UNIVERSITÀ POLITECNICA DELLE MARCHE

Dipartimento di Ingegneria dell'Informazione

TESI DI LAUREA MAGISTRALE

# Thesis Title


*Author:*
Davide DE ZUANE

*Supervisor:*
Dr. Paolo SANTINI

October 1, 2024

*?Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism.?*

Dave Barry

UNIVERSITÀ POLITECNICA DELLE MARCHE

# *Abstract*

Faculty Name
Dipartimento di Ingegneria dell'Informazione

Doctor of Computer Sciences

**Thesis Title**

by Davide DE ZUANE

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

# *Acknowledgements*

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **DH** | **D**iffie **H**ellman |
| **KE** | **K**ey **E**xchange |
| **PQ** | **P**ost **Q**uantum |
| **IKE** | **I**nternet **K**ey **E**xchange |
| **KEM** | **K**ey **E**ncapsulation **M**echanism |
| **PRF** | **P**seudo **R**andom **F**unction |
| **MTU** | **M**aximum **T**ransmission **U**nit |
| **ISP** | **I**nternet **S**ervice **P**rovider |

# List of Symbols

| | | |
|---|---|---|
| $\mid$ | concatenazione | |
| $a$ | distance | m |
| $P$ | power | W ($\mathrm{J\,s^{-1}}$) |
| $\omega$ | angular frequency | rad |

*For/Dedicated to/To my…*

# Chapter 1

# Introduction

Introduzione al problema e al contensto in cui ci troviamo e le crtiticità presenti
Contributo apportato
Fare un'introduzione al problema e dire quale è stato il contributo.
Suddividere il contributo in:

- La parte di benchmarking, in cui andiamo a definire tutte quelle che sono le problematiche relative al post quantum
- la parte di implementazione

## 1.1 Post-Quantum

Il postquantum può rompere gli schemi di crittografia classici tuttavia ì sono nati quelli nuovi.

## 1.2 Problematiche

Tuttavia il quantum computing porta ad aumentare la dimensione della chiave, in questo modo si perde di efficienza. Inoltre ha particolare effetto sui sistemi a chiave pubblica, mentre su quelli basati su chiave segreta o funzioni di hash non sono molto vulnerabili a questo. Le sfide della post-quantum cryptography sono le seguenti:

- Occorre migliorare l'efficienza
- Occorre migliorarne l'usabilità

Ovvero occorre preparare il mondo per una transizione alla crittografia post-quantum
Dire quelle che sono le problematiche del post quantum Esempi di problematiche:

- non esiste il diffie hellman post quantum
- i messaggi diventano molto più lunghi (problemi per frammentazione, certificati molto più grandi, chiavi pubbliche piu grandi)

Queste hanno conseguenze importanti sui protocolli che ne fanno uso a causa dell'aumento di dimensioni dellee chiavi.
Gli schemi sono divisi in
- l1:aes128 - l3:aes192 - l5:aes256

# Chapter 2

# Fondamenti di Comunicazioni Sicure

*In questo capitolo esamineremo le problematiche relative alla sicurezza nelle comunicazioni. Inizieremo con un'analisi degli strumenti matematici fondamentali che sono alla base della protezione dei dati, esplorando le tecniche crittografiche e i loro principi teorici. Successivamente, ci concentreremo su come queste tecniche vengono effettivamente applicate per garantire la sicurezza nelle comunicazioni sulle reti di computer.*

## 2.1 Teoria

Dalla crittografia classica, come il cifrario di Cesare, fino alle tecniche più sofisticate del ventesimo secolo, come i sistemi di cifratura a chiave pubblica, la storia della crittografia è caratterizzata da una continua evoluzione e innovazione. Le fondamenta sui cui si basa non sono cambiate, andiamo a vedere strumenti matematici che ne fanno parte.

### 2.1.1 Hash Function

Una funzione **hash crittografiche** è una funzione matematica che prende in input un messaggio di lunghezza arbitraria e restituisce un output di lunghezza fissa, noto come digest.

$$H : \{0,1\}^* \rightarrow \{0,1\}^n \tag{2.1}$$

- $\{0,1\}^*$: rappresenta l'insieme di tutte le stringhe binarie di lunghezza arbitraria.
- $\{0,1\}^n$: rappresenta l'insieme delle stringhe binarie di lunghezza fissa $n$.

Le funzioni di hash crittografiche sono strumenti fondamentali nel campo della sicurezza informatica, progettate per garantire l'integrità e l'autenticità dei dati. Per questo motivo, a questo tipo di funzioni sono richieste le seguenti proprietà:

- **Resistenza alle collisioni**: devono essere progettate in modo tale che sia computazionalmente impraticabile invertire il processo, ovvero, dato il digest è dificile risalire al messaggio che lo ha prodotto.
- **Proprietà di diffusione**: una leggere variazione dell'input deve produrre un hash completamente diverso, questa è fondamentale per garantire che gli attaccanti non possano prevedere o manipolare il valore di hash a seguito di modifiche all'input.

