabile un sistema crittografico la cui sicurezza si basi solo sulla segretezza dell'algoritmo.

A.1 Crittografia simmetrica e asimmetrica

La crittografia tradizionale effettua cifratura e decifratura con la stessa chiave. Tra gli algoritmi simmetrici più diffusi ci sono il DES (Data Encryption Standard) e il Triple DES.

Il problema di questo approccio è la distribuzione delle chiavi, infatti se due interlocutori vogliono usare un algoritmo di questo tipo per comunicare in modo sicuro devono prima accordarsi in qualche modo sulla chiave. Poiché il canale dati non è sicuro non può essere impiegato per trasmettere la chiave, è necessario quindi un canale alternativo "out-of-band".

Il problema è stato risolto in tempi relativamente recenti (anni Settanta) con l'introduzione della crittografia a chiave pubblica. Con algoritmi di questo tipo ogni utente ha due chiavi: una pubblica da distribuire a tutti quelli con cui vuole comunicare, e una privata da tenere segreta. La conoscenza della chiave pubblica non permette di determinare quella privata, per questo la chiave pubblica può essere resa pubblica senza compromettere la sicurezza del sistema. L'insieme di dati che vengono cifrati con la chiave pubblica (operazione che può essere fatta da chiunque) può essere decifrato solo con la chiave privata corrispondente (operazione che può essere fatta solo dal proprietario della chiave): in questo modo non c'è più il problema di comunicare segretamente la chiave perché questa è nota a tutti. Per comunicare in modo sicuro con una persona basta cifrare il messaggio con la sua chiave pubblica. Gli algoritmi di questo tipo sono detti a chiave asimmetrica.

A.2 Firma digitale

La firma digitale (digital signature) è una delle importanti innovazioni permesse dalla crittografia asimmetrica ed è la tecnologia con cui possono essere effettivamente soddisfatti tutti i requisiti richiesti per dare validità legale ad un documento elettronico. La firma digitale garantisce i servizi di integrità, autenticazione e non ripudio.

1. **Autenticità della firma**: la firma deve assicurare il destinatario del documento che il mittente ha deliberatamente sottoscritto il contenuto del documento.
2. **Non falsificabiltà:** la firma è la prova che solo il firmatario e nessun altro ha apposto la firma sul documento.
3. **Non riusabilità:** la firma fa parte del documento e non deve essere riutilizzabile su un altro documento.
4. **Non alterabilità:** una volta firmato, il documento non deve poter essere alterato.
5. **Non contestabilità:** il firmatario non può rinnegare con successo la paternità dei documenti firmati; la firma attesta la volontà del firmatario di sottoscrivere quanto contenuto nel documento.

In generale, nel caso di firma digitale, il mittente di un messaggio può firmarlo grazie alla sua chiave privata (che solo lui possiede), ma tutti sono in grado di verificare l'autenticità della firma grazie alla chiave pubblica che è globalmente nota. La firma può poi essere abbinata alla normale cifratura, ottenendo così messaggi firmati e cifrati.

La firma digitale viene realizzata tramite tecniche crittografiche a chiave pubblica insieme all'utilizzo di particolari funzioni matematiche, chiamate funzioni hash unidirezionali. Il processo di firma digitale passa attraverso tre fasi:

1. Generazione dell'impronta digitale
2. Generazione della firma

3. Apposizione della firma

Nella prima fase viene applicata al documento in chiaro una funzione di hash (vedi Appendice **Errore. L'origine riferimento non è stata trovata.**) appositamente studiata che produce una stringa binaria di lunghezza costante e piccola, normalmente 128 o 160 bit, chiamata *message digest* o *digital fingerprint*, ossia impronta digitale. Queste funzioni devono avere due proprietà:

1. Unidirezionalità, ossia dato x è facile calcolare f(x), ma data f(x) è computazionalmente difficile risalire a x.
2. Prive di collisioni (*collision-free*), ossia a due testi diversi deve essere computazionalmente impossibile che corrisponda la medesima impronta ($x \neq y$ allora $f(x) \neq f(y)$).

L'utilità dell'uso delle funzioni di hash consente di evitare che per la generazione della firma sia necessario applicare l'algoritmo di cifratura, che è intrinsecamente inefficiente, all'intero testo che può essere molto lungo.

