Assignment no 14

Title : To study the IPsec (ESP and AH) protocol by capturing the packets using Wireshark tool.

Theory :

IPSec uses two distinct protocols, Authentication Header (AH) and Encapsulating Security Payload (ESP), which are defined by the IETF.

The AH protocol provides a mechanism for authentication only. AH provides data integrity, data origin authentication, and an optional replay protection service. Data integrity is ensured by using a message digest that is generated by an algorithm such as HMAC-MD5 or HMAC-SHA. Data origin authentication is ensured by using a shared secret key to create the message digest. Replay protection is provided by using a sequence number field with the AH header. AH authenticates IP headers and their payloads, with the exception of certain header fields that can be legitimately changed in transit, such as the Time To Live (TTL) field.

The ESP protocol provides data confidentiality (encryption) and authentication (data integrity, data origin authentication, and replay protection). ESP can be used with confidentiality only, authentication only, or both confidentiality and authentication. When ESP provides authentication functions, it uses the same algorithms as AH, but the coverage is different. AH-style authentication authenticates the entire IP packet, including the outer IP header, while the ESP authentication mechanism authenticates only the IP datagram portion of the IP packet.

Either protocol can be used alone to protect an IP packet, or both protocols can be applied together to the same IP packet. The choice of IPSec protocol is determined by the security needs of your installation, and is configured by the administrator. It does not have to be applied system-wide, and can be configured differently for each set of connection endpoints. For a dynamic tunnel, the choice of IPSec protocol is configured using the IpDataOffer statement in an IP security policy configuration file. For a manual tunnel, the choice of IPSec protocol is configured using the IpManVpnAction statement in an IP security policy configuration file..

IPSec provides **confidentiality, integrity, authenticity, and replay protection** through two new protocols. These protocols are called Authentication Header (AH) and Encapsulated Security Payload (ESP). AH provides authentication, integrity, and replay protection (but not confidentiality).The size of the *Authentication Data* field is variable to support different datagram lengths and hashing algorithms. Its total length must be a multiple of 32 bits. Also, the entire header must be a multiple of either 32 bits (for IPv4) or 64 bits (for IPv6), so additional padding may be added to the *Authentication Data* field if necessary. The format of the Authentication Header is shown as below

able 79: IPSec Authentication Header (AH) Format

Field Name	Size (bytes)	Description
Next Header	1	***Next Header:*** Contains the protocol number of the next header after the AH. Used to link headers together.
Payload Len	1	***Payload Length:*** Despite its name, this field measures the length of the authentication header itself, not the payload. (I wonder what the history is behind that!) It is measured in 32 bit units, with 2 subtracted for consistency with how header lengths are normally calculated in IPv6.
Reserved	2	***Reserved:*** Not used; set to zeroes.
SPI	4	***Security Parameter Index (SPI):*** A 32-bit value that when combined with the destination address and security protocol type (which here is obviously the one for AH) identifies the security association to be used for this datagram. See the topic on security associations for more details.
Sequence Number	4	***Sequence Number:*** This is a counter field that is initialized to zero when a security association is formed between two devices, and then

		incremented for each datagram sent using that SA. This uniquely identifies each datagram on an SA and is used to provide protection against replay attacks by preventing the retransmission of captured datagrams.
Authentication Data	Variable	***Authentication Data:*** This field contains the result of the hashing algorithm performed by the AH protocol, the Integrity Check Value (ICV).

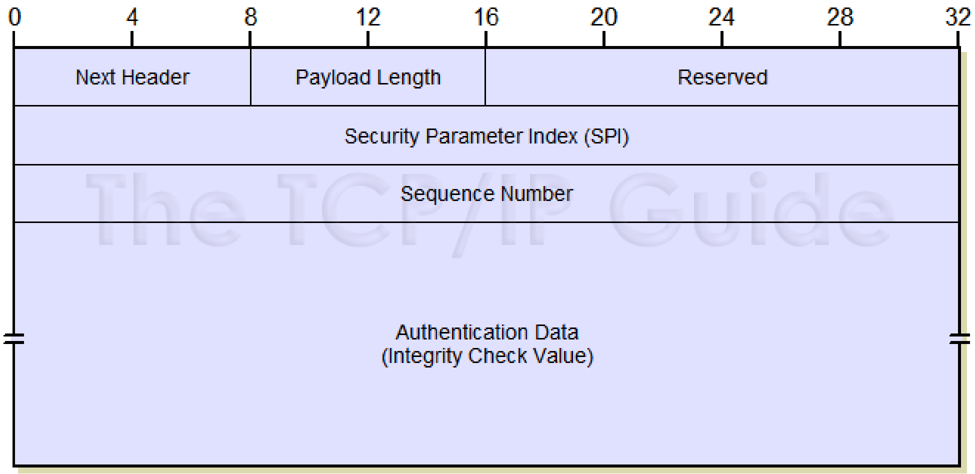
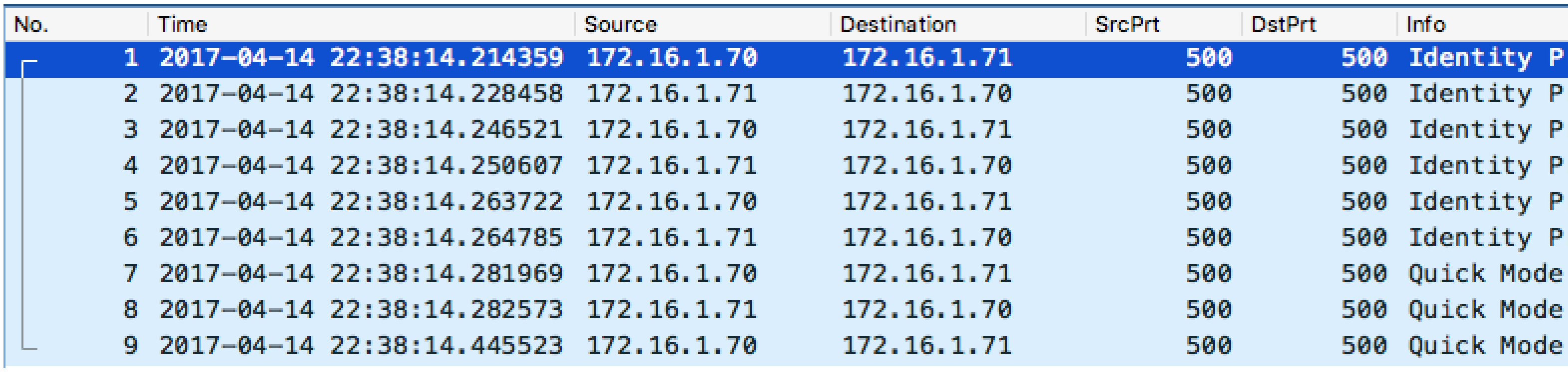