### 2.1.2 Schemi crittografici

Uno schema di cifratura è un insieme di algoritmi e funzioni che definisce come trasformare un messaggio in chiaro (plaintext) in un messaggio cifrato (ciphertext) e viceversa, al fine di garantire la confidenzialità e la sicurezza delle comunicazioni. Formalmente possiamo rappresentarlo come una quintupla:

$$(\mathcal{P}, \mathcal{C}, \mathcal{K}, E, D)$$

- $\mathcal{P}$: Insieme dei messaggi in chiaro (plaintext).
- $\mathcal{C}$: Insieme dei messaggi cifrati (ciphertext).
- $\mathcal{K}$: Insieme delle chiavi utilizzate per la cifratura e decifratura, *key space*.
- $E : \mathcal{K} \times \mathcal{P} \to \mathcal{C}$: Funzione di cifratura.
- $D : \mathcal{K} \times \mathcal{C} \to \mathcal{P}$: Funzione di decifratura.

Deve esistere una relazione inversa tra le operazioni di cifratura e decifratura:

$$D(k, E(k, m)) = m \quad \forall m \in \mathcal{P}, k \in \mathcal{K} \tag{2.2}$$

Lo schema in *Fig. 2.1*, mostra il funzionamento generale di uno schema crittografico. Tuttavia andando a caratterizzare le chiavi utilizzate nelle operazioni di cifratura e decifratura possiamo dare una prima classificazione:

- se $K_1 = K_2$ allora si parla di un schema di crittografia *simmetrico*.
- se $K_1 \neq K_2$ allora si parla di schema di crittografia *asimmetrico*.

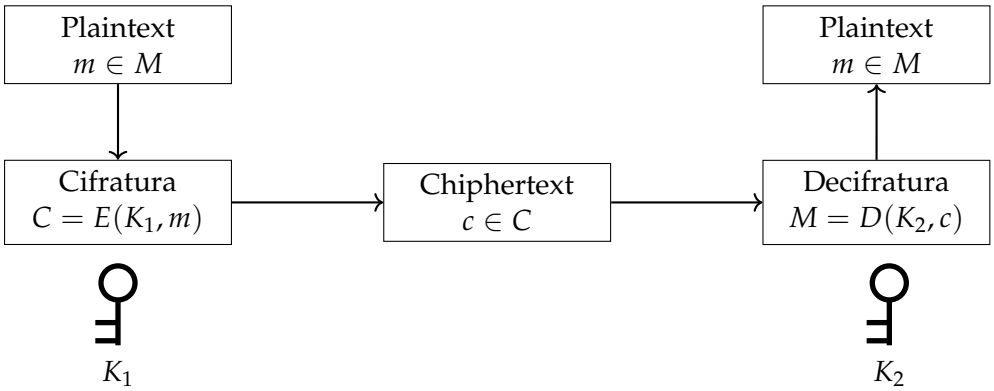


FIGURE 2.1: Funzionamento di uno schema crittografico

#### Simmetrici

Come descritto precedentemente, in uno schema simmetrico si utilizza la stessa chiave sia per le operazioni di cifratura che di decifratura. Ciò implica che le due parti coinvolte nella comunicazione debbano possedere la medesima chiave segreta, nota anche come chiave pre-condivisa (PSK - Pre-Shared Key).
Questa caratteristica fondamentale rende gli schemi di crittografia simmetrica particolarmente veloci ed efficienti, esempi ne sono AES e DES.

Le loro caratteristiche li rendono ideali per:

- *Cifratura* di Dati: proteggere file e database memorizzati su disco, garantendo che le informazioni sensibili rimangano riservate anche in caso di accesso non autorizzato. Sia proteggere i dati mentre vengono trasmessi su reti.
- *HMAC* (Hash-based Message Authentication Code): combinati con funzioni di hash, gli algoritmi simmetrici possono generare codici HMAC, che forniscono autenticità e integrità ai messaggi. Fondamentale per garantire che i dati non vengano manomessi durante la trasmissione.

### Asimmetrici

In uno schema di cifratura asimmetrica, detto anche a **chiave pubblica**, lo spazio delle chiavi $\mathcal{K}$ è costituito da una coppia di chiavi $(k_{\text{pub}}, k_{\text{priv}})$, dove:

- La chiave pubblica $k_{\text{pub}}$ viene condivisa liberamente e utilizzata da chiunque per cifrare messaggi destinati al proprietario della chiave.
- La chiave privata $k_{\text{priv}}$ è mantenuta segreta dal proprietario e viene utilizzata per decifrare i messaggi cifrati con la corrispondente chiave pubblica.

