# 13. RFC 7815：Minimal Internet Key Exchange Version 2 (IKEv2) Initiator Implementation
# RFC 7815：最小限度網際網路金鑰交換(IKEv2) 啟動器實現

網際網路工程任務組（IETF）
Request for Comments: 7815
分類：資訊類
ISSN: 2070-1721

T. Kivinen
INSIDE Secure

2016 年 5 月

## 最小限度網際網路金鑰交換(IKEv2) 啟動器實現

## 摘要

這份文件描述在一個受限的節點上，最小及初始版本的網際網路金鑰交換協定(IKEv2)。IKEv2 歸屬於 IPsec 協定之下，用於身份驗證以及建立與維持安全關聯(Security Associations，SAs)的運作。IKEv2 包含數個可選功能，在最小限度實現的要求中是不需要的。本文描述對最小限度實現的要求，以及多種可被實現的最佳化成果。此處描述的協定可與使用共享密鑰身份驗證的完整 IKEv2 方法相互操作(IKEv2 無需使用身份驗證)。這個最小的啟動器實現只能與扮演回應者的完整 IKEv2 進行溝通。因此，兩個最小的啟動器無法彼此通信。

本文不會更新或修改 RFC 7296 ，但會提供更多此協定中最小版本的緊密描述。如果此文件與 RFC 7296 有相衝突的地方，將以 RFC 7296 作為官方版本參考。

## 本文的狀態

此文件並非一個網際網路標準的規範，只是用於提供訊息為目的。

此文件是網際網路工程任務組(IETF)的一項成品。它代表 IETF 社群的共識。它已經通過公眾審查，並獲得網際網路工程指導小組（IESG）的批准發布。但並不是所有 IESG 批准的文件皆可成為網路標準的候選。可參閱 RFC 5741 第二節。

有關本文當前狀態的資訊、勘誤表，以及如何提供反饋可以前往以下網站：http://www.rfc-editor.org/info/rfc7815。

## 版權聲明

目錄

## 1. 前言

網際網路協定套件越來越多地用於對電源、內存和處理資源有嚴格限制的小型設備上。本文描述最小的 IKEv2 實現，設計用於可與“網際網路密鑰交換協定版本 2（IKEv2）”[RFC7296]相互操作的限制節點。

最小的 IKEv2 實現僅支援協定的發起端。它只支援初始化的 IKE_SA_INIT 和 IKE_AUTH 交換，
不會產生任何其他交換。它也會向所有傳入請求回覆空值（或錯誤）訊息。

這意味著 IKEv2 的大多數可選功能都被省略：NAT 遍歷，IKE SA 密鑰，Child SA 密鑰，多個子 SA，刪除子/ IKE SA，配置負載，可擴充身份驗證協定（EAP）身份驗證，COOKIE 等。

由於受支援的功能集有限，因此可以進行一些優化，此文件不應考慮通用 IKEv2 實現（例如，訊息 ID 可以按指定的方式完成，因為最小的實現只發送 IKE_SA_INIT 和 IKE_AUTH 請求，而不是其他請求）。

本文旨在獨立，這意味著除了加密演算法的描述之外，此處複製實現 IKEv2 所需的所有內容。IKEv2 規範中具有許多背景訊息和基本原理，本文中省略這些訊息。

本文中省略 IANA 註冊管理機構的許多數值；僅列出最小化實施所關注的數值。

本文的主要描述如何在 IKEv2 中使用共享密鑰進行身份驗證，因為它最容易實現。在某些情況下，這還不夠，附錄 B.2 描述如何使用原始公鑰而不是共享密鑰身份驗證。

有關更多訊息,請查看[RFC7296]和[IKEV2IANA]中的完整 IKEv2 規範

本文件中的關鍵詞「必須」、「絕不」、「必須」、「將」、「將不」、「應」、「不應」、「建議」、「可能」、以及「可選」按照[RFC2119]中的描述進行解釋。術語"Constrained Node"在“Terminology for Constrained-Node Networks”[RFC7228]中定義。

## 1.1. 實例

這種最小化實現的一項實例是在進行機器與機器通訊的小型設備中。在這樣的環境中，發起連接的節點可以非常小，而通訊管道的另一端點是某種較大的設備。

小的啟動節點示例可以是遠端開啟車庫門的裝置，即具有打開和關閉車庫門的功能，並通過無線連接到家裡區域網路伺服器，且含有按鈕的裝置。

這種設備的另一個例子是一種感測器設備，例如室溫感測器，其將周期性溫度資料發送到某個中控節點。

這些設備通常會長時間休眠，只會定期運作或因用戶互動而啟動。當設備啟動時，資料傳輸會從休眠的節點開始發送。在發送封包後，可能會有 ACKs 或其他封包在休眠狀態之前返回。如果需要將一些資料從伺服器節點傳送到小型設備，則可以通過輪詢來實現，即小節點週期性地輪詢伺服器以查看它是否具有某些配置變化或相似性。當設備處於休眠狀態時，它將無法維護 IKEv2 SA。也就是說，它會在啟動時再次創建 IKEv2 SA。這意味著不需要對伺服器進行活動檢查，因為在設備再次啟動後，最小的實現將從頭開始。

# 2. 交換

## 2.1. 初始化交換

所有的 IKEv2 通信都包含一個對稱訊息：請求與回應。又稱為“交換”，有時名為“請求/回應”。每項請求都需要一個回應。

對於每組 IKEv2 訊息，發送方負責在超時的情況下重新傳送訊息。回應者絕不能重新發送回應，除非它收到重傳請求。

IKEv2 是一種可靠的協定：發送者必須重新發送請求，直到它收到相對的回應或認為 IKE SA 失敗。來自發送者的重傳請求必須與原始請求位元相同。重傳時間必須呈指數成長。

IKEv2 在 UDP 500 埠上實現。所有 IKEv2 實現必須能夠發送、接收及處理長達 1280 個八位元組的 IKEv2 訊息。即使來源埠口不是 500，IKEv2 實現也必須接受傳入的請求，並且必須回應從中接收請求的位址埠口。

IKEv2 的最小實現僅使用前兩個交換，稱為 IKE_SA_INIT 和 IKE_AUTH。這些用於創建 IKE SA 和第一個 Child SA。除了這些訊息之外，最小的 IKEv2 實現需要理解 CREATE_CHILD_SA 請求，足以生成包含 NO_ADDITIONAL_SAS 錯誤通知的 CREATE_CHILD_SA 回應。它需要理解 INFORMATIONAL 的請求以生成一個空的 INFORMATIONAL 作為回應。沒有需求能夠回應其他請求。

IKE_SA_INIT 交換後的所有訊息都使用 IKE_SA_INIT 交換時的加密演算法和密鑰進行加密保護。

每則 IKEv2 訊息都包含一個訊息 ID 作為其固定標頭的一部分。此訊息 ID 用於匹配請求和回應以及辨識重新傳輸的訊息。

最小實現只需要支援發送者，所以它通常僅發送一個 IKE_SA_INIT 請求，當得到答覆時，後面會產生一個 IKE_AUTH。由於這些訊息具有固定的訊息 ID（0 和 1），因此在此之後不需要追蹤自己的訊息 ID。

最小實現還可以優化傳入請求的訊息 ID 處理，因為它們不需要保護傳入的請求以防止重新傳送。這是可能發生的，因為最小實現只會返回錯誤或空訊息給傳入的請求。這意味著這些傳入請求對最小實現沒有任何影響，因此再次處理它們不會造成任何傷害。也因為最小實現總是可以回覆傳入請求，使用與請求相同的訊息 ID，然後立即忘記請求/回應。這代表著無需追蹤傳入請求的訊息 ID。

在以下描述中，訊息中包含的負載由下面列出的名稱表示。

```
Notation    Payload
----------------------------------------
AUTH        Authentication
CERTREQ     Certificate Request
D           Delete
HDR         IKE header (not a payload)
IDi         Identification - Initiator
IDr         Identification - Responder
KE          Key Exchange
Ni, Nr      Nonce
N           Notify
SA          Security Association
SK          Encrypted and Authenticated
TSi         Traffic Selector - Initiator
TSr         Traffic Selector - Responder
```

初始化交換如下：

```
Initiator                           Responder
-----------------------------------------------------------------------
HDR(SPIi=xxx, SPIr=0, IKE_SA_INIT,
    Flags: Initiator, Message ID=0),
    SAi1, KEi, Ni  -->

                    <--  HDR(SPIi=xxx, SPIr=yyy, IKE_SA_INIT,
                             Flags: Response, Message ID=0),
                             SAr1, KEr, Nr, [CERTREQ]
```

HDR 包含安全參數索引（SPI）、版本號和各種標記。每個端點必需從兩個 SPI 中選擇一個，以便成為 IKE SA 的唯一辨識符。SPI 值為零是特殊的：它表示發送方尚不知道遠程 SPI 值。

傳入的 IKEv2 封包僅使用封包的 SPI 映射到 IKE SA，而不使用封包的來源 IP 位址。

SAi1 負載顯示發起者支援 IKE SA 的加密演算法。 KEi 和 KEr 負載包含 Diffie-Hellman 值，Ni 和 Nr 是 nonce。SAr1 包含來自發起者所選的加密套件。使用共享機密的最小實現，將忽略 CERTREQ 負載。

最小實現很可能只支援一組加密演算法，這意味著 SAi1 負載將是靜態的。它需要檢查收到的 SAr1 是否與它發送的相匹配。

在交談時，每一方都可以生成 SKEYSEED，從中為該 IKE SA 導出所有密鑰。

```
SKEYSEED = prf(Ni | Nr, g^ir)

{SK_d | SK_ai | SK_ar | SK_ei | SK_er | SK_pi | SK_pr }
                = prf+ (SKEYSEED, Ni | Nr | SPIi | SPIr )

prf+ (K,S) = T1 | T2 | T3 | T4 | ...

where:
T1 = prf (K, S | 0x01)
T2 = prf (K, T1 | S | 0x02)
T3 = prf (K, T2 | S | 0x03)
T4 = prf (K, T3 | S | 0x04)
...
```