Figure 1: IPSec Authentication Header (AH) Format

First Identity Protection (Main Mode) messages negotiate security parameters to protect the next 3 messages (Quick Mode) and whatever is negotiated in Phase 2 is used to protect production traffic (ESP or AH, normally ESP for site-site VPN).
We call first 6 messages Phase 1 and last 3 messages as Phase 2.



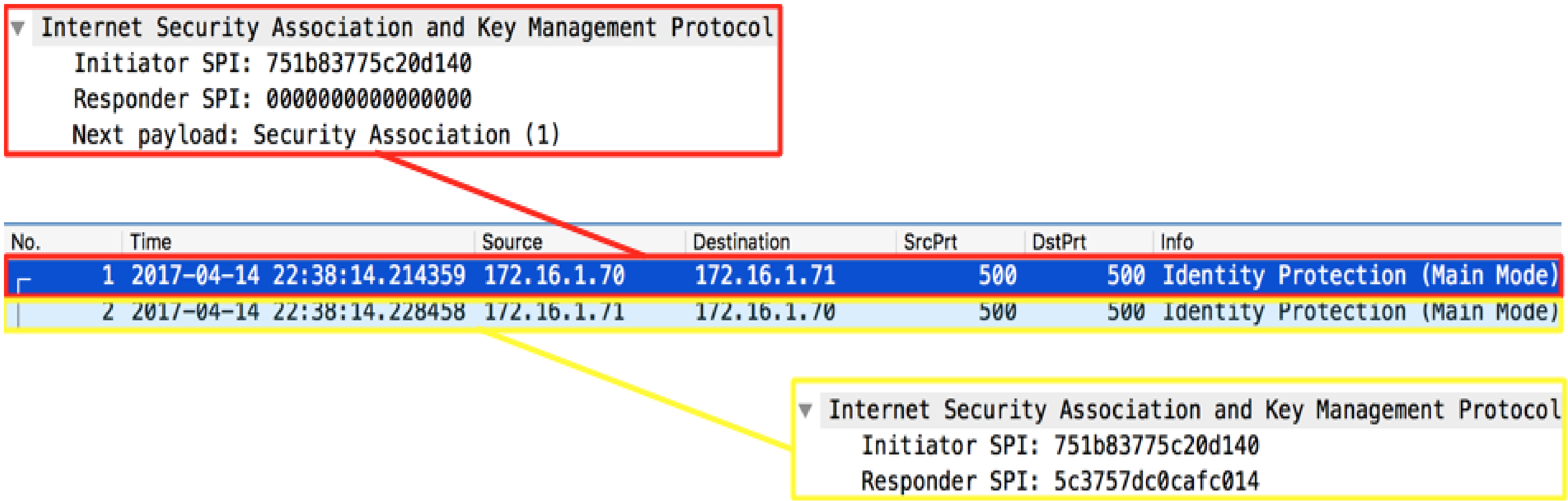
No.	Time	Source	Destination	SrcPrt	DstPrt	Info
1	2017-04-14 22:38:14.214359	172.16.1.70	172.16.1.71	500	500	Identity P
2	2017-04-14 22:38:14.228458	172.16.1.71	172.16.1.70	500	500	Identity P
3	2017-04-14 22:38:14.246521	172.16.1.70	172.16.1.71	500	500	Identity P
4	2017-04-14 22:38:14.250607	172.16.1.71	172.16.1.70	500	500	Identity P
5	2017-04-14 22:38:14.263722	172.16.1.70	172.16.1.71	500	500	Identity P
6	2017-04-14 22:38:14.264785	172.16.1.71	172.16.1.70	500	500	Identity P
7	2017-04-14 22:38:14.281969	172.16.1.70	172.16.1.71	500	500	Quick Mode
8	2017-04-14 22:38:14.282573	172.16.1.71	172.16.1.70	500	500	Quick Mode
9	2017-04-14 22:38:14.445523	172.16.1.70	172.16.1.71	500	500	Quick Mode