Quindi le due funzioni si riscrivono come:

$$E : \mathcal{K}_{\text{pub}} \times \mathcal{P} \to \mathcal{C} \tag{2.3}$$

$$D : \mathcal{K}_{\text{priv}} \times \mathcal{C} \to \mathcal{P} \tag{2.4}$$

Le due chiavi sono matematicamente legate, ma è computazionalmente difficile ottenere la chiave privata a partire da quella pubblica. Il funzionamento si basa sul concetto di **trapdoor** che rende possibile una funzione (come la cifratura o la decifratura) semplice per chi conosce un segreto (la chiave privata) ma estremamente difficile per chi non lo conosce.

Uno dei principali utilizzi della crittografia asimmetrica è il **Key-Exchange**, il quale consente di scambiarsi un'informazione segreta su un canale pubblico. La procedura più conosciuta è quella proposta da *Diffie-Hellman* ed è mostrata in *Fig.* 2.2

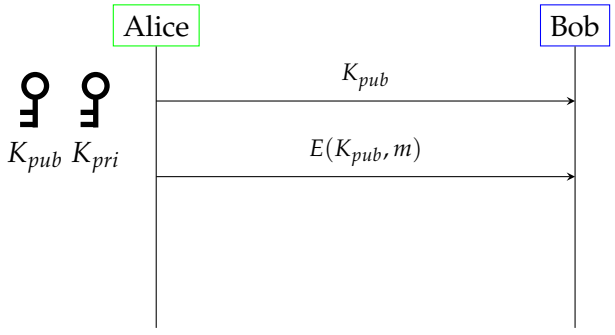


FIGURE 2.2: Scambio di Chiave Difie-Hellman

Le **firme digitali**, sono un meccanismo chiave per garantire l'autenticità e l'integrità dei messaggi. In questo caso si fornisce sia il messaggio che un digest del messaggio firmato, in questo modo chi lo riceve può utilizzare la chiave pubblica per verificare che l'hash firmato equivalga a quello calcolato. Questa pratica si utilizza per garantire integrità e autenticità.

### 2.1.3 Confronto

#### Key Distribution

L'assunto che si è fatto in entrambi le tipologie di schema è che l'altra parte della comunicazione avesse ottenuto in qualche modo la chiave. Tuttavia la distribuzione delle chiavi è un problema importante per il crittosistema.

Le distribuzione delle chiavi per crittosistemi simmetrici deve avvenire tramite un canale segreto, per questo motivo si utilizzano le seguenti modalità:

- *Manuale*: vengono installate manualmente coppie di chiavi per ogni nodo che si vuol far comunicare, se si vogliono far comunicare $n$ nodi sono necessarie $n(n-1)/2$ chiavi. L'approccio è robusto ma impraticabile per reti di grandi dimensioni come mostrato in *Fig.* 2.3.
- *Key Distribution Center (KDC)*: da un'approccio decentralizzato si passa ad uno centralizzato, in cui è presente un server che fa da intermediario fidato per la distribuzione delle chiavi.

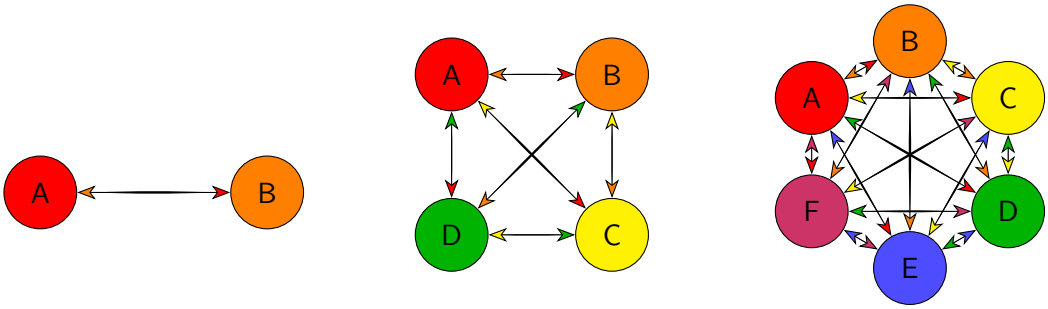


FIGURE 2.3: Esplosione combinatoria

Per loro natura le chiavi pubbliche sono liberamente distribuite, non è dunque necessario un canale segreto. Tuttavia in questo caso si ha il problema dell'autenticità, ovvero che la chiave provenga realmente dalla fonte dichiarata. Per questo nasce la *Public Key Infrastructure (PKI)*, che tramite l'utilizzo dei certificati X.509 consente la distribuzione sicura di chiavi pubbliche.