(表示數量 SK_d、SK_ai、SK_ar、SK_ei、SK_er、SK_pi 和 SK_pr 從 prf +的生成位元開始按順序存取。) g ^ ir 是 Diffie-Hellman 交換中短暫的共享密鑰。 如果需要，g ^ ir 表示為用零填充的大端式八位元組字串，以使其成為模數的長度。 Ni 和 Nr 是 nonce，去除標頭。

SK_d 用於導出子 SA 的新密鑰。 SK_ai 和 SK_ar 使用完整性保護演算法的密鑰，用於驗證後續交換的組件訊息。SK_ei 和 SK_er 用於加密（亦是解密）所有後續交換。 生成 AUTH 負載時使用 SK_pi 和 SK_pr。 SK_d、SK_pi 和 SK_pr 的長度必須是商定之偽隨機函數（PRF）的首選密鑰長度。

為每個方向計算單獨的 SK_e 和 SK_a。用於保護來自原始發送者訊息的密鑰是 SK_ai 和 SK_ei。用於保護另一方向訊息的密鑰是 SK_ar 和 SK_er。符號 SK {...}表示使用該方向的 SK_e 和 SK_a，對這些負載進行加密和完整性保護。

```
Initiator                             Responder
-------------------------------------------------------------------
HDR(SPIi=xxx, SPIr=yyy, IKE_AUTH,
    Flags: Initiator, Message ID=1),
    SK {IDi, AUTH, SAi2, TSi, TSr,
        N(INITIAL_CONTACT)}  -->

                    <--  HDR(SPIi=xxx, SPIr=yyy, IKE_AUTH, Flags:
                                Response, Message ID=1),
                                SK {IDr, AUTH, SAr2, TSi, TSr}
```

發送者使用 IDi 負載聲明其身份，證明對應於 IDi 的密鑰知識，並且完整性使用 AUTH 負載保護第一個訊息的內容。回應者使用 IDr 負載聲明其身份，驗證其身份，並使用 AUTH 負載保護第二個訊息的完整性。

由於最小實現通常只有一個主機連接，這意味著它只有一個共享密鑰。這代表著它不太需要關心 IDr 的負載。如果另一端發送 AUTH 負載，發送者可以使用它擁有的共享密鑰進行驗證，那麼它會知道另一端是配置為與之通信的對等端。

在 IKE_AUTH 請求中，發送者在 TSi 和 TSr 負載中發送 SAi2 負載中的 SA offer(s)和用於子 SA 的建議流量選擇器（TS）。回應者使用 SAr2 負載和選定的流量選擇器回覆接受的 offer。所選擇的流量選擇器可以是發送者建議的子集。

在最小實現中，SA 負載和 TS 負載主要是靜態的。 SA 負載將具有 SPI 值以使用在封裝安全負載（ESP）中，但演算法很可能是唯一且被支援的集合。 發送端的 TS 負載很可能是從任何地方到任何地方，即完整的萬用字元範圍，或從本地 IP 到遠端 IP。在萬用字元情況下，回應者經常將範圍縮小到一個 IP 位址對。 在發送 IKE_AUTH 請求時使用單個 IP 位址對作為流量選擇器將簡化處理，因為回應者將接受 IP 位址對或返回錯誤。如果使用萬用字元範圍，則回應者可能會將流量選擇器範圍縮小到啟動器不可接受的範圍。

IKE_AUTH（和 IKE_SA_INIT）回應可能包含多個狀態通知的負載，最小實現可以忽略這些負載。

還可以有供應商 ID、憑證、憑證請求或配置負載，但任何未知的負載對於最小實現可以簡單地跳過（回應的訊息不能具有關鍵不支援的負載）。

上面的交換包括請求中的 N（INITIAL_CONTACT）通知，因為它通常由最小實現發送。它向另一端指示啟動器在它與回應者之間沒有任何其他 IKE SA，並且如果先前實現中有任何殘存，則回應者可以刪除這些。由於最小實現刪除 IKE SA 而不發送 IKE SA 刪除請求，這將有助於回應者清理殘存狀態。

使用共享密鑰身份驗證時，通過在每個資料區塊上計算訊息身份驗證代碼（MAC）來對用戶群進行身份驗證：

```
對於發送者：
   AUTH = prf( prf(Shared Secret, "Key Pad for IKEv2"),
                     <InitiatorSignedOctets>)
對於回應者：
   AUTH = prf( prf(Shared Secret, "Key Pad for IKEv2"),
                     <ResponderSignedOctets>)
```

字串"Key Pad for IKEv2"是 17 個 ASCII 字符，沒有空終止。實現可以預先計算內部 prf 並僅儲存它的輸出。以上的情形是可能的，因為最小的 IKEv2 實現通常只支援一個 PRF。

在以下計算中，除了固定標頭之外，IDi'和 IDr'是整個 ID 的負載，Ni 和 Nr 只是值，而不是包含它的負載。這裡需注意的是，不發送 nonce Ni / Nr 和 prf（SK_pr，IDr'）/ prf（SK_pi，IDi'）之值。

發送者簽署第一則訊息（IKE_SA_INIT 請求），從標頭中第一個 SPI 的第一個八位元組開始，以第一則訊息中最後一個負載的最後一個八位元組結束。附加於此（用於計算簽署為目的）是回應者的 nonc Nr 和 value prf（SK_pi、IDi'）。

對於回應者，要簽署的八位元組以第二個訊息標頭中第一個 SPI 的第一個八位元組（IKE_SA_INIT 回應）開始，並以該第二個訊息中的最後一個負載的最後一個八位元組結束。附加於此的是發送者的 nonce Ni 和 value prf（SK_pr、IDr'）。

發送者的簽署八位元組可以描述為：

```
InitiatorSignedOctets = RealMessage1 | NonceRData | MACedIDForI
RealIKEHDR =  SPIi | SPIr |  . . . | Length
RealMessage1 = RealIKEHDR | RestOfMessage1
NonceRPayload = PayloadHeader | NonceRData
InitiatorIDPayload = PayloadHeader | RestOfInitIDPayload
RestOfInitIDPayload = IDType | RESERVED | InitIDData
MACedIDForI = prf(SK_pi, RestOfInitIDPayload)
```

回應者的簽署八位元組可以描述為：

```
ResponderSignedOctets = RealMessage2 | NonceIData | MACedIDForR
RealIKEHDR =  SPIi | SPIr |  . . . | Length
RealMessage2 = RealIKEHDR | RestOfMessage2
NonceIPayload = PayloadHeader | NonceIData
ResponderIDPayload = PayloadHeader | RestOfRespIDPayload
RestOfRespIDPayload = IDType | RESERVED | RespIDData
MACedIDForR = prf(SK_pr, RestOfRespIDPayload)
```

需注意 RestOfMessageX 中的所有負載都包含在簽章下，包括本文中未列出的任何負載類型。

啟動器也可能獲得未經身份驗證具有通知負載且包含錯誤代碼的回應。因為該錯誤代碼將是未經身份驗證並且可能是偽造的，所以不需要為此做任何事。最小實現可以快速忽略這些錯誤並重新發送它的請求，直到它超時，但如果發生這種情況，那麼 IKE SA（和子 SA）創建失敗。

回應者還可以使用 IKE_AUTH 回應封包進行回覆，該封包不包含設置子 SA（SAr2，TSi 和 TSr）所需的負載，而是包含 AUTH 負載及錯誤。不支援 CREATE_CHILD_SA 交換的最小實現無法從此方式中恢復。它可以刪除 IKE SA 並重新啟動（如果配置錯誤，可能會再次失敗，如果只是暫時失敗，則可能會成功）。

## 2.2. 其他交換

最小實現必須能夠通過返回一個空的 INFORMATIONAL 回應來回覆 INFORMATIONAL 請求：

```
Minimal implementation            Other end
-------------------------------------------------------------------
                     <--  HDR(SPIi=xxx, SPIr=yyy, INFORMATIONAL,
                                Flags: none,  Message ID=m),
                                SK {...}

HDR(SPIi=xxx, SPIr=yyy, INFORMATIONAL,
    Flags: Initiator | Response,
    Message ID=m),
    SK {}  -->
```

最小實現必須能夠回覆傳入的 CREATE_CHILD_SA 請求。典型實現將通過發送 NO_ADDITIONAL_SAS 錯誤通知來拒絕 CREATE_CHILD_SA 交換：

```
Minimal implementation            Other end
-------------------------------------------------------------------
                     <--  HDR(SPIi=xxx, SPIy=yyy, CREATE_CHILD_SA,
                                Flags: none, Message ID=m),
                                SK {...}

HDR(SPIi=xxx, SPIr=yyy, CREATE_CHILD_SA,
    Flags: Initiator | Response, Message ID=m),
    SK {N(NO_ADDITIONAL_SAS)}  -->
```

需注意 INFORMATIONAL 和 CREATE_CHILD_SA 請求可能包含不受支援的關鍵負載，在這種情況下，兼容實現必須忽略該請求並發送具有 UNSUPPORTED_CRITICAL_PAYLOAD 通知的回應訊息。該通知負載封包含不受支援的關鍵負載的 1 個八位元組負載類型。

## 2.3. 生成鍵值元素

IKE_AUTH 交換創建的 Child SA 密鑰元素生成如下：

KEYMAT = prf+(SK_d, Ni | Nr)

其中 Ni 和 Nr 是來自 IKE_SA_INIT 交換的 nonce。

單個 CHILD_SA 協調可能會導致多個安全關聯。 ESP 和驗證標頭（AH）SAs 成對存在（每個方向一個），因此在單個子 SA 協調中會為它們創建兩個 SA。 每個子 SA 的密鑰元素必須從延伸的 KEYMAT 中獲取來使用以下規則：