La seconda fase, la generazione della firma, consiste semplicemente nella cifratura con la propria chiave privata dell'impronta digitale generata in precedenza. In questo modo la firma risulta legata, da un lato (attraverso la chiave privata usata per la generazione) al soggetto sottoscrittore, e dall'altro (attraverso l'impronta) al teso sottoscritto. In realtà, l'operazione di cifratura viene effettuata, anziché sulla sola impronta, su una struttura di dati che la contiene insieme con altre informazioni utili, quali ad esempio l'indicazione della funzione di hash usata per la sua generazione. Sebbene tali informazioni possano essere fornite separatamente rispetto alla firma, la loro inclusione nell'operazione di codifica ne garantisce l'autenticità.

Nell'ultima fase, la firma digitale generata precedentemente viene aggiunta in una posizione predefinita, normalmente alla fine, al testo del documento.

L'operazione di verifica della firma digitale viene effettuata ricalcolando, con la medesima funzione di hash usata nella fase di firma, il valore dell'impronta digitale (hash "fresco"); poi si decifra con la chiave pubblica del firmatario la firma (hash firmato) e si controlla che il valore così ottenuto coincida con l'hash "fresco".

Gli algoritmi per la firma digitale più utilizzati sono RSA e DSA, mentre per le funzioni di hash abbiamo SHA-1, MD2 [RFC 1319], MD4 [RFC 1320], MD5 [RFC 1321]

A.3 Algoritmi asimmetrici

A.3.1 Diffie Helmann

Il protocollo di Diffie–Hellman permette a due interlocutori che non hanno mai avuto nessun precedente contatto di accordarsi su una chiave segreta utilizzando un canale pubblico.Alcune definizioni necessarie per poter illustrare il protocollo:

Sia Z l'insieme dei numeri interi.

1. Un numero $p \in Z$ è detto primo se è divisibile solo per 1 e per sé stesso.
2. Un numero $p \in Z$ si dice primo rispetto a $q \in Z$ se il massimo comune divisore tra p e q è 1. Si dice anche che p e q sono primi tra loro.
3. Sia Z_p l'insieme dei numeri interi tra 0 e p-1 compresi, ovvero l'insieme $\{0,\ldots,p\text{-}1\}$.
4. Sia Z^*_p il sottoinsieme di Z_p comprendente solo i numeri primi rispetto a p. Se p è un numero primo allora $Z^*_p = \{0,\ldots,p\text{-}1\}$.
5. Sia a un numero intero e m un intero positivo, allora indichiamo con $a \bmod m$ il resto della divisione di a per m, ovvero $a \bmod m = a - \left\lfloor \frac{a}{m} \right\rfloor m$.
6. Siano a e b due numeri interi e m un intero positivo, allora diciamo che $a \equiv b \pmod m$ se $a \bmod m = b \bmod m$, ovvero se m divide $b - a$.
7. Un numero $\alpha \in Z^*_p$ è detto elemento generatore di Z^*_p se il più piccolo numero q tale che $\alpha^q \equiv 1 \pmod p$ è p-1.

Detti A e B i due interlocutori, il protocollo procede come segue:

1. A e B si accordano pubblicamente su un numero primo p e su un numero α che sia elemento generatore di Z^*_p.
2. B sceglie un numero a_1 tale che $1 \le a_1 \le p-2$, calcola $b_1 = \alpha^{a_1} \bmod p$ e lo manda a A.
3. A sceglie un numero a_2 tale che $1 \le a_2 \le p-2$, calcola $b_2 = \alpha^{a_2} \bmod p$ e lo manda a B.
4. La chiave segreta è $K = \alpha^{a_1 a_2} \bmod p$: B può calcolarla come $K = b_2^{a_1} \bmod p$, mentre A può calcolarla come $K = b_1^{a_2} \bmod p$.

La sicurezza del protocollo di Diffie-Hellman, ovvero il fatto che un intruso non possa calcolare la chiave K avendo a disposizione solo i dati pubblici p, α , b_1, b_2, si basa sulla congettura dell'intrattabilità del problema dei logaritmi discreti in Z_p, per cui conoscendo solo i dati pubblici è molto difficile ricavare a_1 e a_2. Così com'è, ovvero senza alcuna forma di autenticazione (come ad esempio dei certificati), è però soggetto ad un attacco di tipo man–in–the–middle. Se infatti un intruso, che chiamiamo X, si pone nel mezzo della comunicazione può intercettare b_1 e b_2 e mandare, rispettivamente, a B un valore $b_3 = \alpha^{a_3} \bmod p$ e ad A un valore $b_4 = \alpha^{a_4} \bmod p$ (calcolati a partire da dei valori a_3 e a_4 scelti da lui). In questo modo X riuscirà a stabilire una chiave $K_1 = \alpha^{a_1 a_3} \bmod p$ con B e una chiave $K_2 = \alpha^{a_2 a_4} \bmod p$ con A, mentre B e A penseranno di essersi accordati su una chiave tra di loro: così X avrà la successiva comunicazione tra A e B in chiaro, e potrà leggere i messaggi, modificarli, inserirne altri.