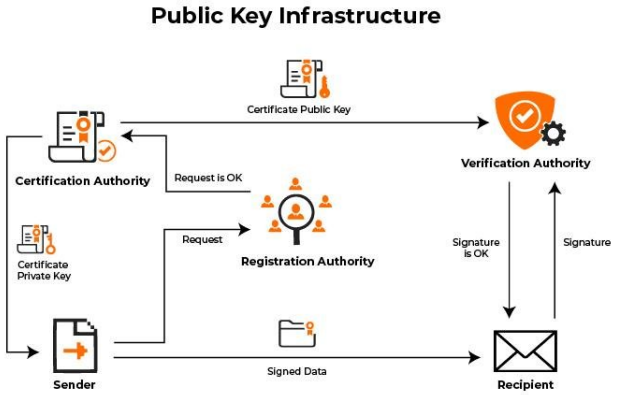


FIGURE 2.4: Public Key Infrastructure

Questo approccio consente una gestione scalabile delle chiavi.

#### Attacchi Quantum

Gli schemi asimmetrici sono vulnerabili alla minaccia del quantum computer Parlare degli algoritmi di Shor e ....

### 2.1.4 Sicurezza

Il **security level** è una misura della forza che una primitiva crittografica raggiunge rispetto ad attacchi. Solitamente viene espresso come un numero di "bit di sicurezza", dove $n$-bit di sicurezza significa che l'attaccante dovrebbe eseguire $2^n$ operazioni per romperlo.

- Per i cifrari simmetrici il livello di sicurezza è pari alla dimensione del key-space. Equivale ad un attacco a forza bruta.
- La sicurezza dei cifrari simmetrici si basa su problemi matematici noti. Tuttavia, gli attacchi contro gli attuali sistemi a chiave pubblica sono sempre più veloci della ricerca a forza bruta dello spazio delle chiavi.

Il NIST (National Institute of Standards and Technology) ha introdotto livelli di sicurezza, definiti in *Tabella 2.1* per gli algoritmi di cifratura asimmetrica e post-quantistica come parte della sua iniziativa per standardizzare algoritmi che resistano anche ai computer quantistici.

| Security Level | Descrizione |
|---|---|
| **Livello 1** | Sicurezza equivalente alla cifratura simmetrica con chiavi da 128 bit, come AES-128. |
| **Livello 2** | Sicurezza equivalente ad attacchi contro SHA-256, con complessità circa pari a 128 bit. Leggermente più sicuro del livello 1. |
| **Livello 3** | Sicurezza equivalente alla cifratura simmetrica con chiavi da 192 bit, come AES-192. |
| **Livello 4** | Sicurezza equivalente ad attacchi contro SHA-384. Leggermente più sicuro del Livello 3. |
| **Livello 5** | Sicurezza equivalente alla cifratura simmetrica con chiavi da 256 bit, come AES-256. |

TABLE 2.1: Security Levels definiti dal NIST

## 2.2 Applicazioni

Le reti, per loro natura, rappresentano un mezzo di comunicazione intrinsecamente insicuro, soprattutto quando operano in modalità broadcast. In questo contesto, la crittografia riveste un ruolo cruciale nel garantire la sicurezza dei dati scambiati tra entità remote. I crittosistemi, ossia le applicazioni crittografiche, integrano algoritmi di cifratura, autenticazione e gestione delle chiavi per fare in modo che vengano rispettati i requisiti di sicurezza per le informazioni trasmesse.

Il modello di riferimento per la comunicazione su Internet è il modello **TCP/IP**, il quale suddivide il processo di trasmissione dei dati in vari livelli, ciascuno con delle funzioni specifiche che non si sovrappongono con quelle degli altri livelli. Come mostrato in *Figura 2.5*, è possibile applicare la sicurezza ai vari livelli della pila e di lato sono riportati i protocolli che vengono utilizzati.

- SSH (Secure Shell): Protegge l'accesso remoto e il trasferimento di file, fornendo autenticazione e cifratura.
- TLS (Transport Layer Security): permette di instaurare una connessione TCP sicura. Viene utilizzato per cifrare e autenticare i dati tra client e server.
- IPsec (Internet Protocol Security): Protegge i pacchetti IP scambiati tra due nodi, fornendo autenticazione, integrità e cifratura.

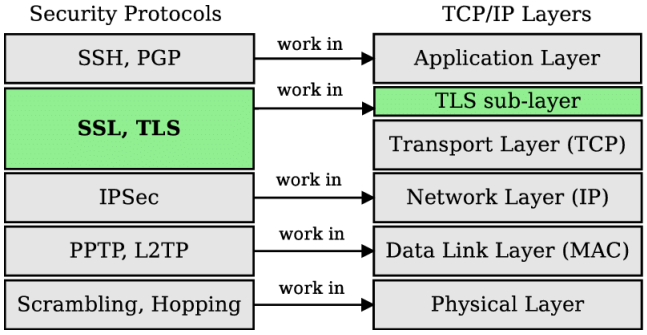


FIGURE 2.5: Modello TCP/IP con Protocolli di Sicurezza

Ogni protocollo avrà le proprie caratteristiche, tuttavia fare sicurezza a L3 dello stack TCP/IP offre un vantaggio significativo: poiché tutti gli strati superiori dipendono da esso per la trasmissione dei dati, non è necessario apportare modifiche ai singoli protocolli o applicazioni che si basano su di esso. Questo consente di implementare soluzioni di sicurezza centralizzate e trasparenti, senza dover intervenire su ciascun servizio o applicazione a livello più alto.