- 在 SA 從回應者發送到發送者之前，SA 的所有密鑰都將資料從發送者傳送到回應者。

- 如果 IPsec 協定需要多個密鑰，取自 SA 按順序排列的鍵值元素需要在協定規範中描述。 對於 ESP 和 AH，[IPSECARCH] 定義順序，即：必須從第一位元中獲取加密密鑰（如果有的話），並且必須從剩餘的位元中獲取完整性密鑰（如果有的話）。

每個加密演算法採用固定數量的密鑰元素位元，這些密鑰元素被指定為演算法的一部分或在 SA 負載中協調。

## 3. 一致性要求

對於符合 RFC 7296 的實現規範，必須將其配置為可接受以下內容：

- 包含使用 X.509（PKIX）憑證的公鑰基礎結構並由大小為 1024 或 2048 位元的 RSA 密鑰簽署，其中傳遞的 ID 是 ID_KEY_ID，ID_FQDN，ID_RFC822_ADDR 或 ID_DER_ASN1_DN 中的任何一個。

- 傳遞 ID 的共享密鑰身份驗證是 ID_KEY_ID，ID_FQDN 或 ID_RFC822_ADDR 中的任何一個。

- 使用 PKIX 對回應者進行身份驗證，使用共享密鑰身份驗證對憑證和啟動器進行身份驗證。

本文僅支援第二個描述；它根本不支援 PKIX 憑證。 由於完整的 RFC 7296 回應者還必須支援共享密鑰身份驗證，因此這允許最小實現能夠與符合 RFC 7296 的所有實現進行交互操作。

PKIX 憑證從最小實現中省略，因為這會讓實現增加更多複雜性。 IKEv2 協定中所需的程式碼更動小，但憑證的驗證程式碼比最小 IKEv2 實現本身更複雜。如果出於可伸縮性原因，需要基於公鑰的身份驗證，那麼原始公鑰可能是最好的方案（參見附錄 B.2）。

## 4. 實作狀況

本文描述此篇作者編寫的最小實現。 最小實現支援基礎 IKE_SA_INIT 和 IKE_AUTH 交換，並成功地與完整的 IKEv2 伺服器進行交互操作。這個最小實現在 2011 年 3 月布拉格的互聯智慧對象與網際網路研討會上進行介紹[Kiv11]。這個實現是作為 perl 中的概念證明而編寫的。

還有另一個用 python 編寫的概念驗證實現，它還與一個完整的 IKEv2 伺服器交互操作。

這兩種實現都是為了展示目的而編寫的，包括程式碼中內建的固定配置，並且還將 ESP、ICMP 和 IP 層實現到發送和接收一個 ICMP echo 封包所需的級別。兩種實現都是大約 1000 行程式碼，不包括加密資源庫，但包括 ESP，ICMP 和 IP 層。

## 5. 安全考量

由於這實現與 RFC 7296 相同的協定，意味著它的所有安全注意事項也適用於本文。

## 6. 參考文獻

### 6.1. 規範性參考文獻

[RFC2119] Bradner, S., "Key words for use in RFCs toIndicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296,October 2014, <http://www.rfc-editor.org/info/rfc7296>.

### 6.2. 訊息化參考文獻

[EAI] Yang, A., Steele, S., and N. Freed, "Internationalized Email

Headers", RFC 6532, DOI 10.17487/RFC6532, February 2012, <http://www.rfc-editor.org/info/rfc6532>.

[IDNA] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <http://www.rfc-editor.org/info/rfc5890>.

[IKEV2IANA] IANA, "Internet Key Exchange Version 2(IKEv2) Parameters", <http://www.iana.org/assignments/ikev2-parameters>.

[IPSEARCH] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <http://www.rfc-editor.org/info/rfc4301>.

[Kiv11] Kivinen, T., "Interconnecting Smart Objects with Internet Workshop 2011-03025； IKEv2 and Smart Objects", March2011, <https://www.iab.org/wp-content/IAB-uploads/2011/04/Kivinen.pdf>.

[MODES] National Institute of Standards and Technology, U.S. Department of Commerce, "Recommendation for BlockCipher Modes of Operation", SP 800-38A, 2001.

[PKCS1] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, DOI 10.17487/RFC3447,February 2003, <http://www.rfc-editor.org/info/rfc3447>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate RevocationList (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May2008,

<http://www.rfc-editor.org/info/rfc5280>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <http://www.rfc-editor.org/info/rfc5322>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminologyfor Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <http://www.rfc-editor.org/info/rfc7228>.

[RFC7619] Smyslov, V. and P. Wouters, "The NULL Authentication Method in the Internet Key Exchange Protocol Version2 (IKEv2)", RFC 7619, DOI 10.17487/RFC7619, August2015, <http://www.rfc-editor.org/info/rfc7619>.

[RFC7670] Kivinen, T., Wouters, P., and H. Tschofenig, "Generic Raw Public-Key Support for IKEv2", RFC 7670, DOI 10.17487/RFC7670, January 2016, <http://www.rfc-editor.org/info/rfc7670>.

# 附錄 A.標頭和負載格式

本附錄描述實際的封包負載格式。這是本文所必需的。這些描述大多是從 RFC 7296 複製而來，可以從那裡找到更多訊息。

各種負載包含 RESERVED 欄位，必須將它們歸為零發送，且必須在接收時忽略。

表示整數的所有多個八位元組欄位以大端順序排列(也稱為“最高有效位元組優先”或“網路位元組順序”）。

## A.1. IKE 標頭

每個 IKEv2 訊息都以 IKE 標頭為開頭，在本文中表示為 HDR。標頭之後是一個或多個 IKE 負載，每個負載由前一個負載中的下一個負載欄位識別。通過查看 IKE 標頭中的下一個負載欄位，然

後根據 IKE 負載本身中的下一個負載欄位，直到下一個負載欄位為零表示沒有，按照它們在 IKE 訊息中出現的順序來識別負載。負載如下，如果找到“加密”類型的負載，則解密該負載並將其內容解析為附加負載。加密的負載必須是資料封包中的最後一個負載，加密的負載絕不能包含另一個加密的負載。

IKE 標頭的格式如圖 1 所示。

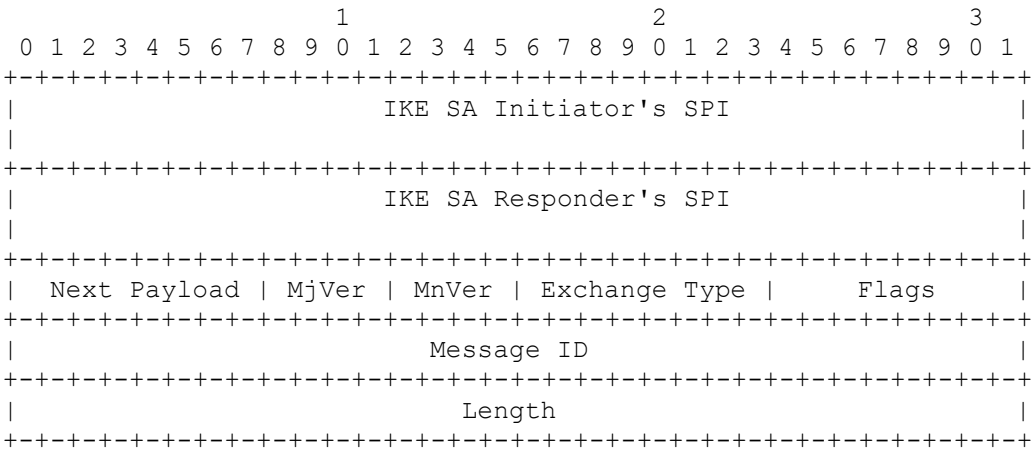


圖 1：IKE 標頭格式

- IKE SA Initiator’s SPI (啟動器的 SPI)（8 個八位元組） - 啟動器選擇的值，用於識別唯一的 IKE 安全關聯。該值不得為零。

- IKE SA Responder’s SPI(回應者的 SPI)（8 個八位元組） - 回應者選擇的值，用於識別唯一的 IKE 安全關聯。 在 IKE 初始交換的第一則消息中，該值必須為零。

- Next Payload(下一個負載)（1 個八位元組）- 表示緊跟在標頭之後的負載類型。每個負載的格式和值定義如下。

- MjVer(主要版本)（4 位元） - 表示正在使用的 IKE 協定主要版本。 基於此版本的 IKE 的實現必須將主要版本設置為 2，且必須刪除具有更高主版本號的訊息。

- MnVer(次要版本)（4 位元） - 表示正在使用的 IKE 協定次要版本。 基於此版本的 IKE 的實現必須將次要版本設置為零。他們必須忽略收到的訊息次要版本號。

- Exchange Type(交換類型)（1 個八位元組）- 表示正在使用的交換類型。這約束交換中每則訊息中發送的負載。

| Exchange Type | Value |
|---|---|
| IKE_SA_INIT | 34 |
| IKE_AUTH | 35 |
| CREATE_CHILD_SA | 36 |
| INFORMATIONAL | 37 |

- Flags(標誌)（1 個八位元組）- 表示為訊息設置的特定選項。選項的存在由正在設置的標誌欄位中的相應位元指示。 這些位元如下：

```
+-+-+-+-+-+-+-+-+
|X|X|R|V|I|X|X|X|
+-+-+-+-+-+-+-+-+
```

在下面的描述中，“設置”位元表示其值為“1”，而“清除”表示其值為“0”。 發送時必須清除'X'位，收到時必須忽略。

R（回應）- 該位元表示此訊息是對包含相同訊息 ID 的訊息回應。 必須在所有請求消息中清除該位元，且必須在所有回應中設置該位元。 IKEv2 端點絕不能生成對標記為回應的訊息回應。