Sample pcap: IPSEC-tunnel-capture-1.pcap (for instructions on how to decrypt it just go to website where I got this sample capture: http://ruwanindikaprasanna.blogspot.com/2017/04/ipsec-capture-with-decryption.html)

2. Phase 1

2.1 Policy Negotiation

Both peers add a unique SPI just to uniquely identify each side's Security Association (SA):

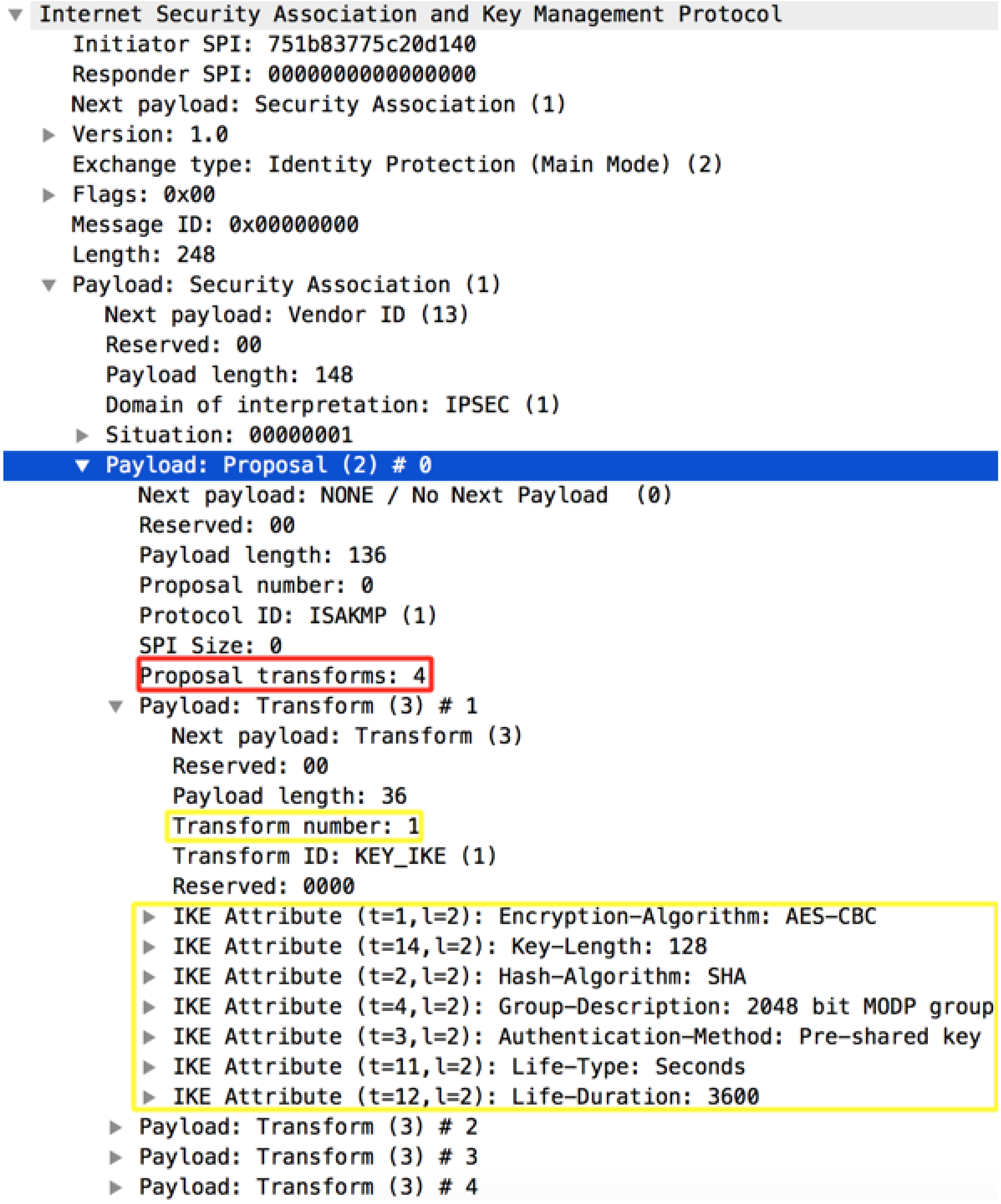


```
▼ Internet Security Association and Key Management Protocol
    Initiator SPI: 751b83775c20d140
    Responder SPI: 0000000000000000
    Next payload: Security Association (1)
```

```
No.   Time                        Source        Destination   SrcPrt  DstPrt  Info
  1   2017-04-14 22:38:14.214359  172.16.1.70   172.16.1.71      500     500  Identity Protection (Main Mode)
  2   2017-04-14 22:38:14.228458  172.16.1.71   172.16.1.70      500     500  Identity Protection (Main Mode)
```

