Overview of EAPOL message exchange and formats

CableLabs

Steve Goeringer

s.goeringer@cablelabs.com

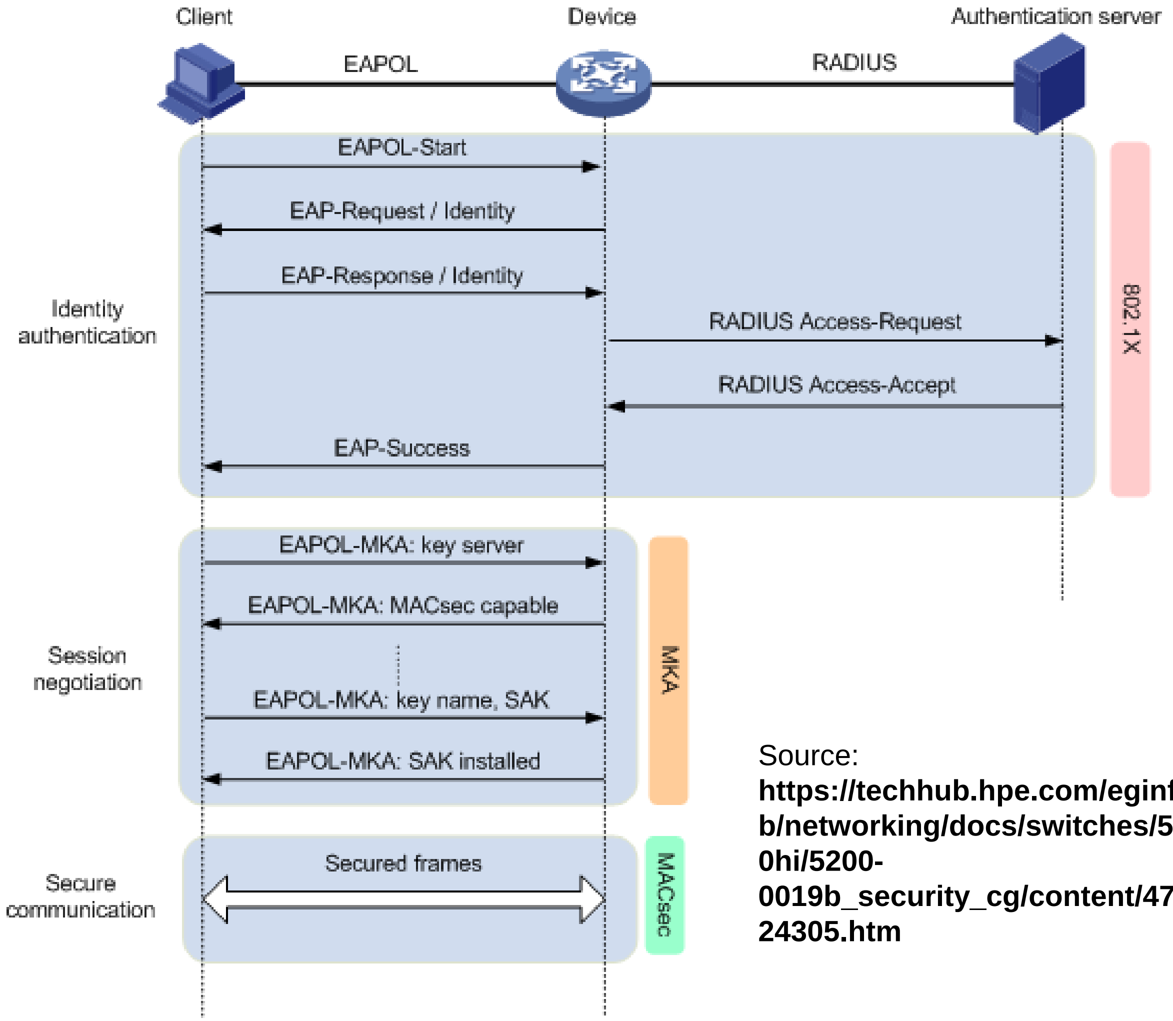
EAP/MKA based authentication

Security & Privacy Technologies
CableLabs®

- What do the messages shown here look like?
- How is this encapsulated?
- Where do the fields and values come from?