## 2.3 IPsec

IPsec (Internet Protocol Security) è un insieme di protocolli standard, rappresentati in *Figura 2.6*, utilizzati in modo tale da fornire meccanismi per l'autenticazione, la cifratura e l'integrità dei dati trasmessi tra due o più dispositivi, proteggendo così le comunicazioni IP da intercettazioni e manomissioni.

### 2.3.1 Architettura

Come definito dall'RFC 1825, l'architettura di IPsec è composta dai seguenti componenti:

- **AH (Authentication Header)**: si tratta di un protocollo di sicurezza che fornisce autenticazione e integrità dei dati, garantendo che i pacchetti non vengano modificati durante la trasmissione. Non offre cifratura, quindi i dati rimangono in chiaro.
- **ESP (Encapsulating Security Payload)**: protocolo che fornisce cifratura per garantire la riservatezza dei dati, oltre a integrità e autenticazione opzionale. ESP è il protocollo più utilizzato per garantire sia sicurezza che riservatezza.
- **SA (Security Association)**: un insieme di parametri che definisce come i dati devono essere protetti durante la comunicazione tra due entità su una rete. Ogni SA contiene le informazioni necessarie per stabilire e mantenere una connessione sicura.
- **IKE (Internet Key Exchange)**: protocollo che consente di negoziare, autenticare e distribuire dinamicamente le chiavi crittografiche che vengono poi impiegate dai protocolli di sicurezza per proteggere le comunicazioni.
- **Algoritmi**: gli algoritmi crittografici e di hashing utilizzati per ottenere sicurezza.

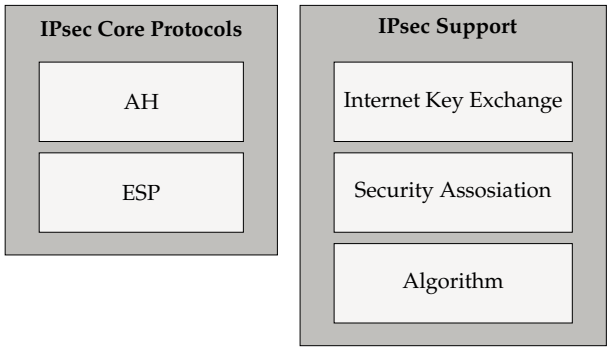


FIGURE 2.6: IPsec Protolocol Suite

### 2.3.2 Security Association

IP è un protocollo *stateless*, ovvero non mantiene informazioni o stato relativo alle connessioni o ai pacchetti che gestisce. Tuttavia affinchè IPsec possa garantire la sicurezza è necessario che mantenga il contesto di ogni connessione, le principali motivazioni sono:

- *Replay Protection*: per evitare attacchi di tipo reply, IPsec tiene traccia dei numeri di sequenza dei pacchetti, un'inforamzione di stato che va mantenuta per ogni connessioni e che IP non fa nativamente.
- *Connessioni Multiple*: in uno scenario di rete complesso, un singolo dispositivo potrebbe avere più connessioni sicure in corso simultaneamenamente, ognuna delle quali ha i propri parametri di sicurezza. IPsec deve tenere traccia di queste informazioni per sapere come trattare i pacchetti in entrata e uscita in base alla connessione a cui appartengono.

- *Protezione*: i protocolli di sicurezza AH e ESP richiedono di conoscere le chiavi crittografiche corrette e gli algoritmi utilizzati per cifrare e decifrare i pacchetti.

IP diventa in grado di mantenere un'insieme di informazioni di stato grazie al concetto di *Security Association* (SA). Più precisamente si tratta di un'insieme di parametri che servono per associare a ciascun canale uno stato condiviso tra le entità coinvolte nello comunicazione, tra questi abbiamo:

- *Security Parameter Index* (SPI): un'identificatore della SA.
- *Destination Address*: serve all'host per determinare quale SA utilizzare.
- *Lifetime*: il tempo di vita della SA, si obbliga a refresh periodici.
- *Protocol Identifier*: determina il tipo di protezione da applicare ai pacchetti, dunque anche chiavi e algoritmi associati.
- Altri parametri opzionali, per una lista completa fare riferimento all'RFC.

La SA è caratterizzata dall'essere un canale *simplex*, dunque al fine di stabilire un canale di comunicazione bidirezionale IPsec tra due entità occorrono due SA unidirezionali di verso opposto. La *Figura* 2.7 mostra il tunnerl virtuale in esecuzione tra i due host.