V（版本）- 該位元表示發送器能夠產生比主要版本欄位中指示協定更高的主要版本號。IKEv2 的實現必須在發送時清除該位元，且必須在傳入訊息中忽略它。

I（啟動器）- 必須在 IKE SA 的原始發送者發送的訊息中設置該位元，並且必須在原始回應者發送的訊息中清除。接收方使用它來確定接收方生成 SPI 的是哪 8 個八位元組。該位元發生變化以反映誰發起 IKE SA 的最後一次重定密鑰。

Message ID(訊息 ID)（4 個八位元組，無號整數）- 用於控制丟失封包重傳以及請求和回應匹配的訊息識別符。它對協定的安全性至關重要，因為它用於防止訊息重播攻擊。

Length(長度)（4 個八位元組，無號整數）- 八位元組中總訊息的長度（標頭+負載）。

## A.2. 通用負載標頭

每個 IKE 負載以通用負載標頭開始，如圖 2 所示。下面每個負載的數字將包括通用負載標頭，但為簡潔起見，將省略每個欄位的描述。

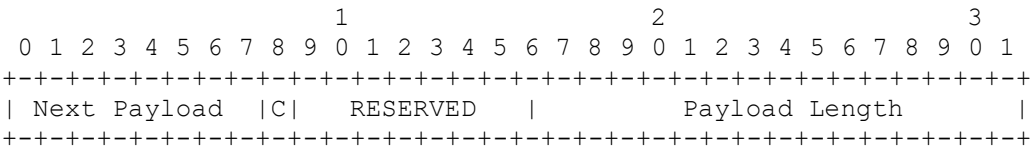


圖 2：通用負載標頭

通用負載標頭欄位定義如下：

- Next Payload(下一個負載)（1 個八位元組）- 訊息中下一個負載的負載類型識別符。如果當前負載是訊息中的最後一個，則該欄位將為零。 該欄位提供“鏈接”能力，通過將每個負載附加到訊息的末尾並設置前一個負載的下一個負載欄位以指示新的負載的類型，可以將附加的負載添加到訊息中。 加密的負載必須始終是訊息的最後一個負載，這是一個例外。 它包含附加負載格式的資料結構。 在加密的負載標頭中，下一個負載欄位設置為第一個包含的負載的負載類型（而不是零）；相反地，最後包含的負載的下一個負載欄位設置為零）。 此處列出最小實現所需的負載類型值。

| Next Payload Type | Notation | Value |
|---|---|---|
| No Next Payload | | 0 |
| Security Association | SA | 33 |
| Key Exchange | KE | 34 |
| Identification - Initiator | IDi | 35 |
| Identification - Responder | IDr | 36 |
| Certificate | CERT | 37 |
| Certificate Request | CERTREQ | 38 |
| Authentication | AUTH | 39 |
| Nonce | Ni, Nr | 40 |
| Notify | N | 41 |
| Delete | D | 42 |
| Traffic Selector - Initiator | TSi | 44 |
| Traffic Selector - Responder | TSr | 45 |
| Encrypted and Authenticated | SK | 46 |

- Critical（1 位元）- 如果發送者希望接收者在前一個負載的下一個負載欄位中不理解負載類型程式碼時希望跳過此負載，則必須設置為零。 如果發送者希望接收者拒絕整個訊息，且它也不理解負載類型，則必須設置為 1。如果發送者理解負載類型程式碼，接收者必須忽略它。 對於本文中定義的負載類型，必須設置為零。需注意關鍵位元適用於當前負載而不是類型程式碼出現在第一個八位元組中的“下一個”負載。

- Payload Length(負載長度)（2 個八位元組，無號整數）- 當前負載的八位元組長度，包括通用負載標頭。

## A.3. 安全關聯負載

安全關聯負載（在本文中表示為 SA）是用於協商安全關聯的屬性。

SA 負載由一個或多個提案組成。每個提案都包含一個協定。每個協定都包含一個或多個轉換 - 每個轉換都指定一個加密演算法。每個轉換包含零個或多個屬性（僅當轉換 ID 未完全指定加密演算法時才需要屬性;當前唯一的屬性是可變長度密碼的“密鑰長度”屬性，這意味著確實存在零個或一個屬性）。

回應者必須選擇單個套件，該套件可以是遵循以下規則 SA 提案的任何子集。

每個提案都包含一個協定。 如果提案被接受，SA 回應必須包含相同的協定。每個 IPsec 協定提案都包含一個或多個轉換。 每個轉換都包含一個轉換類型。公認的加密套件必須包含提案中包含的每種類型的一個轉換。例如：如果 ESP 提案包括轉換 ENCR_3DES、ENCR_AES w / keysize 128、ENCR_AES w / keysize 256、AUTH_HMAC_MD5 和 AUTH_HMAC_SHA，則接受的套件必須包含 ENCR_transforms 之一和 AUTH_transforms 之一。 因此，以上六種組合是可接受的。

最小實現可以創建非常簡單的 SA 提案，即包括一個提案，其對於每個轉換類型僅包含一個轉換。 在提案中僅包含一個 Diffie-Hellman 組非常重要，因此無需在回應中執行 INVALID_KE_PAYLOAD 處理。

解析 SA 時，實現必須檢查總負載長度是否與負載的內部長度和計數一致。Proposals、Transforms 和 Attributes 都有自己的可變長度編碼。它們被嵌套，使得 SA 的負載長度包括 SA，Proposals、Transforms 和 Attributes 訊息的組合內容。 Proposals 的長度包括它包含的所有變換和屬性的長度。Transforms 的長度包括它包含的所有屬性的長度。

每個提案/ 協定結構後跟隨一個或多個轉換結構。不同變換的數量通常由協定確定。AH 通常具有兩種變換：擴充序列號（ESN）和完整性檢查演算法。

ESP 通常有三種：ESN，加密演算法和完整性檢查演算法。IKEv2 通常具有四種變換：Diffie-Hellman 組，完整性檢查演算法，PRF 算法和加密演算法。對於每個協定，為允許變換集分配轉換 ID 號，它們出現在每個變換的標頭中。

如果有多個具有相同轉換類型的轉換，則提案是這些轉換的 OR。如果存在具有不同轉換類型的多個轉換，則提案是不同組的 AND。

給定的轉換可以有一個或多個屬性。當轉換可以以多種方式使用時，屬性是必需的，如加密演算法具有可變密鑰大小。轉換將指定演算法，屬性將指定密鑰大小。要為屬性建議代替值（例如，AES 加密演算法的多個密鑰大小），實現必須包含具有相同轉換類型的多個轉換，每個轉換具有單個屬性。

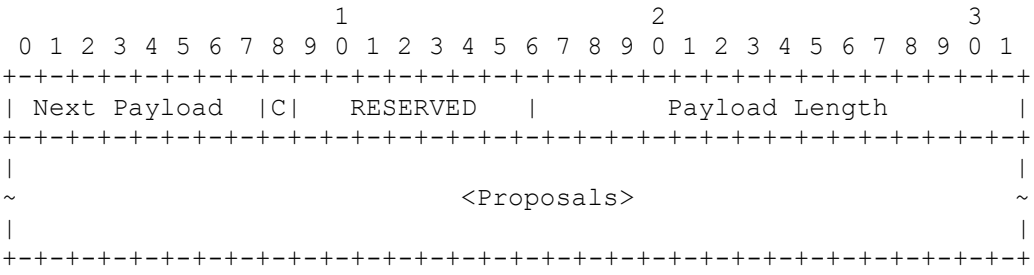


圖 3:安全關聯負載

- Proposals(提案（變數）) - 一個或多個提案子結構。

### A.3.1 建議結構

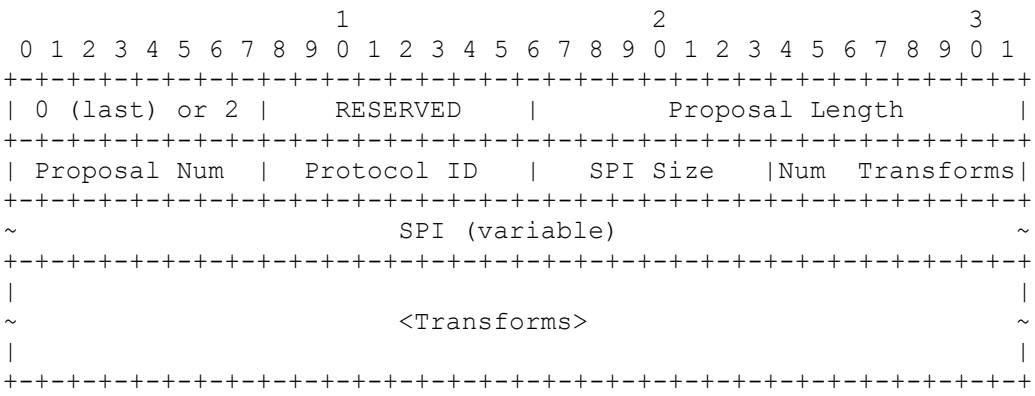


圖 4：建議結構

- 0（最後）或 2（更多）（1 個八位元組）- 指定是否為 SA 的最後一個提案子結構。

- Proposal Length(提案長度)（2 個八位元組，無號整數）- 此提案的長度，包括後面的所有轉換和屬性。

- Proposal Num(提案號)（1 個八位元組）- 提出提案時，SA 負載中的第一個提案必須為 1，後續提案必須比前一提案多一個。當提案被接受時，SA 負載中的提案編號必須與已接受的提案編號相匹配。