Key requirements

- Convey an initial encryption key for an LLID
  - How are different connectivity associations/security associations delineated – how does the LLC/SecY know which key to apply?
- Convey the high order value for the IC which will then be incremented using the MSPC clock



Source:
https://techhub.hpe.com/eginfolib/networking/docs/switches/5510hi/5200-0019b_security_cg/content/471724305.htm

Core References

- 802.1x-2020
- RFC 3748
  - EAP Packet Format
  - Initial EAP Request/Response Types

802.1x architecture

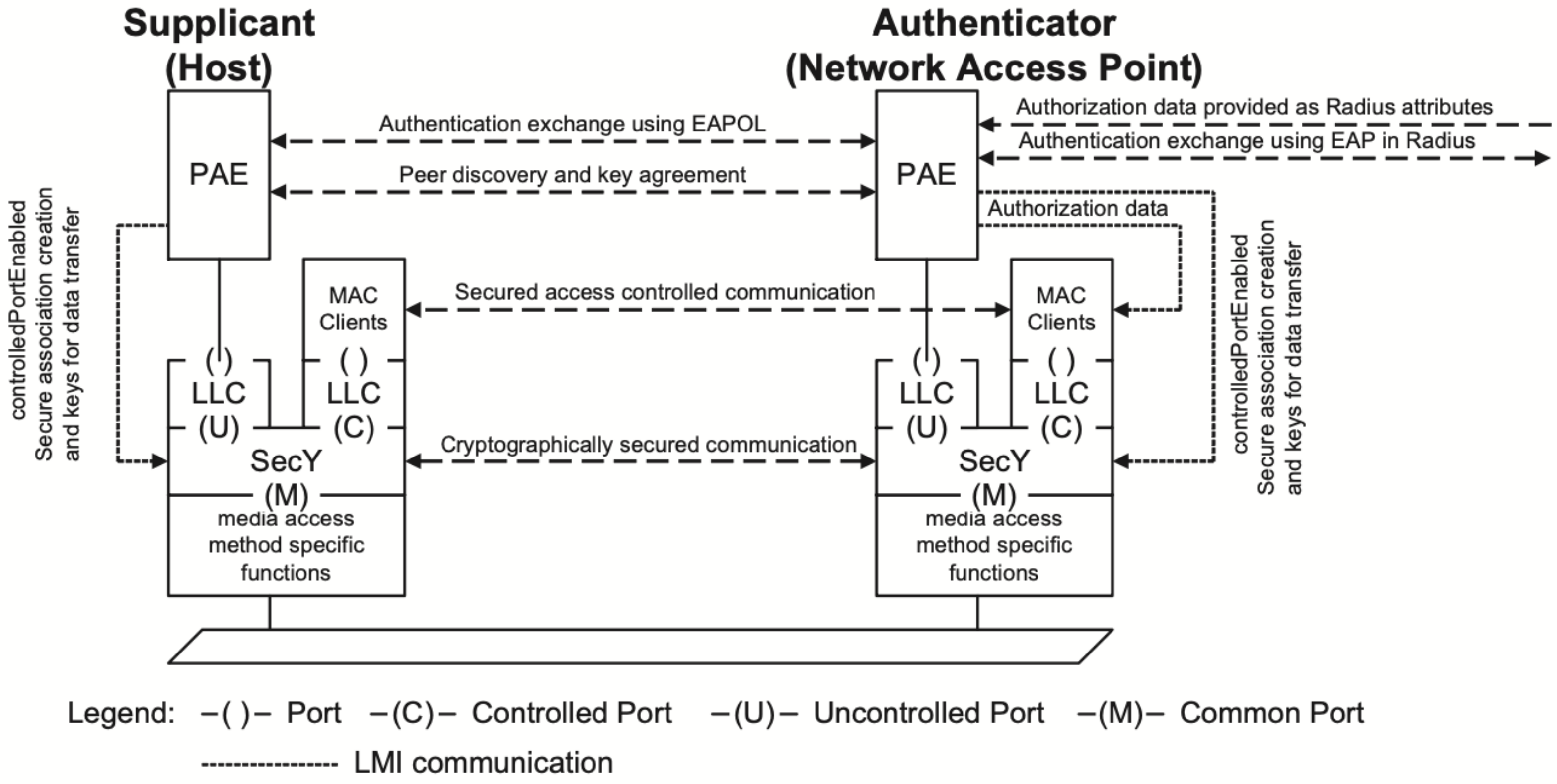

Supplicant (Host)
Authenticator (Network Access Point)
PAE
Authentication exchange using EAPOL
Peer discovery and key agreement
PAE
Authorization data provided as Radius attributes
Authentication exchange using EAP in Radius
Authorization data
controlledPortEnabled
Secure association creation
and keys for data transfer
MAC Clients
Secured access controlled communication
MAC Clients
( )
LLC
(U)
( )
LLC
(C)
SecY
(M)
media access method specific functions
Cryptographically secured communication
( )
LLC
(U)
( )
LLC
(C)
SecY
(M)
media access method specific functions
controlledPortEnabled
Secure association creation
and keys for data transfer
Legend: –( )– Port –(C)– Controlled Port –(U)– Uncontrolled Port –(M)– Common Port
---------------- LMI communication

Figure 7-7—Network access control with MACsec and a point-to-point LAN

- EAPOL exists between two PAEs
  - Supplicant
  - Network Access Point
- This assumes MACsec, but there is a model that doesn’t include MACSec