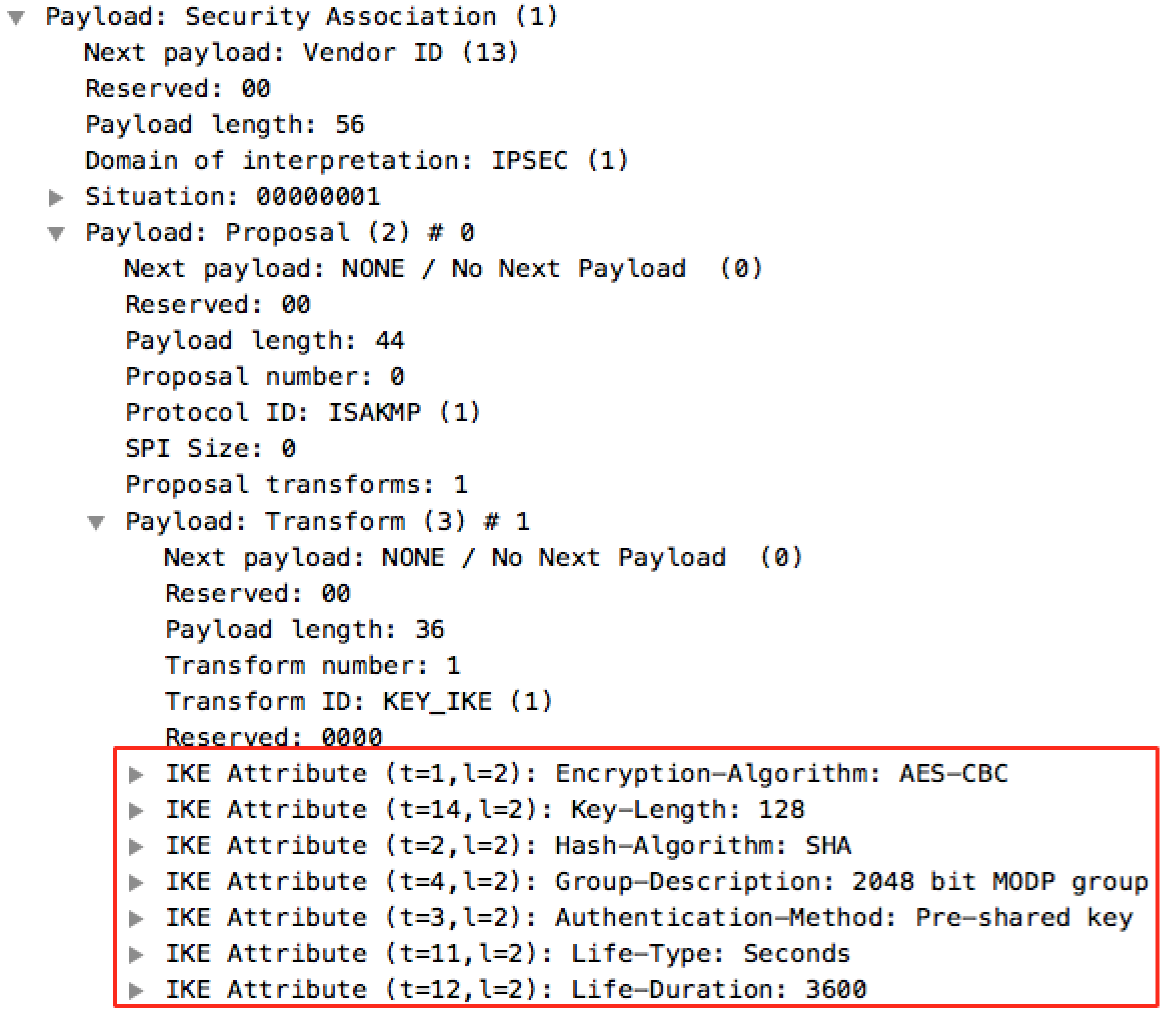
```
▼ Internet Security Association and Key Management Protocol
    Initiator SPI: 751b83775c20d140
    Responder SPI: 5c3757dc0cafc014
```

In **frame #1**, the Initiator (**.70**) sends a set of Proposals containing a set of security parameters (**Transforms**) that Responder (**.71**) can pick if it matches its local policies:



```
▼ Internet Security Association and Key Management Protocol
    Initiator SPI: 751b83775c20d140
    Responder SPI: 0000000000000000
    Next payload: Security Association (1)
  ▶ Version: 1.0
    Exchange type: Identity Protection (Main Mode) (2)
  ▶ Flags: 0x00
    Message ID: 0x00000000
    Length: 248
  ▼ Payload: Security Association (1)
      Next payload: Vendor ID (13)
      Reserved: 00
      Payload length: 148
      Domain of interpretation: IPSEC (1)
    ▶ Situation: 00000001
    ▼ Payload: Proposal (2) # 0
        Next payload: NONE / No Next Payload  (0)
        Reserved: 00
        Payload length: 136
        Proposal number: 0
        Protocol ID: ISAKMP (1)
        SPI Size: 0
        Proposal transforms: 4
      ▼ Payload: Transform (3) # 1
          Next payload: Transform (3)
          Reserved: 00
          Payload length: 36
          Transform number: 1
          Transform ID: KEY_IKE (1)
          Reserved: 0000
        ▶ IKE Attribute (t=1,l=2): Encryption-Algorithm: AES-CBC
        ▶ IKE Attribute (t=14,l=2): Key-Length: 128
        ▶ IKE Attribute (t=2,l=2): Hash-Algorithm: SHA
        ▶ IKE Attribute (t=4,l=2): Group-Description: 2048 bit MODP group
        ▶ IKE Attribute (t=3,l=2): Authentication-Method: Pre-shared key
        ▶ IKE Attribute (t=11,l=2): Life-Type: Seconds
        ▶ IKE Attribute (t=12,l=2): Life-Duration: 3600
      ▶ Payload: Transform (3) # 2
      ▶ Payload: Transform (3) # 3
      ▶ Payload: Transform (3) # 4
```