- Protocol ID(協定 ID)（1 個八位元組）- 指定當前協商的 IPsec 協定識別符。

| Protocol | Protocol ID |
|---|---|
| IKE | 1 |
| AH | 2 |
| ESP | 3 |

- SPI Size(SPI 大小)（1 個八位元組）- 對於初始 IKE SA 協商，該欄位必須為零； SPI 從外部標頭獲得。在後續協商期間，它等於相應協定的 SPI 大小（以八位元組為單位）（IKE 為 8，ESP 和 AH 為 4）。

- Num Transforms（1 個八位元組）- 指定此提案中的轉換數。

- SPI（變量）- 發送實體的 SPI。 當 SPI 大小欄位為零時，此欄位不存在於安全關聯有效內容中。

- 轉換（變量）- 一個或多個轉換子結構。

### A.3.2 轉換結構

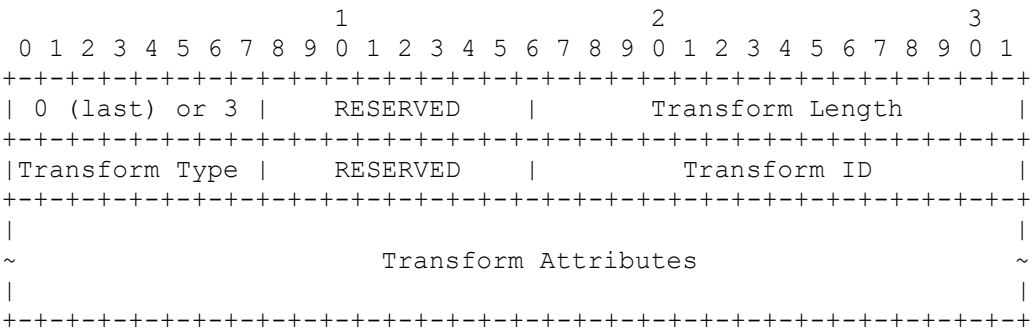


圖 5：轉換結構

- 0（最後）或 3（更多）（1 個八位元組）- 指定這在提案中是否為最後一個轉換子結構。

- Transform Length(轉換長度) - 轉換子結構的長度（以八位元組為單位），包括標頭和屬性。

- Transform Type(轉換類型)（1 個八位元組）- 在此轉換中指定的轉換類型。不同協定支援不同的轉換類型。對於某些協定，某些轉換可能是可選的。 如果轉換是可選的並且發送者希望省略轉換，則提案中不包括給定類型的轉換。如果發送者希望對回應者使用轉換可選，則它包括轉換子結構，其中轉換 ID = 0 作為選項之一。

- Transform Type(轉換 ID)（2 個八位元組）- 建議的轉換類型的特定實例。

下面列出相關的轉換類型值。有關更多訊息，請參閱[RFC7296]。

| Description | Trans. Type | Used In |
|---|---|---|
| Encryption Algorithm (ENCR) | 1 | IKE and ESP |
| Pseudorandom Function (PRF) | 2 | IKE |
| Integrity Algorithm (INTEG) | 3 | IKE, AH, optional in ESP |
| Diffie-Hellman group (D-H) | 4 | IKE, optional in AH & ESP |
| Extended Sequence Numbers (ESN) | 5 | AH and ESP |

對於轉換類型 1（加密演算法），下面列出相關的轉換 ID。

| Name | Number |
|---|---|
| ENCR_AES_CBC | 12 |
| ENCR_AES-CCM_8 | 14 |

對於轉換類型 2（偽隨機函數），下面列出相關的轉換 ID。

| Name | Number |
|---|---|
| PRF_HMAC_SHA1 | 2 |

對於轉換類型 3（完整性演算法），下面列出相關的轉換 ID。

| Name | Number |
|---|---|
| AUTH_HMAC_SHA1_96 | 2 |
| AUTH_AES_XCBC_96 | 5 |

對於轉換類型 4（Diffie-Hellman 組），下面列出相關的轉換 ID。

| Name | Number |
|---|---|
| 1536-bit MODP | 5 |
| 2048-bit MODP | 14 |

對於轉換類型 5（擴充序列號），下面列出相關的轉換 ID。

| Name | Number |
|---|---|
| No Extended Sequence Numbers | 0 |
| Extended Sequence Numbers | 1 |

請注意，支援 ESN 的發送者通常會在其提案中包含兩個 ESN 轉換，值為“0”和“1”。
包含值為“1”的單個 ESN 轉換的提案意味著使用正常（非擴充）序列號是不可接受的。

### A.3.3 按協定的有效轉換類型

伴隨 SA 負載轉換的數量和類型取決於 SA 本身的協定。建立 SA 的負載具有以下強制和可選的轉換類型。一個兼容的實現必須理解它支援的每個協定中所有強制和可選類型（儘管它不需要接受具有不可接受的套件提案）。如果提案的唯一值是 NONE，則提案可以省略可選類型。

| Protocol | Mandatory Types | Optional Types |
|---|---|---|
| IKE | ENCR, PRF, INTEG, D-H | |
| ESP | ENCR, ESN | INTEG, D-H |
| AH | INTEG, ESN | D-H |

### A.3.4 轉換屬性

轉換類型 1（加密算法）轉換可能包括一個轉換屬性：密鑰長度。

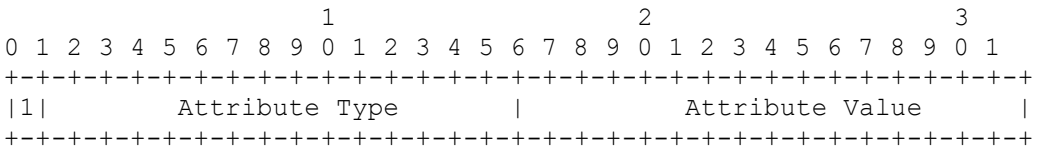


圖 6：資料屬性

- Attribute Type(屬性類型)（15 位元）- 每種屬性的唯一識別符（見下文）。
- Attribute Value(屬性值) - 與屬性類型關聯的屬性的值。

| Attribute Type | Value |
|---|---|
| Key Length (in bits) | 14 |

鍵值長度屬性指定某些轉換的密鑰長度（必須使用網路字元順序），如下所示：

- 密鑰長度屬性不得與使用固定長度密鑰的轉換一起使用。
- 某些轉換指定必須始終包含鍵值長度屬性。 例如，ENCR_AES_CBC。

## A.4. 密鑰交換負載

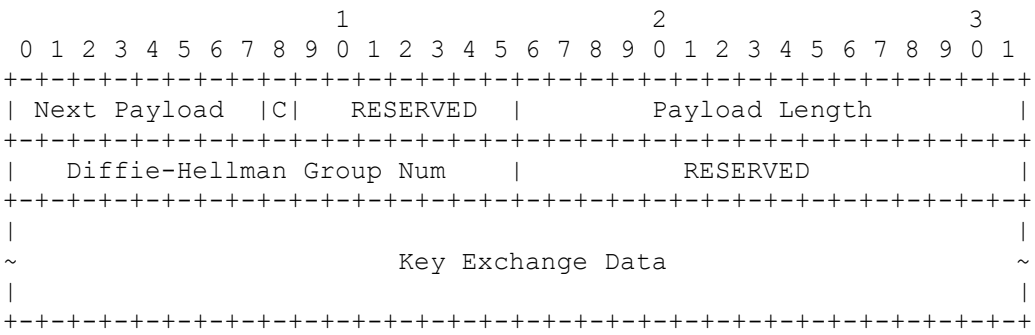


圖 7：密鑰交換負載格式

密鑰交換負載是通過將一個 Diffie-Hellman 公開值複製到負載的“密鑰交換資料”部分來建構的。模冪運算組（MODP）的 Diffie-Hellman 公開值長度必須等於執行求冪的基本模數長度，必要時在該值之前加零。

Diffie-Hellman Group Num 識別計算密鑰交換資料的 Diffie-Hellman 組。該 Diffie-Hellman Group Num 必須匹配在同一訊息裡發送的 SA 負載的提議中指定之 Diffie-Hellman 組。

## A.5. 識別負載

識別負載(在本文中表示為IDi和IDr)允許對等方彼此聲明身份。在 IDi / IDr 負載中使用 ID_IPV4_ADDR / ID_IPV6_ADDR 識別類型時，IKEv2 不會求此位址匹配 IKEv2 資料封包的 IP 頭中位址或 TSi / TSr 有效載負中任何內容。IDi / IDr 的內容純粹用於獲取與另一方相關的策略和驗證資料。在最小實現中，始終使用 KEY_ID 類型可能是最簡單的。 這允許 ID 負載是靜態的。在動態分配 IP 位址的環境中使用 IP 位址存在其問題。