802.1x-2020, Figure 8-1—PAEs, PACP, EAP Messages, and EAPOL PDUs

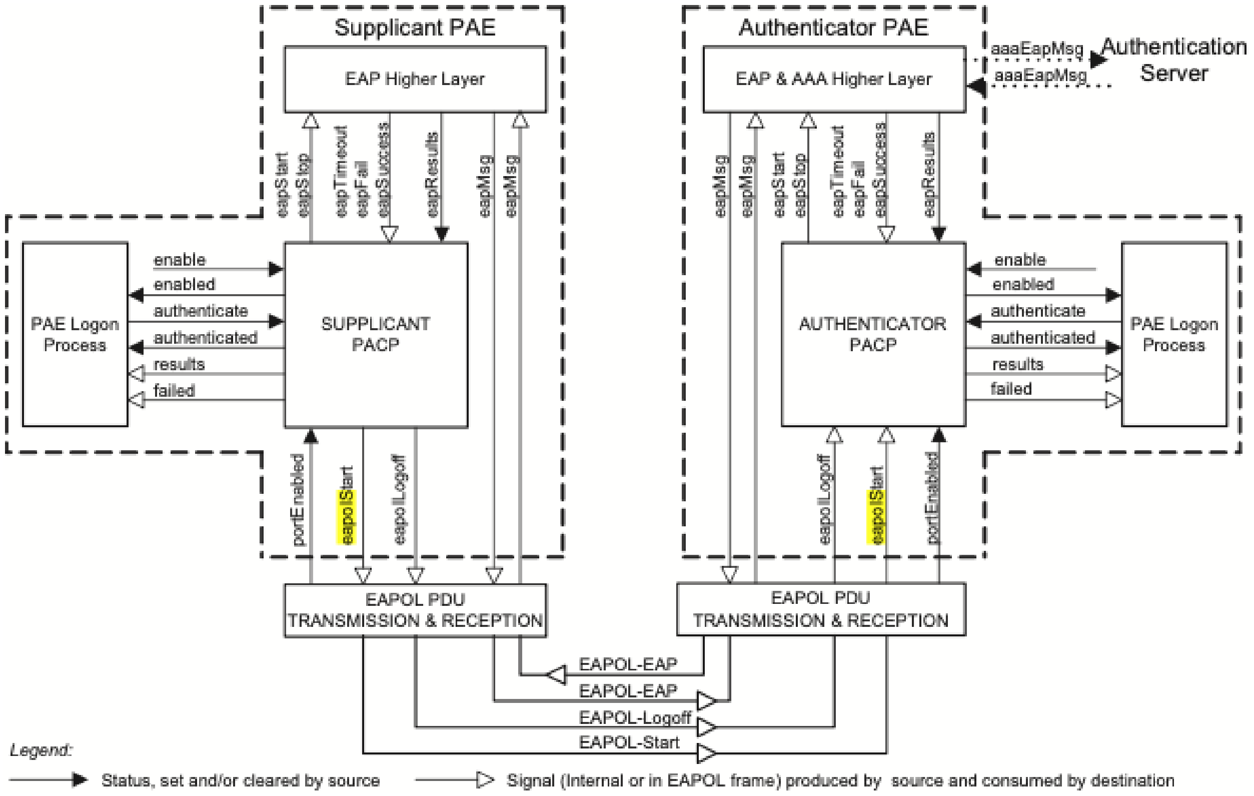
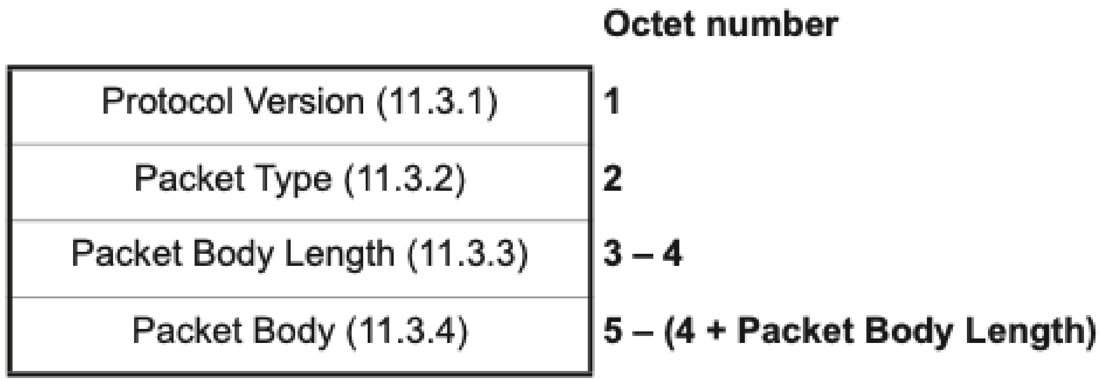