Fair enough, in frame #2 the Responder (**.71**) picks one of the **Transforms**:



```
▼ Payload: Security Association (1)
    Next payload: Vendor ID (13)
    Reserved: 00
    Payload length: 56
    Domain of interpretation: IPSEC (1)
  ▶ Situation: 00000001
  ▼ Payload: Proposal (2) # 0
      Next payload: NONE / No Next Payload  (0)
      Reserved: 00
      Payload length: 44
      Proposal number: 0
      Protocol ID: ISAKMP (1)
      SPI Size: 0
      Proposal transforms: 1
    ▼ Payload: Transform (3) # 1
        Next payload: NONE / No Next Payload  (0)
        Reserved: 00
        Payload length: 36
        Transform number: 1
        Transform ID: KEY_IKE (1)
        Reserved: 0000
      ▶ IKE Attribute (t=1,l=2): Encryption-Algorithm: AES-CBC
      ▶ IKE Attribute (t=14,l=2): Key-Length: 128
      ▶ IKE Attribute (t=2,l=2): Hash-Algorithm: SHA
      ▶ IKE Attribute (t=4,l=2): Group-Description: 2048 bit MODP group
      ▶ IKE Attribute (t=3,l=2): Authentication-Method: Pre-shared key
      ▶ IKE Attribute (t=11,l=2): Life-Type: Seconds
      ▶ IKE Attribute (t=12,l=2): Life-Duration: 3600
```

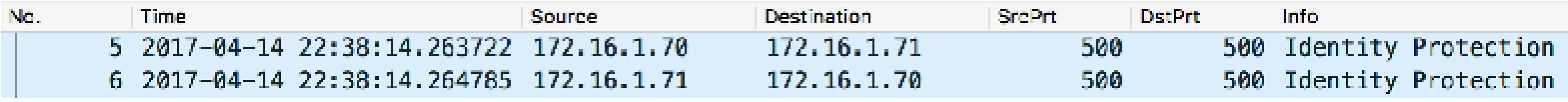
2.2 DH Key Exchange

Then, next 2 Identity Protection packets both peers exchange Diffie-Hellman public key values and nonces (random numbers) which will then allow both peers to agree on a shared secret key:

With DH public key value and the nonce both peers will generate a seed key called SKEYID.
A further 3 session keys will be generated using this seed key for different purposes:
SKEYID_d (d for derivative): not used by Phase 1. It is used as seed key for Phase2 keys, i.e. seed key for production traffic keys in Plain English.
SKEYID_a (a for authentication): this key is used to protect message integrity in every subsequent packets as soon as both peers are authenticated (peers will authenticate each other in next 2 packets). Yes, I know, we verify the integrity by using a hash but throwing a key into a hash adds stronger security to hash and it's called HMAC.
SKEYID_e (e for encryption): you'll see that the next 2 packets are also encrypted. As selected encryption algorithm for this phase was AES-CBC (128-bits) then we use AES with this key to symmetrically encrypt further data.
Nonce is just to protect against replay attacks by adding some randomness to key generation