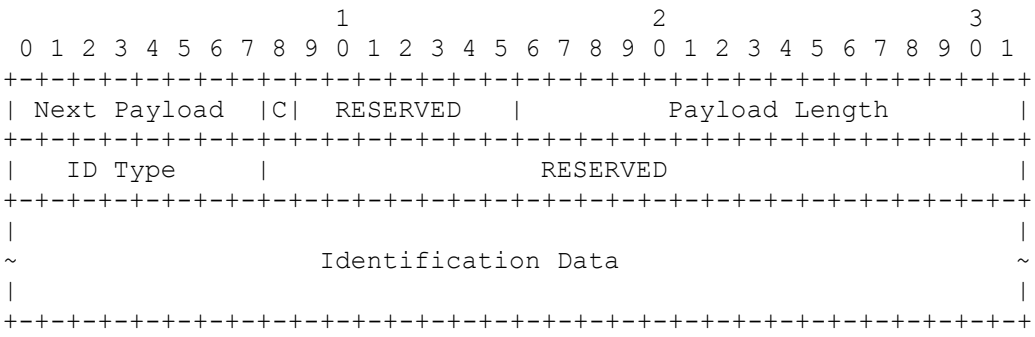


圖 8：識別負載格式

- ID Type(ID 類型)（1 個八位元組）- 指定正在使用的識別類型

- Identification Data(識別資料)（可變長度）- 值，由識別類型指示。 識別資料的長度根據 ID 負載標頭中的大小計算。

下表列出“識別類型”欄位已分配的語義。

| ID Type | Value |
|---|---|
| ID_IPV4_ADDR<br>單一四（4）個八位元組 IPv4 位址。 | 1 |
| ID_FQDN<br>完全限定的域名字串。ID_FQDN 的一個範例是“example.com”。字串絕不能包含任何終止符（例如，NULL、CR 等）。 ID_FQDN 中的所有字串都是 ASCII 碼；對於“國際化域名”，語法如[IDNA]中所定義，例如“xn--tmonesimerkki-bfbb.example.net”。 | 2 |
| ID_RFC822_ADDR<br>基於[RFC5322]完全限定的 RFC 822 電子郵件字串。 ID_RFC822_ADDR 的一個範例是“jsmith@example.com”。字串絕不能包含任何終結符。 由於[EAI]，而明智地將此欄位視為 UTF-8 編碼的文本而不是純 ASCII。 | 3 |
| ID_IPV6_ADDR<br>單一十六（16）個八位元組 IPv6 位址。 | 5 |
| ID_KEY_ID<br>不透明的八位元組串流，可用於傳遞進行某些專有類型識別所必需的供應商特定訊息。最小實現可能使用此類型為設備發送序列號或類似設備的唯一靜態識別資料。 | 11 |

## A.6. 憑證負載

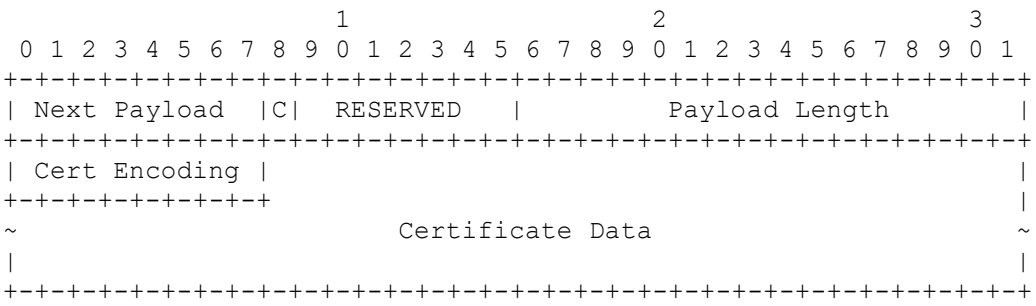


圖 9：憑證負載格式

- Certificate Encoding(憑證編碼)（1 個八位元組）- 該欄位表示憑證資料欄位中包含的憑證或憑證相關訊息的類型。

| Certificate Encoding | Value |
|---|---|
| X.509 Certificate - Signature | 4 |
| Raw Public Key | 15 |

- Certificate Data(憑證資料)（可變長度）- 憑證資料的實際編碼。 憑證類型由憑證編碼欄位指示。

上述類型的語法是：

- ”X.509 憑證 - 簽署”包含 DER 編碼的 X.509 憑證，其公鑰用於驗證發送者的辨識負載。需注意使用此編碼時，如果需要發送憑證鏈，則使用多個 CERT 負載，只有第一個持有用於驗證發送方的辨識負載的公鑰。

- 憑證負載包含 PKIX 憑證的 Subject Public Key Info 部分（參見[RFC5280]的第 4.1.2.7 節）。這是一個非常簡單的 ASN.1 對象，它在實際公鑰值之前主要包含靜態部分。 有關更多訊息，請參見[RFC7670]。

## A.7. 憑證請求負載

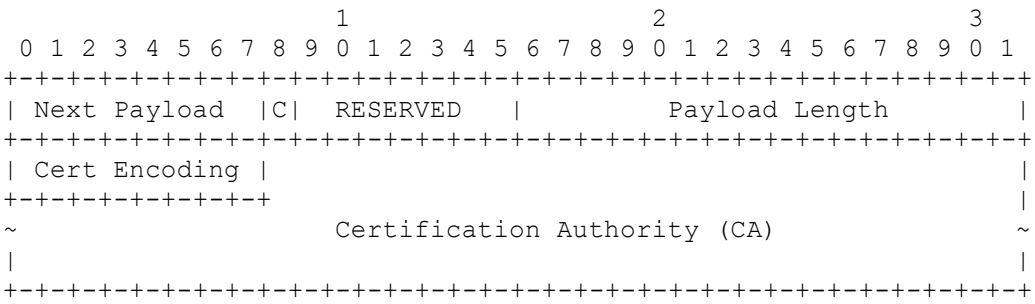


圖 10：憑證請求負載格式

- Certificate Encoding(憑證編碼)（1 個八位元組）- 包含所請求憑證的類型或格式的編碼。

- Certification Authority(數位憑證認證機構)（可變長度）- 包含所請求憑證類型的可接受數位憑證認證機構之編碼。

- 憑證編碼欄位具有與憑證負載定義值的相同值。“數位憑證認證機構”欄位包含此憑證類型受信任機構的指示符。數位憑證認證機構值是受信任的數位憑證認證機構之公鑰 SHA-1 雜湊串聯列表。每個都從信任鏈憑證中被編碼為 Subject Public Key Info 元素的 SHA-1 雜湊（參見[RFC5280]的第 4.1.2.7 節）。20 個八位元組的雜湊值連接在一起，沒有其他格式。

## A.8. 身份驗證負載

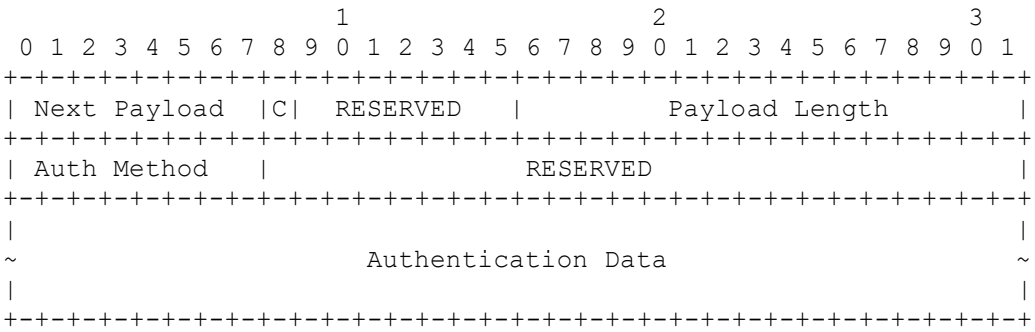


圖 11：身份憑證負載格式

- Auth Method(身份驗證方法)（1 個八位元組）- 指定使用的身份驗證方法。

| Mechanism | Value |
|---|---|
| RSA Digital Signature | 1 |

使用 RSA 私鑰和[PKCS1]中指定的 RSASSA-PKCS1-v1_5 簽署方案；有關詳細訊息，請參見[RFC7296]的第 2.15 節。

| Shared Key Message Integrity Code | 2 |
|---|---|

使用與 ID 負載中的識別和協商的 PRF 關聯共享密鑰，按照前面的規定進行計算。

- Authentication Data(身份驗證資料)（可變長度）- 參見第 2.1 節。

## A.9. Nonce 負載

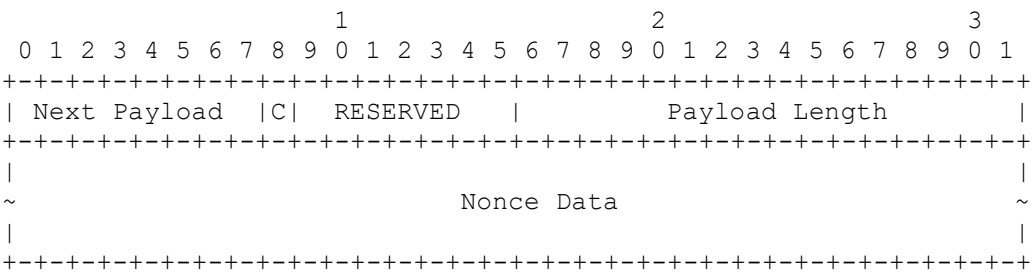


圖 12：Nonce 負載格式

- Nonce Data(Nonce 資料)（可變長度）- 包含發送實體生成的 Nonce 資料。Nonce 資料的大小必須在 16 到 256 個八位元組之間。 Nonce 值絕不能重覆使用。

## A.10. 通知負載

通知負載（在本文中表示為 N）用於將訊息資料（例如錯誤條件和狀態轉換）傳輸到 IKE 對等方。通知負載可能出現在回應訊息中（通常指定拒絕請求的原因），在 INFORMATIONAL 交換中（回報不在 IKE 請求中的錯誤），或在其他訊息中出現以指示發送方功能或修改其含義之請求。

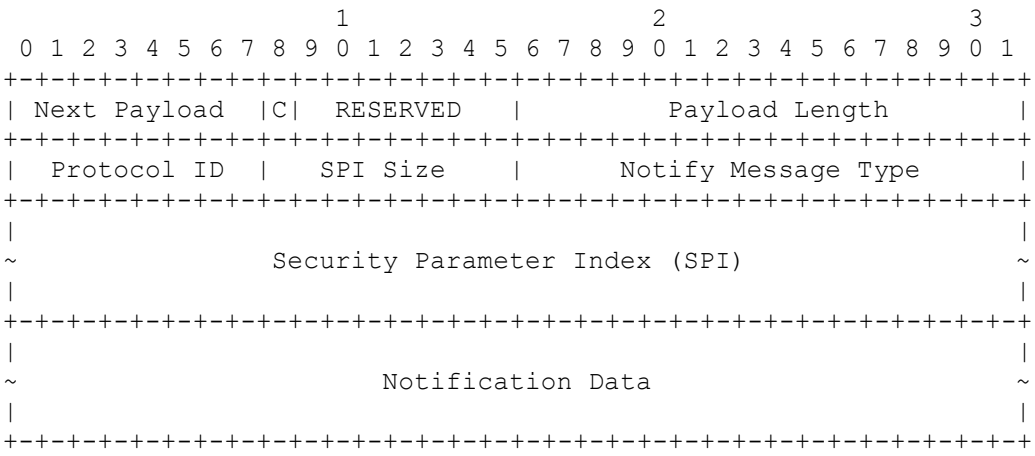


圖 13：通知負載格式

- Protocol ID(協定 ID)（1 個八位元組）- 如果此通知涉及 SPI 在 SPI 欄位中現存的 SA，則此欄位指示為該 SA 的類型。如果 SPI 欄位為空，則該欄位必需發送為零，且必須在接收時忽略。