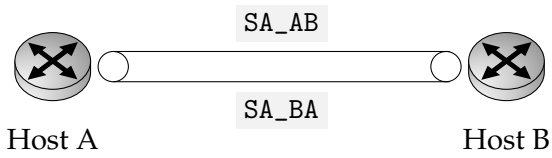


FIGURE 2.7: Security Association bidirezionali

### 2.3.3 IKE

In un contesto come quello delle reti potremmo avere che lo stesso nodo ha connessini IPsec multiple, le SA consentono di distinguere e identificare in modo univoco la configurazioni di sicurezza da applicare alla comunicazione.
Tuttavia queste SA come si configurano?. IPsec prevede tecniche di negoziazione delle SA di tipo:

- *Manuale*: occorre configurire manualmente le chiavi e le impostazioni di sicurezza per ciascun dispositivo o punto finale di comunicazione.
- *Automatico*: si utilizzanoprotocolli per stabilire automaticamente le chiavi di crittografia e le politiche di sicurezza senza intervento umano diretto.

L'utilizzo di tecniche di negoziazione automatica offre un approccio sicuro, flessibile e scalabile alla gestione delle Security Association, un esempio di questo è IKE definito nell'`RFC 7296`. Questo protocollo definisce una serie di scambi, mostrati in *Figura* 2.8, al termine del quale i due peer avranno negioziati i parametri di sicurezza e le chiavi crittografiche per una SA.

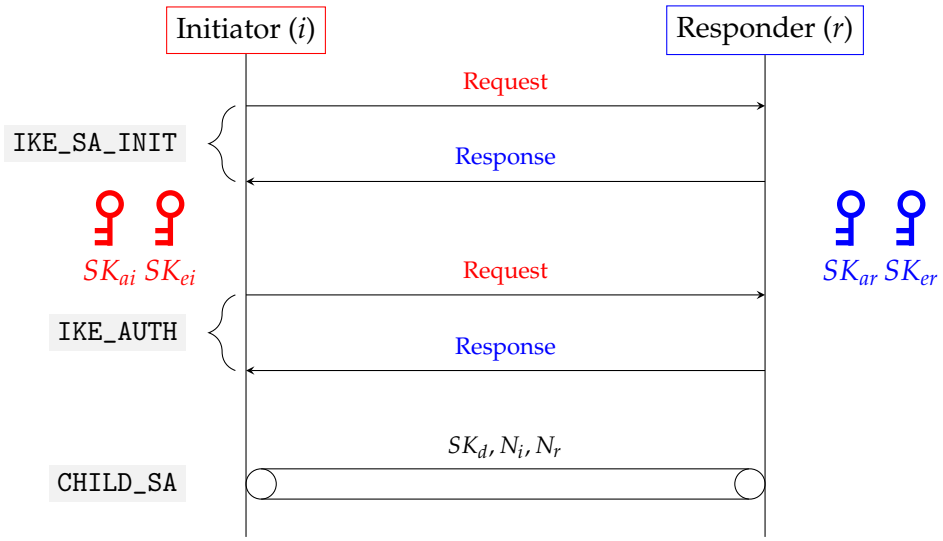


FIGURE 2.8: Fasi di Negoziazione del Protocollo IKEv2

### 2.3.4 IKE_SA_INIT

Lo scopo di questa prima fase è quello di creare una **IKE SA**, che consenta di rendere sicure i successivi scambi di dati al fine di realizzare una **IPsec SA**. Dunque funge da apripista al fine di stabilire quelli che sono i parametri di sicurezza al fine di avere una comunicazione sicura. Per questo motivo in questo scambio i peer si scambiano le seguenti informazioni:

TABLE 2.2: Tabella dei parametri e delle descrizioni

| Parametro | Descrizione |
|---|---|
| *SA* | Security Association, vengono negoziati i parametri per la SA |
| *KE* | Key Exchange, e nel caso classico è l'esponente DH |
| *N* | Nonce |

Al termine di questo scambio i due peer ottengono il *DH Shared Secret* (indicato con $g^{ir}$), il quale insisme ai nonce, consentirà di ottenere quelli che sono i parametri di sicurezza della $IKESA$ al fine di instauare un canale sicuro, per approfondimenti in appendice.

### 2.3.5 IKE_AUTH

Il risultato della fase precedente è un canale sicuro su cui comunicare, in quanto è cifrato e utenticato. Si questo hanno luogo gli scambi per instaurare la IPsec SA. In questa fase i nodi si autenticano mutuamente:

TABLE 2.3: Tabella dei parametri e delle descrizioni

| Parametro | Descrizione |
|---|---|
| *AUTH* | Payload che deve essere firmato affinchè ci sia autenticazione |
| *CERT* | Si allega il certificato digitale per la chiave pubblica |
| *CERTQ* | Si fa richiesta al peer di fornire il certificato |

Tutto il contenuto appena descritto è protetto mediante le chiavi segrete di quella direzione. Ciò è indicato mediante la notazione $SK\{...\}$ La modalità di autenticazione può essere: PSK, EAP oppure mediante chiave pubblica.