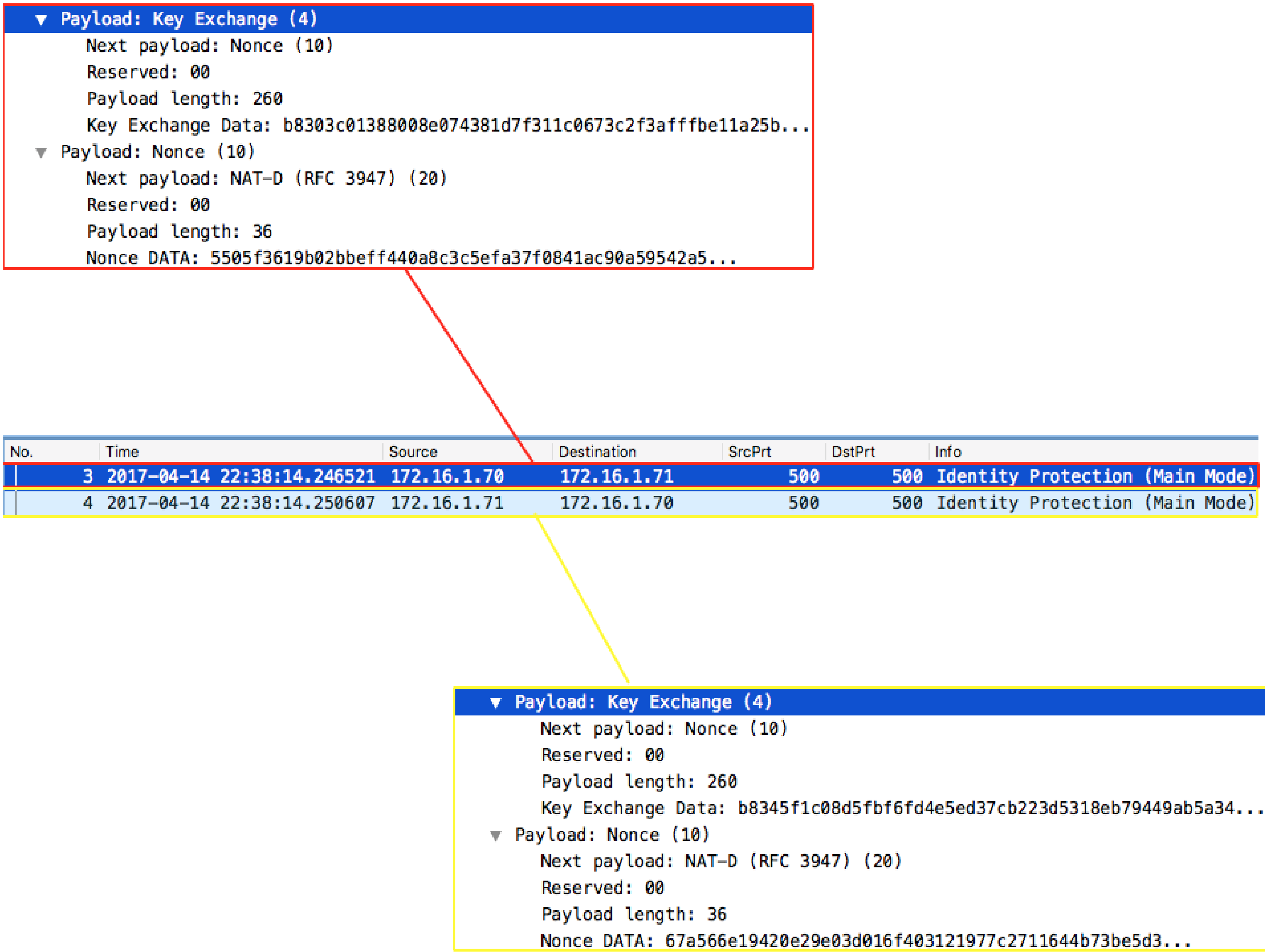
2.3 Authentication

The purpose of this exchange is to confirm each other's identity. If we said we're going to do this using pre-shared keys then verification consists of checking whether both sides has the same pre-shared key. If it is RSA certificate then peers exchange RSA certificates and assuming the CA that signed each side is trusted then verification complete successfully.
In our case, this is done via pre-shared keys:



```
No.      Time                        Source        Destination    SrcPrt   DstPrt   Info
      5  2017-04-14 22:38:14.263722  172.16.1.70   172.16.1.71       500      500  Identity Protection
      6  2017-04-14 22:38:14.264785  172.16.1.71   172.16.1.70       500      500  Identity Protection
```