- SPI Size(SPI 大小)（1 個八位元組）- SPI 中八位元組長度由 IPsec 協定 ID 定義，如果沒有 SPI 適用，則為零。有關 IKE SA 的通知，SPI 大小必須為零，SPI 欄位必須為空。

- Notify Message Type(通知訊息類型)（2 個八位元組）- 指定通知訊息的類型。

- SPI（可變長度）- 安全參數索引。

- Notification Data(通知資料)（可變長度）- 除通知訊息類型外傳輸狀態或錯誤資料。 此欄位的值是特定於類型的。

### A.10.1 通知訊息類型

通知訊息可以是指出無法建立 SA 的錯誤消息。它也可以是管理 SA 資料庫與對等方進行通信的狀態資料。

範圍為 0 至 16383 的類型用於回報錯誤。接收帶有其中一種類型的通知負載之實現，它在回應中無法識別，必須假定相應的請求完全失敗。請求中無法識別的錯誤類型以及請求或回應中的狀態類型必須被忽略，並且應記錄它們。

狀態類型的通知負載可以添加到任何訊息中，如果不能識別則必須忽略。它們旨在指示功能，並且作為 SA 協商的一部分，用於協商非加密參數。

| NOTIFY messages: error types | Value |
|---|---|
| UNSUPPORTED_CRITICAL_PAYLOAD | 1 |

表示通知資料欄位中包含的 1 個八位元組負載類型是未知的。

| | |
|---|---|
| INVALID_SYNTAX | 7 |

表示接收到的 IKE 訊息無效，因為某些類型，長度或值超出範圍或因為策略原因拒絕請求。為避免使用偽造訊息進行阻斷服務（DoS）攻擊，如果訊息 ID 和加密校驗有效，則只能在加密資料封包中返回此狀態。 為了避免將訊息洩露給探測節點的人，必須發送此狀態以回應其他狀態類型中未涵蓋的錯誤。為了幫助除錯，應將更詳細的錯誤訊息寫入控制台或記錄。

| | |
|---|---|
| NO_PROPOSAL_CHOSEN | 14 |

所提出的加密套件都不可接受。這可以在任何情況下發送，其中提供的提案對於回應者是不可接受的。

| | |
|---|---|
| NO_ADDITIONAL_SAS | 35 |

指定節點不再接受任何子 SA。

| NOTIFY messages: status types | Value |
|---|---|
| INITIAL_CONTACT | 16384 |

聲明此 IKE SA 是身份驗證之間當前唯一活躍的 IKE SA。

## A.11. 流量選擇器負載

流量選擇器（TS）負載允許端點將其安全策略資料庫（SPD）中的一些訊息傳遞給其對等端。流量選擇器負載指定將通過新設置的 SA 轉發的封包選擇標準。

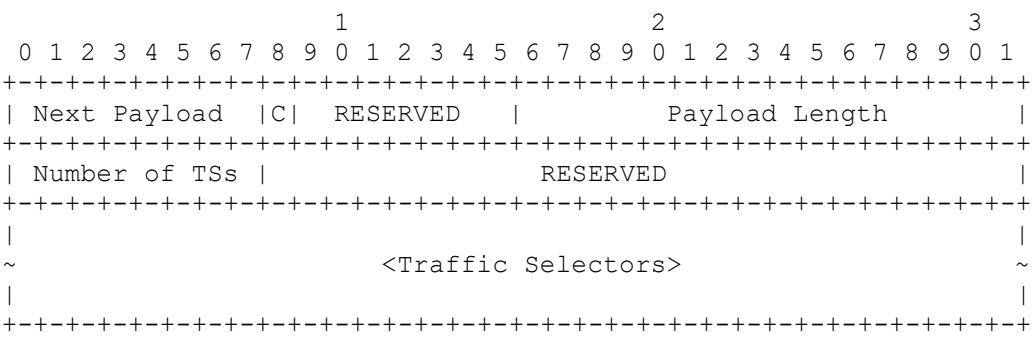


圖 14：流量選擇器負載格式

- Number of TSs(流量選擇器數量)（1 個八位元組）- 提供的流量選擇器數量。

- Traffic Selectors(流量選擇器)（可變長度）- 一個或多個單獨的流量選擇器。

流量選擇器負載的長度包括流量選擇器標頭和所有流量選擇器。

不要求 TSi 和 TSr 包含相同數量的單個流量選擇器。 因此，它們被解釋如下：如果匹配 TSi 中含有至少一個單獨選擇器及 TSr 中含有至少一個單獨選擇器，則分組匹配給定 TSi / TSr。

交換中的每則訊息都會出現兩個流量選擇器負載，這些訊息會創建一個子 SA 對。每個流量選擇器負載包含一個或多個流量選擇器。每個流量選擇器由 IP 位址（IPv4 或 IPv6）、埠口和 IP 協定 ID 組成。

兩個流量選擇器負載中的第一個被稱為 TSi（流量選擇器 - 發送者）。第二種稱為 TSr（流量選擇器 - 回應者）。 TSi 指定從子 SA 對發送方轉發的流量來源位址（或轉發流量的目的地位址）。TSr 指定轉發到 Child SA 對回應者的流量目的地位址（或從其轉發的流量來源位址）。

IKEv2 允許回應者選擇發送者提案的流量子集。

當回應者選擇發送者提案的流量子集時，它將流量選擇器縮小到發送者提議的某個子集（假設該集合不會成為空集）。如果建議的流量選擇器類型未知，則回應者會忽略該流量選擇器，以便在縮小的集合中不返回未知類型。

為了使回應者能夠選擇合適的範圍，如果發送者由於封包而請求 SA，則發送者應該在每個 TSi 和 TSr 中包含作為第一個流量選擇器之特定的流量選擇器，包括觸發封包的位址。如果發送者創建子 SA 對而不是回應到達的封包，在啟動時，則發送者可能沒有特定位址，初始通道優先於任何其他位址。 在這種情況下，TSi 和 TSr 中的第一個值可以是範圍而不是特定值。

由於最小實現可能僅支援一個 SA，因此流量選擇器通常從發送者的 IP 位址到回應者的 IP 位址（即，沒有埠口或協定選擇器且僅一個範圍）。

### A.11.1 流量選擇器

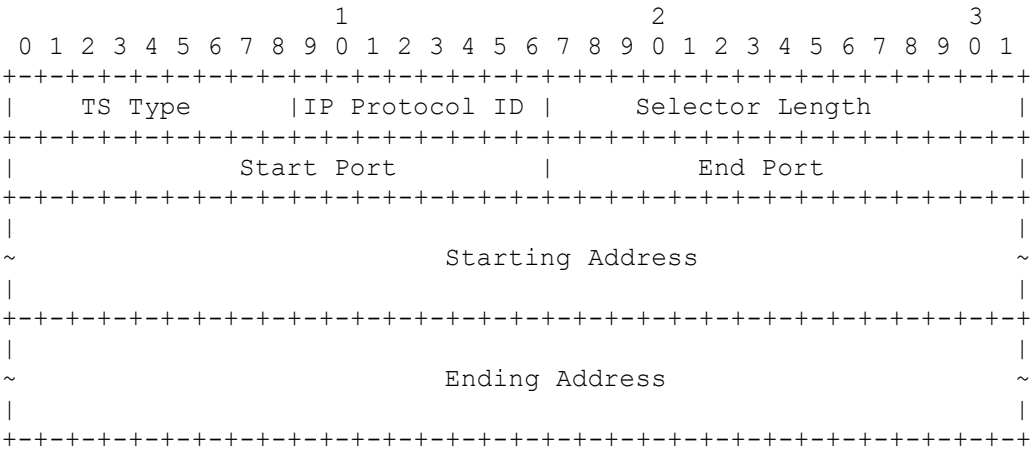


圖 15：流量選擇器

- TS Type(流量選擇器類型)（1 個八位元組）- 指定流量選擇器的類型。
- IP Protocol ID(IP 協定 ID)（1 個八位元組）- 指定關聯的 IP 協定 ID（例如 UDP，TCP 和 ICMP）的值。 值為零表示協定 ID 與此流量選擇器無關 - SA 可以承載所有協定。
- Selector Length(選擇器長度) - 指定此流量選擇器子結構的長度，包括標頭。
- Start Port(啟始埠口)（2 個八位元組，無號整數）- 指定此流量選擇器允許的最小埠口號值。對於未定義埠口的協定（包括協定 0），或者如果允許所有埠口，則此欄位必須為零。
- End Port(結束埠口)（2 個八位元組，無號整數）- 指定此流量選擇器允許的最大埠口號值。對於未定義埠口的協定（包括協定 0），或者如果允許所有端口，則此欄位必須為 65535。