### 2.3.6 CHILD_SA

## 2.4 Problemi

IKEv2 utilizza come porotocollo a livello trasporto UDP per per inoltrare i propri messaggi. La maggior parte dei messaggi che i peer si scambiano hanno dimensioni relativamente piccole e quindi che non eccedono l'**MTU** di un pacchetto IP, tuttavia abbiamo degli scambi che richiedono un trasferimento di dati abbastanza grandi.

Per esempio nel caso di autenticazione tramite pubkey nella fase di `IKE_AUTH` è necessario trasferire il proprio certificato che in base allo schema di firma utilizzato può arrivare anche a diversi Kbyte di dimensione. In questi casi si verifica la frammentazione a livello IP.

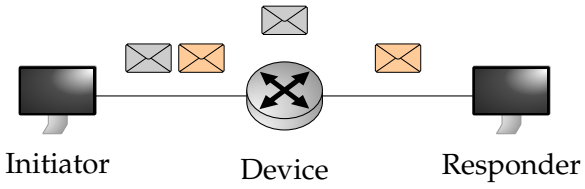


FIGURE 2.9: Drop Pacchetti

Diversi test hanno mostrato che nel caso in cui i peer si trovino in presenza di CGNAT potrebbero non istausarsi le SA. Questo è dovuto al fatto che i device degli ISP non consentono ai frammenti IP di passare attravers di loro, ovvero scartano i pacchetti e di conseguenza bloccano le comunicazioni IKE. Questo è riportato schematicamente in Fig. 2.9. Questo drop dei pacchetti avviene perchè esistono numerosi vettori di attacco che fanno affidamento sulla frammentazione IP, per questo motivo gli ISP operano un filtro su questa tipologia di pacchetti. Ance se in teoria uno dei requisiti del CGNAT definito dagli RFC è proprio consentire la frammentazione.

Per risolvere questa problematica e dunque consentire il passaggio dei messaggi attraverso i dispositivi di rete che non consentono il passaggio degli IP fragment attraverso di loro nell' RFC 7283 viene introdotta la IKEv2 *Message Fragmentation*. In cui la frammentazione dei messaggi è gestita direttamente da parte di chi implementa IKEv2

### 2.4.1 `IKE_INTERMEDIATE`

Per evitare che nel trasferimento di grandi dati ciò avvenga viene introdotto uno scambio aggiuntivo. Questo scambio è introdotto per quei casi in cui la dimensione dei dati da trasferire ecceda la dimensione massima che causerebbe la frammentazione IP. Questo scambio va fatto dopo la `IKE_INIT_SA` e prima della `IKE_AUTH` in questo modo è sia autenticato che cifrato tramite le chiavi negoziate dal primo scambio.

Questo scambio è posizionato qui in quanto nella `IKE_SA_INIT` per motivi di sicurezza non è possibile applicare la frammentazione. Di solito i messaggi sono piccoli abbastanza da non causare la frammentazione IP, tuttavia questo potrebbe cambiare se si utilizzano scambi di chiave QC-resistant; in quanto hanno chiavi pubbliche larghe e che quindi causerebbero frammentazione IP.

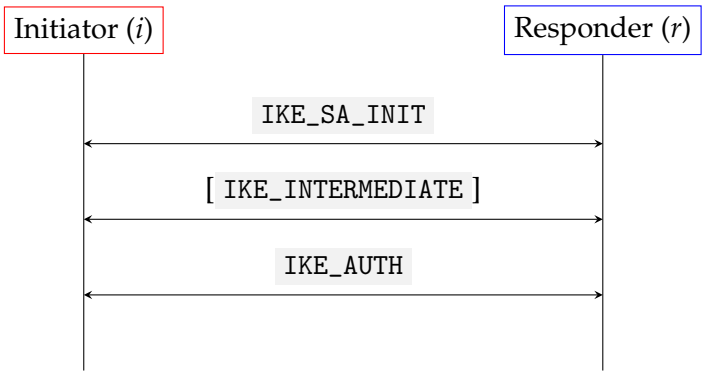


FIGURE 2.10: Scambio nuovo

Per questo viene aggiunto questo scambio che viene utilizzato per trasferire grandi quantità di dati.

L'utilizzo principale di questo scambio è quello di trasferire le chiavi pubbliche QC-resistant, tuttavia in generale può essere utilizzzato per trasferire qualsiasi tipologia di dato. Quindi il principale utilizzo è quello di fare un **enforcing** delle chiavi negoziate tramite DH al fine di renderle QC-resistant. Infatti se durante questo scambio si scambiano altre chiavi allora le coppie $\{SK_{a[i/r]}, SK_{e[i/r]}\}$ vengono aggiornate.