Figure 8-1—PAEs, PACP, EAP Messages, and EAPOL PDUs

EAPOL is a wrapper for EAP

- EAPOL PDUs are Ether Frames
  - PAE Ethertype: 88-8E
  - Encoding of octets explained in 802.1x 11.2
- PDU Data Structure
  - Protocol version: 0x03
    - Protocol version handling is defined in Section 11.5
  - Packet types each have graphical representation
  - Packet body length – two octets; 0 means no packet body present
- There are validation rules for received EAPOL PDUs in Section 11.4
- EAPOL addressing for each packet type is shown in Table 11-4



	Octet number
Protocol Version (11.3.1)	1
Packet Type (11.3.2)	2
Packet Body Length (11.3.3)	3 – 4
Packet Body (11.3.4)	5 – (4 + Packet Body Length)

Figure 11-1—Common EAPOL PDU structure

EAPOL Packet Types

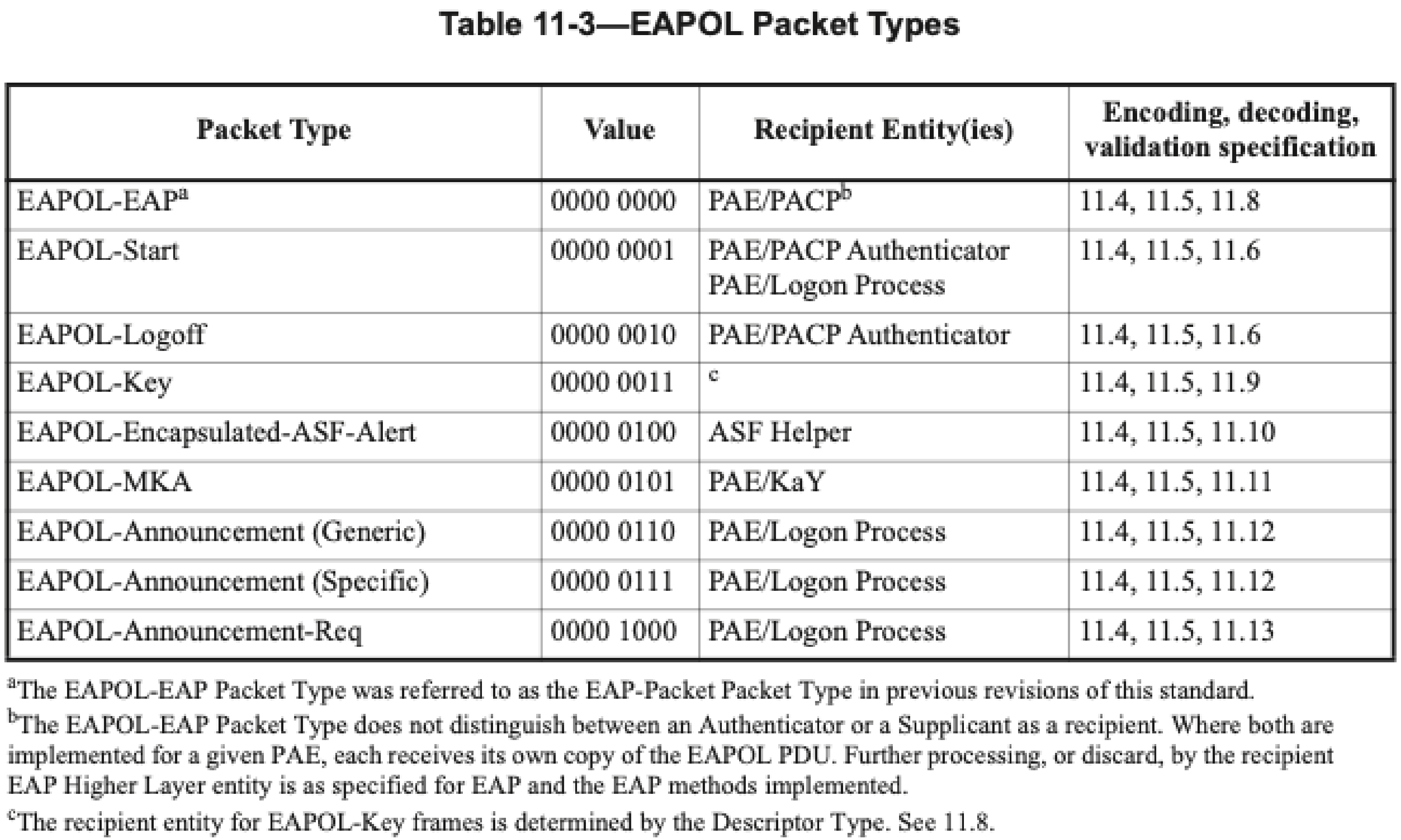


Table 11-3—EAPOL Packet Types

Packet Type	Value	Recipient Entity(ies)	Encoding, decoding, validation specification
EAPOL-EAP[a]	0000 0000	PAE/PACP[b]	11.4, 11.5, 11.8
EAPOL-Start	0000 0001	PAE/PACP Authenticator PAE/Logon Process	11.4, 11.5, 11.6
EAPOL-Logoff	0000 0010	PAE/PACP Authenticator	11.4, 11.5, 11.6
EAPOL-Key	0000 0011	[c]	11.4, 11.5, 11.9
EAPOL-Encapsulated-ASF-Alert	0000 0100	ASF Helper	11.4, 11.5, 11.10
EAPOL-MKA	0000 0101	PAE/KaY	11.4, 11.5, 11.11
EAPOL-Announcement (Generic)	0000 0110	PAE/Logon Process	11.4, 11.5, 11.12
EAPOL-Announcement (Specific)	0000 0111	PAE/Logon Process	11.4, 11.5, 11.12
EAPOL-Announcement-Req	0000 1000	PAE/Logon Process	11.4, 11.5, 11.13

[a] The EAPOL-EAP Packet Type was referred to as the EAP-Packet Packet Type in previous revisions of this standard.

[b] The EAPOL-EAP Packet Type does not distinguish between an Authenticator or a Supplicant as a recipient. Where both are implemented for a given PAE, each receives its own copy of the EAPOL PDU. Further processing, or discard, by the recipient EAP Higher Layer entity is as specified for EAP and the EAP methods implemented.