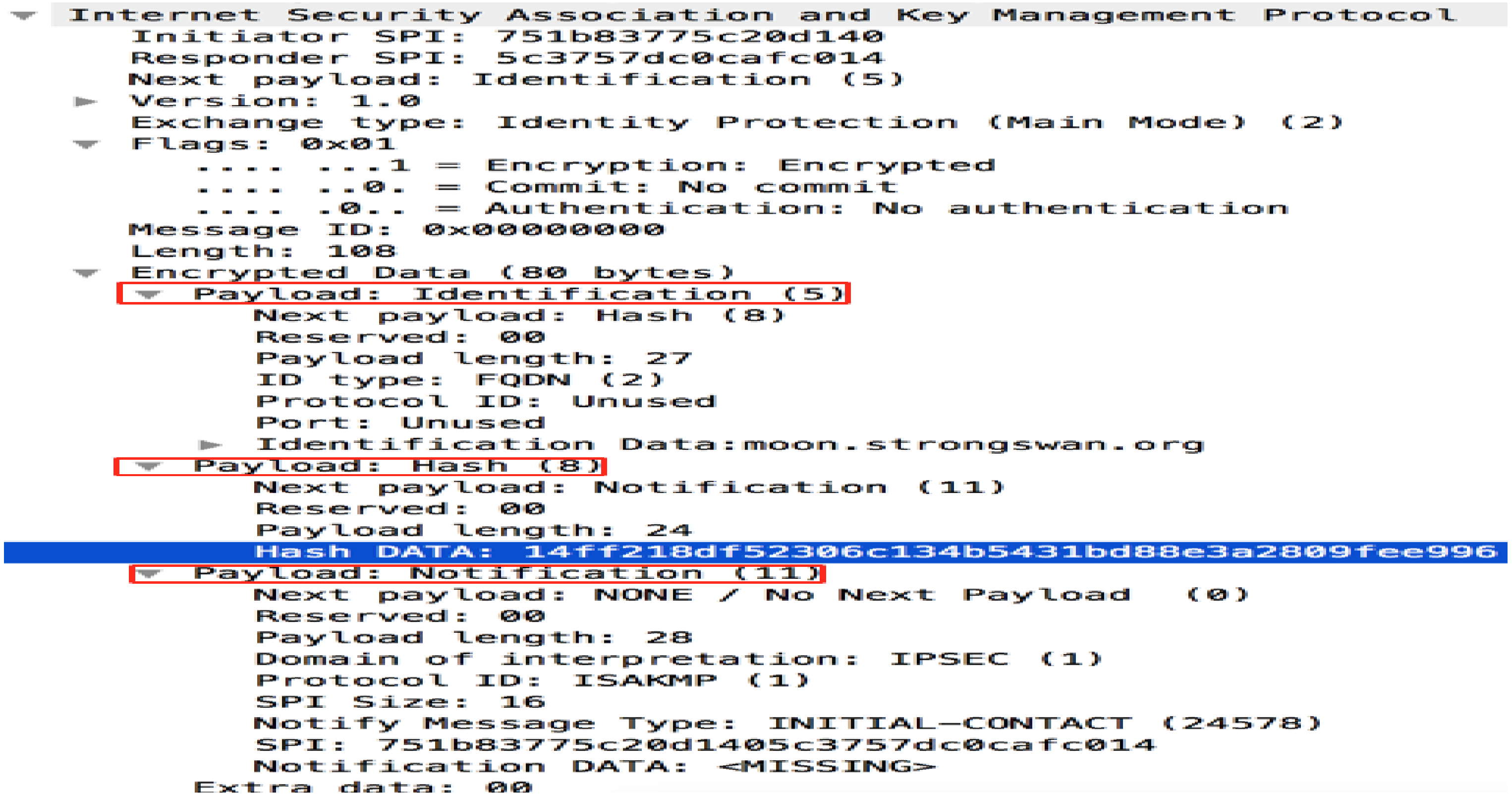
In packet #5 the Initiator sends a hash generated using pre-shared key set as key material so that only those who possess pre-master key can do it:



```
▼ Payload: Key Exchange (4)
      Next payload: Nonce (10)
      Reserved: 00
      Payload length: 260
      Key Exchange Data: b8303c01388008e074381d7f311c0673c2f3afffbe11a25b...
▼ Payload: Nonce (10)
      Next payload: NAT-D (RFC 3947) (20)
      Reserved: 00
      Payload length: 36
      Nonce DATA: 5505f3619b02bbeff440a8c3c5efa37f0841ac90a59542a5...
```

```
No.  Time                        Source       Destination  SrcPrt  DstPrt  Info
  3  2017-04-14 22:38:14.246521  172.16.1.70  172.16.1.71     500     500  Identity Protection (Main Mode)
  4  2017-04-14 22:38:14.250607  172.16.1.71  172.16.1.70     500     500  Identity Protection (Main Mode)
```

```
▼ Payload: Key Exchange (4)
      Next payload: Nonce (10)
      Reserved: 00
      Payload length: 260
      Key Exchange Data: b8345f1c08d5fbf6fd4e5ed37cb223d5318eb79449ab5a34...
▼ Payload: Nonce (10)
      Next payload: NAT-D (RFC 3947) (20)
      Reserved: 00
      Payload length: 36
      Nonce DATA: 67a566e19420e29e03d016f403121977c2711644b73be5d3...
```



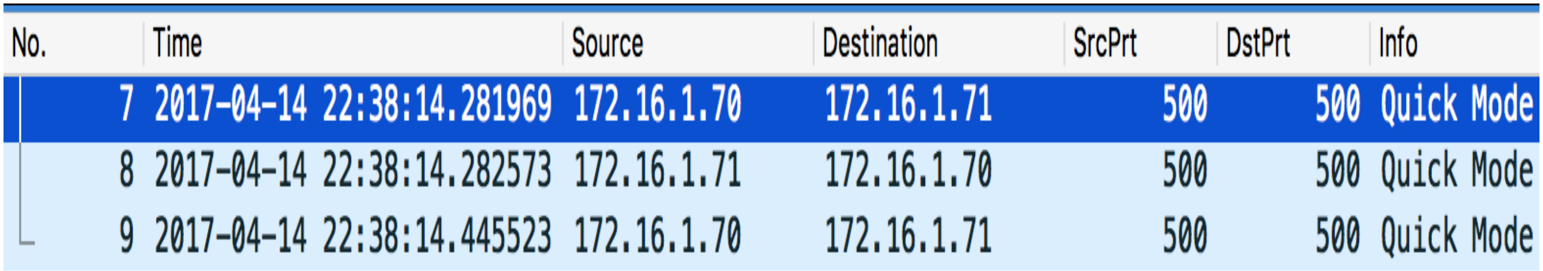
```
▼ Internet Security Association and Key Management Protocol
      Initiator SPI: 751b83775c20d140
      Responder SPI: 5c3757dc0cafc014
      Next payload: Identification (5)
   ► Version: 1.0
      Exchange type: Identity Protection (Main Mode) (2)
   ▼ Flags: 0x01
         .... ...1 = Encryption: Encrypted
         .... ..0. = Commit: No commit
         .... .0.. = Authentication: No authentication
      Message ID: 0x00000000
      Length: 108
   ▼ Encrypted Data (80 bytes)
      ▼ Payload: Identification (5)
            Next payload: Hash (8)
            Reserved: 00
            Payload length: 27
            ID type: FQDN (2)
            Protocol ID: Unused
            Port: Unused
         ► Identification Data:moon.strongswan.org
      ▼ Payload: Hash (8)
            Next payload: Notification (11)
            Reserved: 00
            Payload length: 24
            Hash DATA: 14ff218df52306c134b5431bd88e3a2809fee996
      ▼ Payload: Notification (11)
            Next payload: NONE / No Next Payload  (0)
            Reserved: 00
            Payload length: 28
            Domain of interpretation: IPSEC (1)
            Protocol ID: ISAKMP (1)
            SPI Size: 16
            Notify Message Type: INITIAL-CONTACT (24578)
            SPI: 751b83775c20d1405c3757dc0cafc014
            Notification DATA: <MISSING>
         Extra data: 00
```