Permette di realizzare Multiple Key Exchange Gli scambi di chiave aggiuntivi vengono aggiunti alla proposal tramite `PQ_KEM_1`

Lo scambio `IKE_FOLLOWUP_KE` è introdotto specificatamente per trasferire dati sulla chiavi addizionali da realizzare in una CHILD SA. In questo caso le chiavi aggiuntive vengono utilizzate per aggiornare il KEYMAT

- flag `IKE_FRAGMENTATION_SUPPORT` : il peer dice di supportare la frammentazione IKEv2, affinchè venga utilizzata entrambi i peer devono supportarla.
- flag `INTERMEDIATE_EXCHANGE_SUPPORT` : il peer dice di supportare gli scambi intermedi

Una volta terminati gli scambi, per proteggere lo scambio `IKE_AUTH` e gli scambi successivi vengno utilizzata le ultime chiavi calcolate Dato che i dati trasferiti in questi scambi aggiuntivi vanno autenticati si aggiungono all' `AUTH` payload che poi andrà

Il supporto per lo scambio aggiuntivo viene comunicato aggiungendo all'interno dell `IKE_SA_INIT` il flag `IKE_INT_SUP` (che sta per Intermediate Exchange Support). Se anche il responder lo supporta lo includerà nel messaggio di risposta dello scambio.

Considerazioni, L'IKE fragmentation viene introdotta a causa del NAT tuttavia nel nostro caso di satelliti non ha senso utilizzarla in quanto non credo che si utilizzi il NAT soprattutto perchè introduce ritardi dovuti alla traduzione degli indirizzi

## 2.5 Post-Quantum

Un solo KEM con Kyber L1 usando come suite AES_GCM has vabene come certificato dilithium L1

Nel KEM quanti cifrano?

Cioè l'initiator manda il certificato e poi il responder cifra

# Appendix A

# IKEv2 Notation

Riportare i vari approfondimenti riguardanti IKE

per esempio come vengono generate le varie chiavi e il significato delle informazioni presenti tra i messaggi

### A.0.1 Authentication

L'autenticazione dei peer avviene effettuando il sign (o calcolando il MAC) di un payload che dipende dagli scambi precedenti. In particolare questo payload è cmposto da un ottetto che viene autenticato in base alla modalità di aunteticazione scelta:

- Nel caso di *PubKey* questo viene firmato con la chiave privata del peer e ne viene allegato il certificato della chiave pubblica
- Nel caso di *PSK* l'AUTH payload viene generato a partire dalla chiave condivisa a cui viene aggiunge della unpredicability tramite del padding e una prf

## A.1 Key Derivation

### A.1.1 IKE SA

Le chiavi in una IKE SA vengono derivate a partire dagli attributi dei dirrenti scambi. In particolare al termine del primo scambio viene calcolato il:

$$SKEYSEED = PRF(N_i|N_r, g^{ir})$$

A partire da questo sidder vengono generati i prametri di sicurezza da utililizzare per la IKE SA, questi sono derivati nel seguente modo:

$$\{SK_d|SK_{ai}|SK_{ar}|SK_{ei}|SK_{er}|SK_{pi}|SK_{pr}\} = PRF + (SKEYSEED, N_i|N_r, SPI_i, SPI_r)$$

| Chiave | Descrizione |
|---|---|
| $SK_d$ | Utilizzata per generare il keymaterial per le CHILD_SA |
| $SK_a$ | Chiavi per autenticare gli scambi successivi, una per direzione |
| $SK_e$ | Chiavi per cifrare gli scambi successivi, una per direzione |
| $SK_p$ | Chiavi utilizzata per generare l'AUTH Payload, una per direzione |

TABLE A.1: Chiavi e loro utilizzo

### A.1.2 IPsec SA

Nel caso di una SA questa può essere generata automaticamente dopo l'auth oppure attraverso l'apposito scambio di questo tipo il keymaterial a partire dal quale vengono derivati i parametri di sicurezza è ottenuto nel seguente modo:

$$KEYMAT = prf + (SK_d, N_i|N_r)$$

Nel caso in cui invece si utilizza lo scambio apposito il key material è ottenuto nel seguente modo

## A.2 Security Association Payload

Il Security Association Payload denotato con $SA$ è utlilizzatoper negoziare gli attributi di una Secuiry Association. Dunque può contenere molteplici proposte, le quali devono essere ordinate per preferenza, ogni proposal contiene i seguenti algoritmi crittografici:

- Encryption Algorithm (ENCR)
- Preudorandom Function (PRF)
- Integrity Algorithm (INTEG)
- Diffie-Hellman Group (KE)
- PQ KEM