[c] The recipient entity for EAPOL-Key frames is determined by the Descriptor Type. See 11.8.

- All of these types are annotated in Clause 11
- Note that there is variation depending on Protocol Version
- EAPOL Announce and Start include TLVs that I have not sourced/identified

TLVs

- EAPOL v3 EAPOL-Start and EAPOL-Announce types include TLVs
- I don’t know if these TLVs will be useful
- We may be able to include the high order IC bits
  - Perhaps overhaul NID Set TLV by LLID|IC or something like that

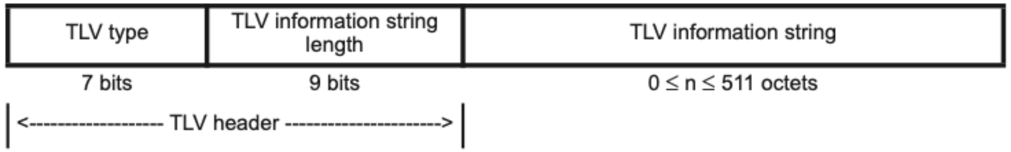
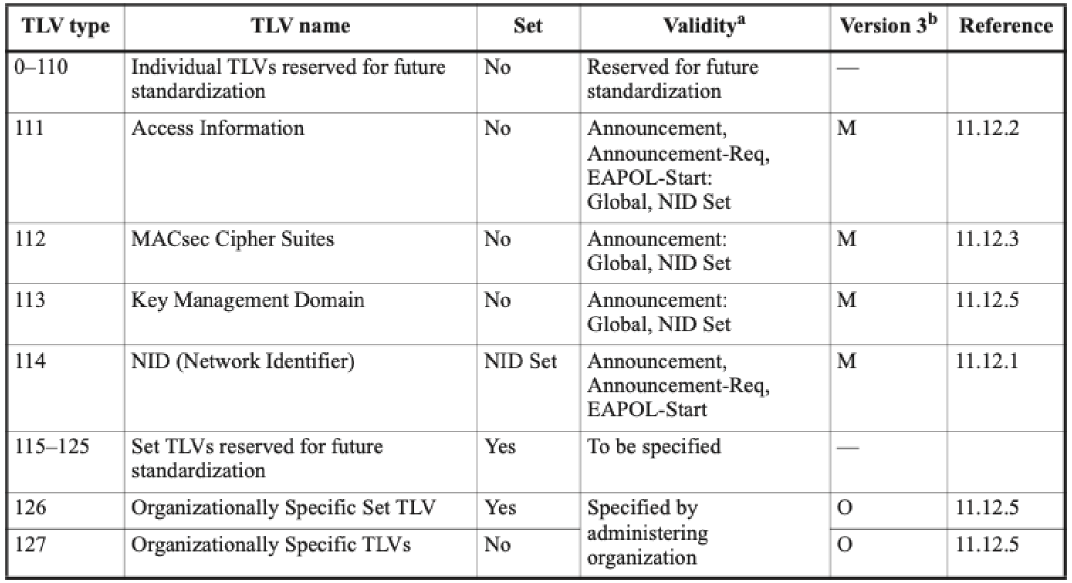

TLV type
TLV information string length
TLV information string
7 bits
9 bits
0 ≤ n ≤ 511 octets
|<------------------ TLV header ---------------------->|

Figure 11-19—EAPOL-Announcement TLV format

Table 11-8—EAPOL-Announcement TLVs



TLV type	TLV name	Set	Validity[a]	Version 3[b]	Reference
0–110	Individual TLVs reserved for future standardization	No	Reserved for future standardization	—	
111	Access Information	No	Announcement, Announcement-Req, EAPOL-Start: Global, NID Set	M	11.12.2
112	MACsec Cipher Suites	No	Announcement: Global, NID Set	M	11.12.3
113	Key Management Domain	No	Announcement: Global, NID Set	M	11.12.5
114	NID (Network Identifier)	NID Set	Announcement, Announcement-Req, EAPOL-Start	M	11.12.1
115–125	Set TLVs reserved for future standardization	Yes	To be specified	—	
126	Organizationally Specific Set TLV	Yes	Specified by administering organization	O	11.12.5
127	Organizationally Specific TLVs	No		O	11.12.5

[a] Specifies the EAPOL Packet Types:Set(s) in which a given TLV is valid.