A.4 Algoritmi per la firma digitale

A.4.1 RSA

RSA (dai nomi dei tre ricercatori che lo proposero, Rivest, Shamir e Adleman) è probabilmente il più noto algoritmo crittografico asimmetrico, e può essere utilizzato sia per la cifratura che per la firma. I passi preliminari sono i seguenti (per alcune definizioni si veda il paragrafo precedente):

1. A sceglie due numeri primi molto grandi, p e q.
2. A calcola n = pq e $\varphi(n) = (p-1)(q-1)$ (la funzione $\varphi(n)$; detta funzione fi di Eulero, indica il numero di elementi di Z_n primi rispetto a n).
3. A sceglie un numero b, tale che $0 \le b \le \varphi(n)$, primo rispetto a $\varphi(n)$.
4. A calcola $a = b^{-1} \bmod \varphi(n)$ (ovvero trova quel numero a tale che $ab \equiv 1 (\bmod \varphi(n))$, per come è stato scelto b tale numero esiste ed è unico, e si può calcolare con un algoritmo detto di Euclide).
5. A rende disponibili i valori n e b, che costituiscono la sua chiave pubblica, e tiene segreti i valori p, q e a, che costituiscono la sua chiave privata.

A questo punto l'uso di RSA per la cifratura funziona come segue:

1. Sia x il testo in chiaro del messaggio che B vuole mandare a A cifrandolo. B calcola $x = y^b \bmod n$ e lo manda ad A.
2. A riceve *y* da B, e può decifrare il messaggio calcolando $x = y^a \bmod n$.

RSA può essere utilizzato per la firma nel modo seguente:

1. Sia x il testo in chiaro del messaggio che A vuole mandare a B firmandolo. A calcola la firma come $y = x^a \bmod n$ e la manda a B.
2. B riceve x e y da A, e verifica che sia $x = y^b \bmod n$. Se è così la firma è autentica, altrimenti è falsa.

La sicurezza di RSA si basa sulla congettura dell'intrattabilità del problema della scomposizione in fattori primi, per cui un intruso, noti n e b, non può calcolare a perché non conosce p e q (e quindi $\varphi(n)$).

A.5 Algorotmi per le funzioni di Hash

A.5.1 Definizione funzioni di Hash

Una funzione di hash è una funzione che, dato un qualunque messaggio di lunghezza arbitraria, ne produce un'impronta (detta digest) di lunghezza prefissata (di solito dell'ordine di 100-200 bit). L'utilità di una funzione di hash è nell'utilizzo dell'impronta come rappresentazione compatta del messaggio stesso, tipicamente firmando l'impronta anziché l'intero messaggio; perché questo possa avvenire è desiderabile che la funzione presenti due proprietà particolari, sia cioè senza collisioni e unidirezionale.

- Una funzione di hash $h(x)$ è detta *debolmente senza collisioni* se, dato un messaggio x, è computazionalmente impraticabile trovare un messaggio $x' \neq x$ tale che $h(x') = h(x)$.

- Una funzione di hash *h(x)* è detta *fortemente senza collisioni* se è computazionalmente impraticabile trovare due messaggi *x* e *x'* tali che $x' \neq x$ e *h(x') = h(x).*

- Una funzione di hash *h(x)* è detta *unidirezionale* se, data un'impronta z, è computazionalmente impraticabile trovare un messaggio *x* tale che *h(x) = z*.

Le funzioni di hash possono essere utilizzate insieme con delle chiavi segrete per ottenere autenticazione e integrità dei messaggi, realizzando così dei MAC (Message Authentication Code). Più precisamente, possiamo definire algoritmo di MAC una funzione $h_k(x)$, parametrizzata da una chiave *k*, con le seguenti proprietà:

- Dati una chiave *k* e un input *x*, è facile calcolare $h_k(x)$. Tale valore è detto brevemente MAC.