- Starting Address(起始位址) - 此流量選擇器中包含的最小位址（長度由流量選擇器類型確定）。

- End Address(結束位址) - 此流量選擇器中包含的最大位址（長度由流量選擇器類型確定）。

下表列出“流量選擇器類型”欄位和相應的“位址選擇器資料”之值。

| TS Type | Value |
|---|---|
| TS_IPV4_ADDR_RANGE | 7 |
| 一系列 IPv4 位址，由兩組 4 個八位元組值表示。第一個值是起始 IPv4 位址（包括），第二個值是結束 IPv4 位址（包括）。 落在兩個指定位址之間的所有位址都被視為在列表中。 | |
| TS_IPV6_ADDR_RANGE | 8 |
| 一系列 IPv6 位址，由兩組 16 個八位元組值表示。第一個值是起始 IPv6 位址（包括），第二個值是結束 IPv6 位址（包括）。 落在兩個指定位址之間的所有位址都被視為在列表中。 | |

## A.12. 加密負載

加密的負載，在本文中表示為 SK {...}，包含加密形式的其他負載。

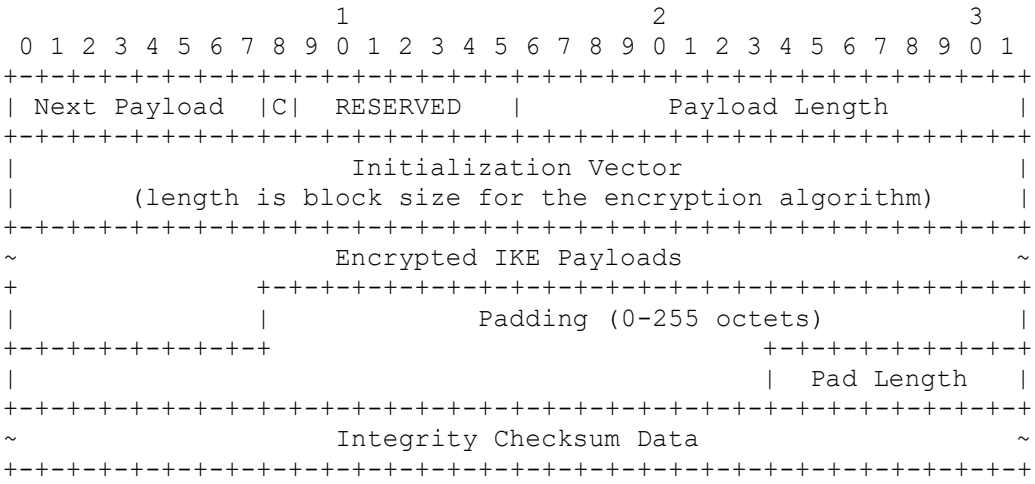


圖 16：加密負載格式

- Next Payload(下一個負載) - 第一個嵌入式負載的負載類型。需注意這是標準標頭格式中的一個例外，因為加密的負載是

訊息中的最後一個負載； 因此，下一個負載欄位通常為零。但是因為這個負載的內容是嵌入式負載，並且沒有自然的位置來放置第一個類型，所以這種類型放在這裡。

- Payload Length(負載長度) - 包括標頭長度、初始化向量（IV）、加密 IKE 負載、填充、填充長度和完整性校驗資料。

- Initialization Vector(初始化向量) - 對於密碼區塊鏈（CBC）模式密碼，初始化向量（IV）的長度等於基礎加密演算法區塊長度。發送者必須為每則訊息選擇一個新的不可預測 IV； 接收者必須接受任何值。鼓勵讀者諮詢[MODES]以獲取有關 IV 生成的建議。特別是，使用先前訊息的最終密文區塊不被認為是不可預測的。 對於 CBC 以外的模式，在指定加密演算法和模式的文件中指定 IV 格式及處理。

- IKE 負載(Encrypted IKE Payloads)如本節前面所述。該欄位使用協商密碼加密。

- 填充(Padding)可以包含發送方選擇的任何值，並且必須具有使負載、填充和填充長度組合為加密區塊大小的倍數長度。該欄位使用協商密碼加密。

- 填充長度(Pad Length)是填充欄位的長度。 發送方應該將填充長度設置為最小值，使得負載、填充和填充長度組合為區塊大小的倍數，但是接收方必須接受適當對齊長度。該欄位使用協商密碼加密。

- 完整性校驗資料(Integrity Checksum Data)是整個訊息從固定 IKE 標頭開始通過填充長度的加密校驗。校驗必須通過加密訊息計算。 其長度由協商的完整性演算法確定。

## 附錄 B.有效可選功能

IKEv2 有一些可選功能，在某些情況下可能對最小實現很有用。這些功能包括原始公鑰認證和發送 IKE SA 刪除通知。

### B.1.IKE SA 刪除通知

在某些情況下，最小實現設備會創建 IKE SA，發送一個或些許封包，可能會返回一些封包，然後設備重新進入休眠狀態，忘記 IKE SA。 在這種情況下，最小實現發送 IKE SA 刪除通知以告知另一端 IKE SA 正在消失，因此它可以釋放資源。

刪除 IKE SA 可以透過發送一個具有固定訊息 ID 且在加密負載內只有一個負載的資料封包來完成。另一端將傳回一個空回應：

```
Initiator                           Responder
-------------------------------------------------------------------
HDR(SPIi=xxx, SPIr=yyy, INFORMATIONAL,
    Flags: Initiator, Message ID=2),
    SK {D}  -->

                    <--  HDR(SPIi=xxx, SPIr=yyy, INFORMATIONAL,
                             Flags: Response, Message ID=2),
                             SK {}
```

刪除負載格式如下:

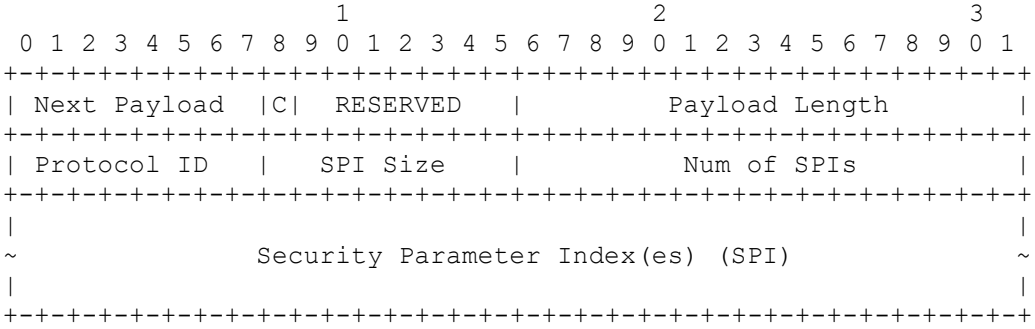


圖 17:刪除負載格式

- Protocol ID(協定 ID)（1 個八位元組） - IKE SA 必須為 1。
- SPI Size(SPI 大小)（1 個八位元組） - SPI 的八位元組長度，由協定 ID 定義。 IKE 必須為零（SPI 在訊息標頭中）。
- Num of SPIs(SPI 的數量)（2 個八位元組，無號整數） - 刪除負載中包含的 SPI 數量。 對於 IKE 而言這必須為零。

- Security Parameter Index(安全參數索引)（可變長度）- 識別要刪除的特定安全關聯。 該欄位的長度由 SPI 大小和 SPI 欄位確定。對於 IKE SA 刪除而言此欄位為空。

## B.2. 原始公鑰

在某些情況下，共享的密鑰身份驗證不夠安全，因為任何知道該密鑰的人都可以冒充伺服器。如果共享密鑰印製在設備的側面，那麼任何對設備進行物理存取的人都可以讀取它。在這樣的環境中，公鑰認證允許以最小的操作進行更強的認證。憑證支援非常複雜，並且通常不需要最小實現。使用原始公鑰更簡單，它與憑證類似。原始公鑰的指紋仍然可以透過例如將其印製在設備的側面來分配，從而允許類似於使用共享密鑰的設置。

原始公鑰也可用於“leap of faith”或 baby duck 模式進行初始設置，其中設備將其自身印入到第一次啟動時看到的第一個設備。在初始連接之後，它將伺服器的原始公鑰指紋儲存在自己的配置中，並驗證它永遠不會更改（除非發出“重置為出廠設置”或類似命令）。

這會更改初始 IKE_AUTH 負載，如下所示：

```
Initiator                          Responder
 -------------------------------------------------------------------
 HDR(SPIi=xxx, SPIr=yyy, IKE_AUTH,
     Flags: Initiator, Message ID=1),
     SK {IDi, CERT, AUTH, SAi2, TSi, TSr,
         N(INITIAL_CONTACT)}  -->

                    <--  HDR(SPIi=xxx, SPIr=yyy, IKE_AUTH, Flags:
                                Response, Message ID=1),
                                SK {IDr, CERT, AUTH, SAr2, TSi, TSr}
```

CERT 負載包含用於在生成 AUTH 負載時對 InitiatorSignedOctects / ResponderSignedOctects 的雜湊進行簽署的原始公鑰。最小實現應該使用 SHA-1 作為雜湊函數，因為它是 RFC 7296 中指定的“SHOULD”支援演算法，因此它最有可能被所有設備支援。

需注意 RFC 7296 已經廢棄舊的 Raw RSA Key 方法，且“IKEv2 的通用原始公鑰支援”[RFC7670]添加一種新格式，允許使用任何類型的原始公鑰和 IKEv2。本文僅指定如何使用新格式。

在這些設置中，可能根本不需要對伺服器進行身份驗證。如果最小設備正在向伺服器發送傳感器訊息，則伺服器想要驗證傳感器是否是其聲稱使用原始公鑰的使用者，但傳感器並不真正關心伺服器是誰。在這種情況下，NULL 身份驗證方法[RFC7619]會很有用，因為它允許設備進行單向身份驗證。

## 致謝

本文的大部分內容都是從 RFC7296 複製而來的。

## 作者資訊

Tero Kivinen
INSIDE 安全
Eerikinkatu 28
赫爾辛基 FI-00180
芬蘭
電子郵件：kivinen@iki.fi