[b] If Announcements claimed for EAPOL Protocol Version 3: M–mandatory to implement, O–optional,— ignore

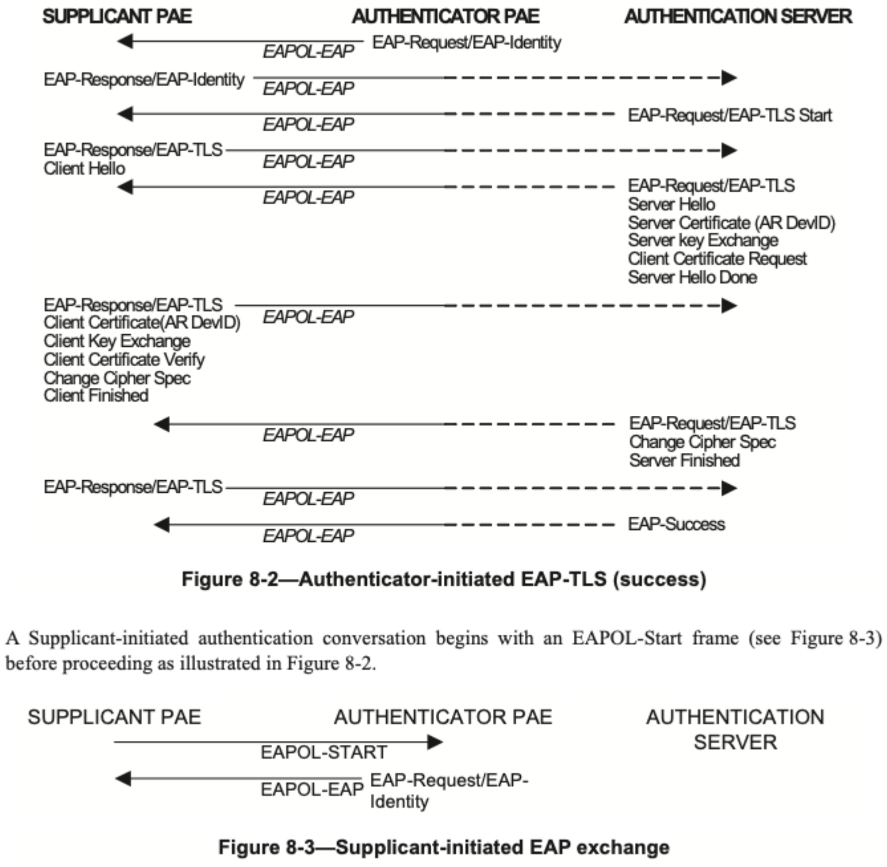
Questions

- Do we really need to use MKA? EAPOL-MKA encapsulates an MKPDU and is a specific EAPOL PDU Packet type. There is lots of encoding there that doesn't apply to our use case. EAP or EAPOL-Key may provide what we need to convey a key, keeping in mind we need to wrap the key (encrypt).
- Do we want to support certificate based mutual authentication? EAP-TLS includes message components beyond the scope of what is shown here.
  - "Specification of the higher layer PAE functions is outside the scope of this standard, though this standard does require the use of an EAP method that provides mutual authentication when EAP is supported, places further constraints on the methods to be used in conjunction with MKA, and mandates the use of EAP-TLS (IETF RFC 5216) for integration with IEEE Std 802.1AR (8.11). EAP protocol exchanges are defined by IETF EAP standards, IETF RFC 3748 [B14], and successor standards. One example of a AAA protocol, RADIUS, and its use for "pass-through" forwarding of EAP Messages to an Authentication Server, is defined by the IETF RADIUS standards, IETF RFC 2865 [B6], IETF RFC 2866 [B7], IETF RFC 3579 [B12], and successor standards."