- Dato un input *x* di lunghezza qualsiasi la funzione genera un output $h_k(x)$ di lunghezza prefissata.

- Noto un certo numero di coppie $(x_i, h_k(x_i))$ è computazionalmente impraticabile calcolare qualunque altra coppia $(x, h_k(x))$ per un qualsiasi nuovo input $x \neq x_i$ senza conoscere *k*.

A.5.2 SHA-1

La funzione di hash SHA (Secure Hash Algorithm) produce un message digest di 160 bit per ogni messaggio di lunghezza inferiore a 2^64 bi. Vediamo le principali fasi algoritmiche:

1. Il messaggio viene completato con dei bit per ottenere multipli di 512 bit (padding): all'inizio del messaggio si mette un 1 seguito da tanti zeri quanti sono necessari affinché il messaggio sia più corto di 64 bit di un multiplo di 512 bit, mentre alla fine del messaggio si mettono 64 bit che danno la lunghezza del messaggio prima del padding.

2. Vengono inizializzate 5 variabili di 32 bit l'una (A,B,C,D,E).

3. L'algoritmo processa blocchi di 512 bit alla volta; ogni blocco viene processato attraverso 4 funzioni non lineari, ognuna applicata 20 volte. Senza entrare nei dettagli implementativi di queste funzioni che fondamentalmente eseguono operazioni logiche (and, or, not, xor) e operazioni di shifting, abbiamo come risultato finale che ogni blocco di 512 bit processato modifica le cinque variabili iniziali. Una volta processati tutti i blocchi che costituiscono il messaggio, concatenando tali variabili otteniamo un hash a 5 x 32 = 160 bit del messaggio.

SHA–1 producendo un hash a 160 bit, risulta più robusto ad attacchi crittografici rispetto a MD4 e MD5 che, invece, producono un message digest di 128 bit.

ACRONIMI

AES	Advanced Encryption Standard
AH	Authentication Header
ASN1	Abstract Sintax Notation 1
BER	Basic Encoding Rules
CA	Certification Authority
CBC	Cipher Block Chaining
CER	Canonical Encoding Rules
CMP	Certificate Management Protocol
CRL	Certificate Revocation List
CSL	Certificate Suspension List
DER	Distinguished Encoding Rules
DES	Data Encryption Standard
DH	Diffie–Hellman
DN	Distinguished Name
DNS	Domain Name System
DoS	Denial of Service
DPD	Dead Peer Detection
ESN	Extended Sequence Number
ESP	Encapsulating Security Payload
FQDN	Fully Qualified Distinguished Name
FTP	File Transfer Protocol
HTTP	HyperText Transfer Protocol
ICMP	Internet Control Message Protocol
ICSA	International Computer Security Association
ICV	Integrity Check Value
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
IP	Internet Protocol
IPSec	Internet Protocol Security

IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISAKMP	Internet Security Association and Key Management Protocol
IV	Initialization Vector
LDAPv2	Lightweight Directory Access Protocol versione 2
MAC	Message Authentication Code
MD2	Message Digest 2
MD5	Message Digest 5
MTU	Maximum Transfer Unit
NAT	Network Address Translation
OID	Object Identifier
OSI	Open System Interconnection
PEM	Private Enhanced Mail
PFS	Perfect Forward Secrecy
PKI	Public Key Infrastructure
PKCS #7	Public-Key Cryptography Standards #7
PKCS #10	Public-Key Cryptography Standards #10
RFC	Request For Comment
RSA	Rivest–Shamir–Adleman
SA	Security Association
SAD	Security Association Database
SAS	Sistema di Accesso Sicuro
SCEP	Simple Certificate Enrollment Protocol
SHA-1	Secure Hash Algorithm
SPD	Security Policy Database
SPI	Security Parameters Index
SSL	Secure Socket Layer
S-VPN	Secure Virtual Private Network
TCP	Transmission Control Protocol
TDES	Triple Data Encryption Standard
TFC	Traffic Flow Confidentiality
TFTP	Trivial File Transfer Protocol

UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VPN	Virtual Private Network

BIBLIOGRAFIA

[1] S. Kent, R. Atkinson. *Security Architecture for the Internet Protocol. Standards Track* RFC 2401, IETF, Novembre 1998.