802.1x-2020, Figure 8-3—Supplicant-initiated EAP [TLS] exchange

Security & Privacy Technologies

CableLabs®



SUPPLICANT PAE | AUTHENTICATOR PAE | AUTHENTICATION SERVER

EAPOL-EAP | EAP-Request/EAP-Identity

EAP-Response/EAP-Identity | EAPOL-EAP

EAPOL-EAP | EAP-Request/EAP-TLS Start

EAP-Response/EAP-TLS
Client Hello | EAPOL-EAP

EAPOL-EAP | EAP-Request/EAP-TLS
Server Hello
Server Certificate (AR DevID)
Server key Exchange
Client Certificate Request
Server Hello Done

EAP-Response/EAP-TLS
Client Certificate(AR DevID)
Client Key Exchange
Client Certificate Verify
Change Cipher Spec
Client Finished | EAPOL-EAP

EAPOL-EAP | EAP-Request/EAP-TLS
Change Cipher Spec
Server Finished

EAP-Response/EAP-TLS | EAPOL-EAP

EAPOL-EAP | EAP-Success

Figure 8-2—Authenticator-initiated EAP-TLS (success)

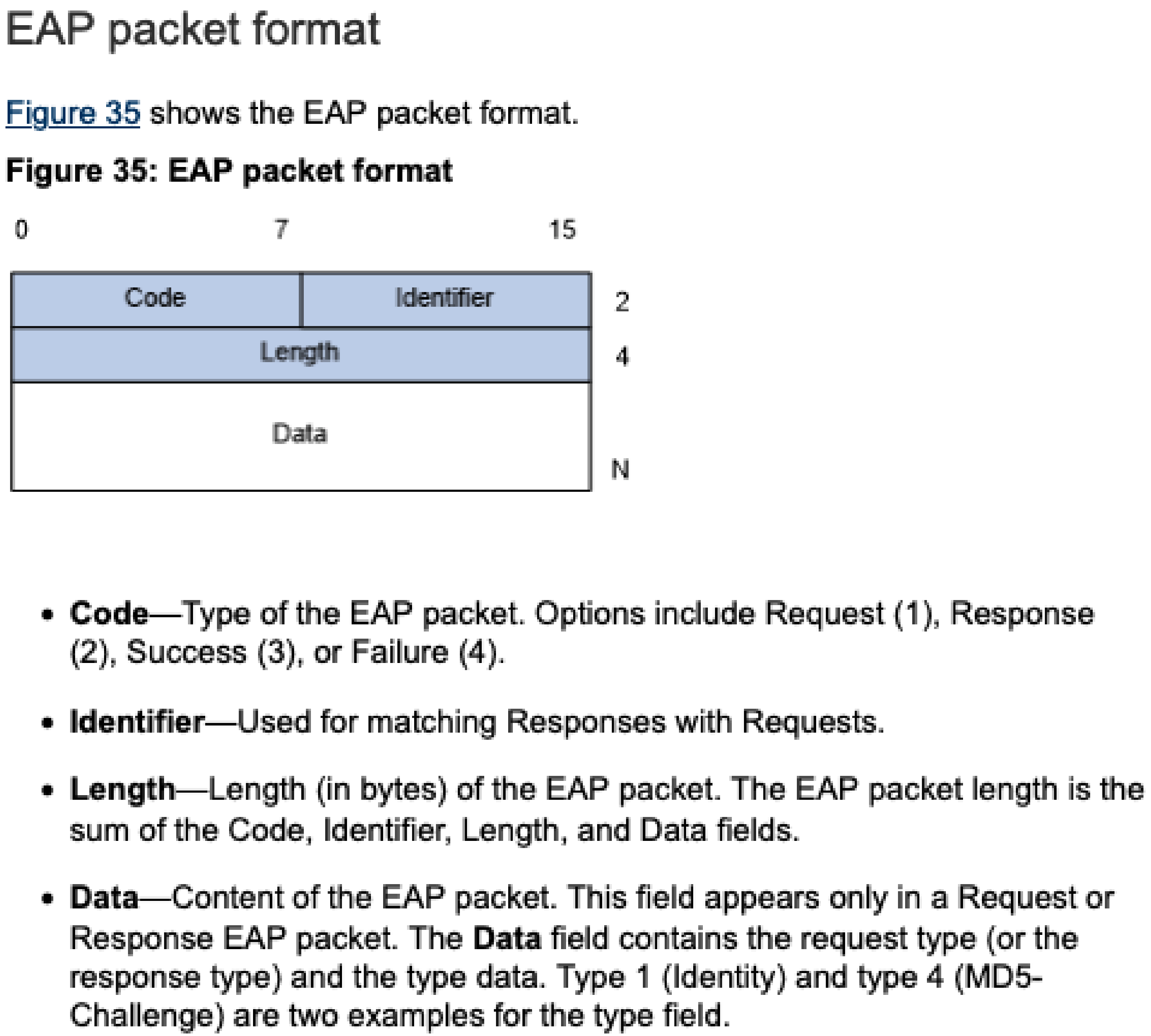
A Supplicant-initiated authentication conversation begins with an EAPOL-Start frame (see Figure 8-3) before proceeding as illustrated in Figure 8-2.

SUPPLICANT PAE | AUTHENTICATOR PAE | AUTHENTICATION SERVER

EAPOL-START

EAPOL-EAP | EAP-Request/EAP-Identity

Figure 8-3—Supplicant-initiated EAP exchange



EAP packet format

Figure 35 shows the EAP packet format.

Figure 35: EAP packet format

0	7	15
Code	Identifier	2
Length		4
Data		N

- **Code**—Type of the EAP packet. Options include Request (1), Response (2), Success (3), or Failure (4).
- **Identifier**—Used for matching Responses with Requests.
- **Length**—Length (in bytes) of the EAP packet. The EAP packet length is the sum of the Code, Identifier, Length, and Data fields.
- **Data**—Content of the EAP packet. This field appears only in a Request or Response EAP packet. The **Data** field contains the request type (or the response type) and the type data. Type 1 (Identity) and type 4 (MD5-Challenge) are two examples for the type field.

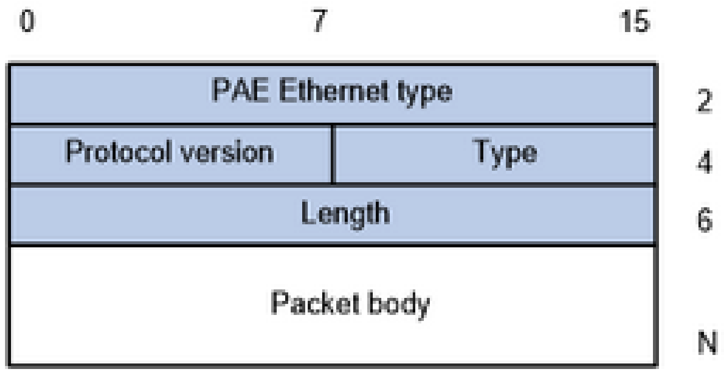
EAP PDUs are defined in RFC 3748

https://techhub.hpe.com/eginfolib/networking/docs/switches/5130ei/5200-3946_security_cg/content/485048061.htm

EAPOL packet format

Figure 36 shows the EAPOL packet format.

Figure 36: EAPOL packet format



0 7 15

PAE Ethernet type 2

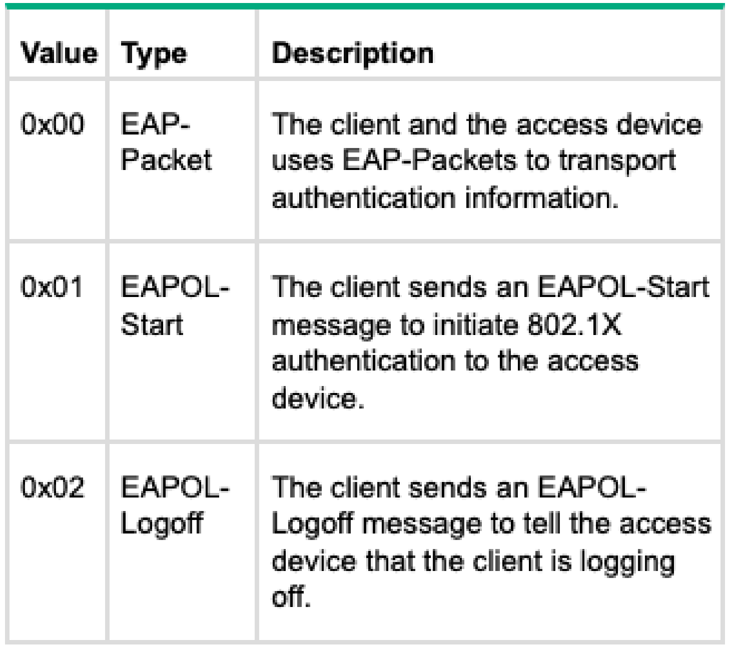
Protocol version | Type 4

Length 6

Packet body N

- **PAE Ethernet type**—Protocol type. It takes the value 0x888E for EAPOL.
- **Protocol version**—The EAPOL protocol version used by the EAPOL packet sender.
- **Type**—Type of the EAPOL packet. Table 5 lists the types of EAPOL packets supported by Hewlett Packard Enterprise implementation of 802.1X.

Table 5: Types of EAPOL packets



Value	Type	Description
0x00	EAP-Packet	The client and the access device uses EAP-Packets to transport authentication information.
0x01	EAPOL-Start	The client sends an EAPOL-Start message to initiate 802.1X authentication to the access device.
0x02	EAPOL-Logoff	The client sends an EAPOL-Logoff message to tell the access device that the client is logging off.

- **Length**—Data length in bytes, or length of the Packet body. If packet type is EAPOL-Start or EAPOL-Logoff, this field is set to 0, and no Packet body field follows.
- **Packet body**—Content of the packet. When the EAPOL packet type is EAP-Packet, the Packet body field contains an EAP packet.

https://techhub.hpe.com/eginfolib/networking/docs/switches/5130ei/5200-3946_security_cg/content/485048061.htm