[2] S. Kent, R. Atkinson. *IP Autentication Header. Standards Track* RFC 2402, IETF, Novembre 1998

[3] S. Kent, R. Atkinson. *IP Encapsulating Security Payload (ESP). Standards Track* RFC 2406, IETF, Novembre 1998.

[4] D. Piper.*The Internet IP Security Domain of Interpretation for ISAKMP* RFC 2407, IETF, Novembre 1998.

[5] D. Maughan, M. Schertler, M. Schnider, J. Turner. *Internet Security Association and Key Management Protocol (ISAKMP). Standards Track* RFC 2408, IETF, Novembre 1998.

[6] D. Harkins, D. Carrel. *The Internet Key Exchange (IKE). Standards Track* RFC 2409, IETF, Novembre 1998.

[7] B. Kaliski *PKCS #1: RSA Encryption. Informational* RFC 2313, RSA Laboratories, Marzo 1998.

[8] B. Kaliski *PKCS #7: Cryptographic Message Syntax. Informational* RFC 2315, RSA Laboratories, Marzo 1998.

[9] M. Nystrom, B. Kaliski *PKCS #9: Selected Object Classes and Attribute Types. Informational* RFC 2985, RSA Laboratories, Novembre 2000.

[10] B. Kaliski *PKCS #10: Certification Request Syntax. Informational* RFC 2314, RSA Laboratories, Marzo 1998.

[11] M. Nystrom, B. Kaliski *PKCS #10: Certification Request Syntax Specification. Informational* RFC 2986, RSA Laboratories, Novembre 2000.

[12] J. Linn *Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures* RFC 1421, IAB IRTF PSRG, IETF PEM WG, Febbraio 1993.

[13] S.Kent *Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management* RFC 1422, IAB IRTF PSRG, IETF PEM WG, Febbraio 1993

[14] D. Balenson *Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers* RFC 1423, IAB IRTF PSRG, IETF PEM WG, Febbraio 1993

[15] B. Kaliski *Privacy Enhancement for Internet Electronic Mail: Part IV: Key Certification and Related Services* RFC 1424, RSA Laboratories, Febbraio 1993

[16] S. Kille, M. Wahl, A. Grimstad *Using Domains in LDAP/X.500 Distinguished Names Standards Track* RFC 2247, IETF, Gennaio 1998.

[17] S. Boeyen, T. Howes, *Internet X.509 Public Key Infrastructure LDAPv2 Schema Standards Track* RFC 2587,IETF, Giugno 1999.

[18] R. Housley, P. Hoffman, *Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP Standards Track* RFC 2585, IETF, Maggio 1999.

[19] R. Housley, W. Ford, W. Polk *Internet X.509 Public Key Infrastructure Certificate and CRL Profile Standards Track* RFC 2459, IETF, Gennaio 1999.

[20] R. Housley, W. Polk, W. Ford, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile Standards Track* RFC 3280, IETF, Aprile 2002.

[21] W. Polk, R. Housley, L. Bassham, *Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile Standards Track* RFC 3279, IETF, Aprile 2002.

[22] C. Adams, S. Farrell, *Internet X.509 Public Key Infrastructure Certificate Management Protocols Standards Track* RFC 2510, IETF, Marzo 1999.

[23] M. Myers, C. Adams, D. Solo, *Internet X.509 Certificate Request Message Format Standards Track* RFC 2511, IETF, Marzo 1999.

[24] M. Myers, R. Ankney, A. Malpani, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP Standards Track* RFC 2560, IETF, Giugno 1999.

[25] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels Best Current Practice* RFC 2119, IETF, Marzo 1997.

[26] X.Liu, C.Madson, D. McGrew, A. Nourse, *Cisco System's Simple Certificate Enrollment Protocol (SCEP)*, SCEP, Febbraio 2000.

[27] X.Liu, C.Madson, D. McGrew, A. Nourse, *Cisco System's Simple Certificate Enrollment Protocol (SCEP)*, INTERNET DRAFT draft-nourse-scep-10.txt, 29 Giugno 2004.

[28] Rodney Thayer, *PKI Requirements for IP Security,* INTERNET DRAFT draft-ietf-ipsec-pki-req-00.txt, Settembre 1998.

[29] Rodney Thayer, Charles Kunzinger, Paul Hoffman, *A PKIX Profile for IKE,* INTERNET DRAFT draft-ietf-ipsec-pki-req-03.txt, Ottobre 1999.