The responder performs the same calculation and confirms the hash is correct.
Responder also sends a similar packet back to Initiator in frame #6 but I skipped for brevity.

Now we're ready for Phase 2.

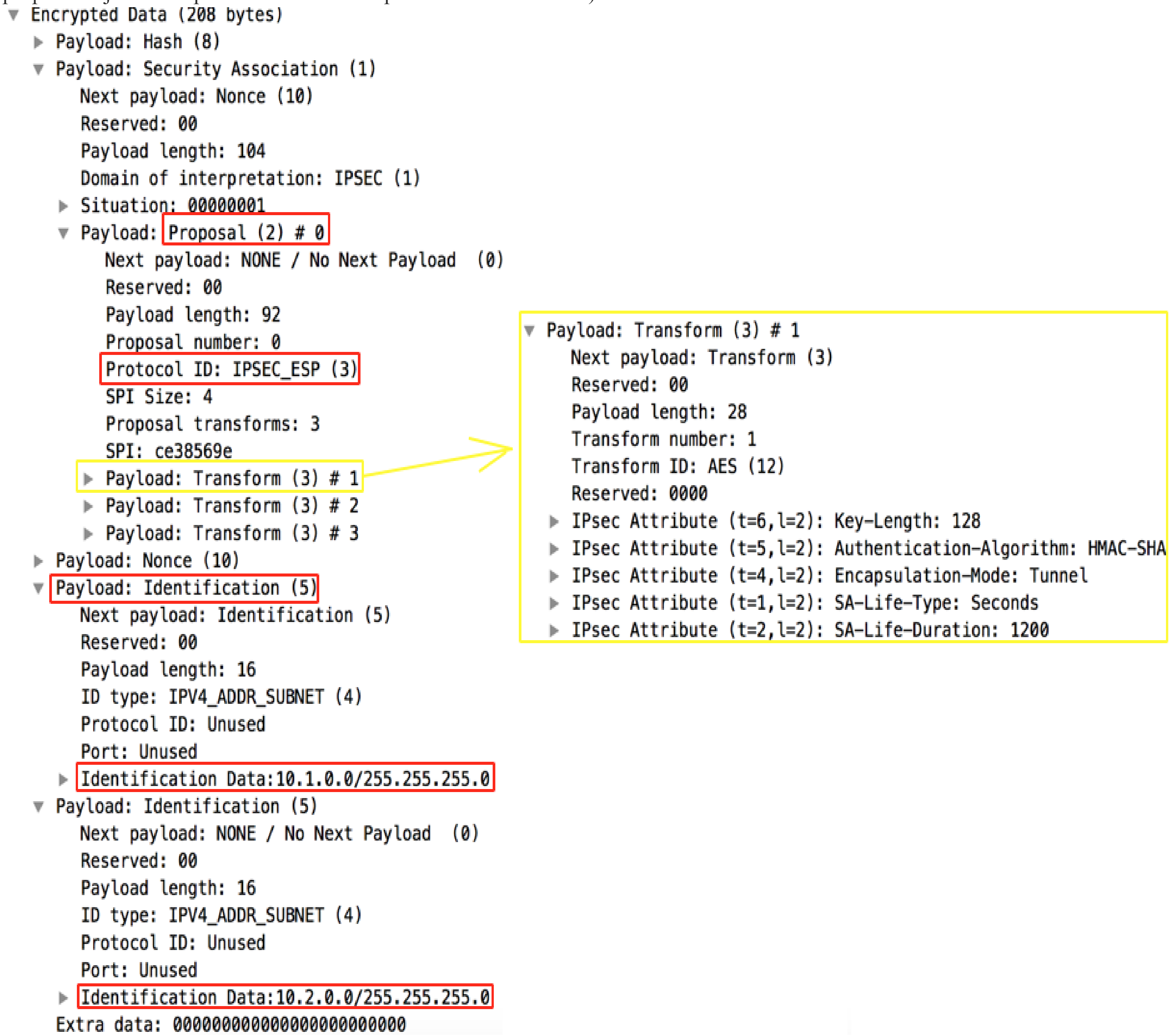
3. Phase 2

The purpose of this phase is to establish the security parameters that will be used for production traffic (IPSec SA):



No.	Time	Source	Destination	SrcPrt	DstPrt	Info
7	2017-04-14 22:38:14.281969	172.16.1.70	172.16.1.71	500	500	Quick Mode
8	2017-04-14 22:38:14.282573	172.16.1.71	172.16.1.70	500	500	Quick Mode
9	2017-04-14 22:38:14.445523	172.16.1.70	172.16.1.71	500	500	Quick Mode

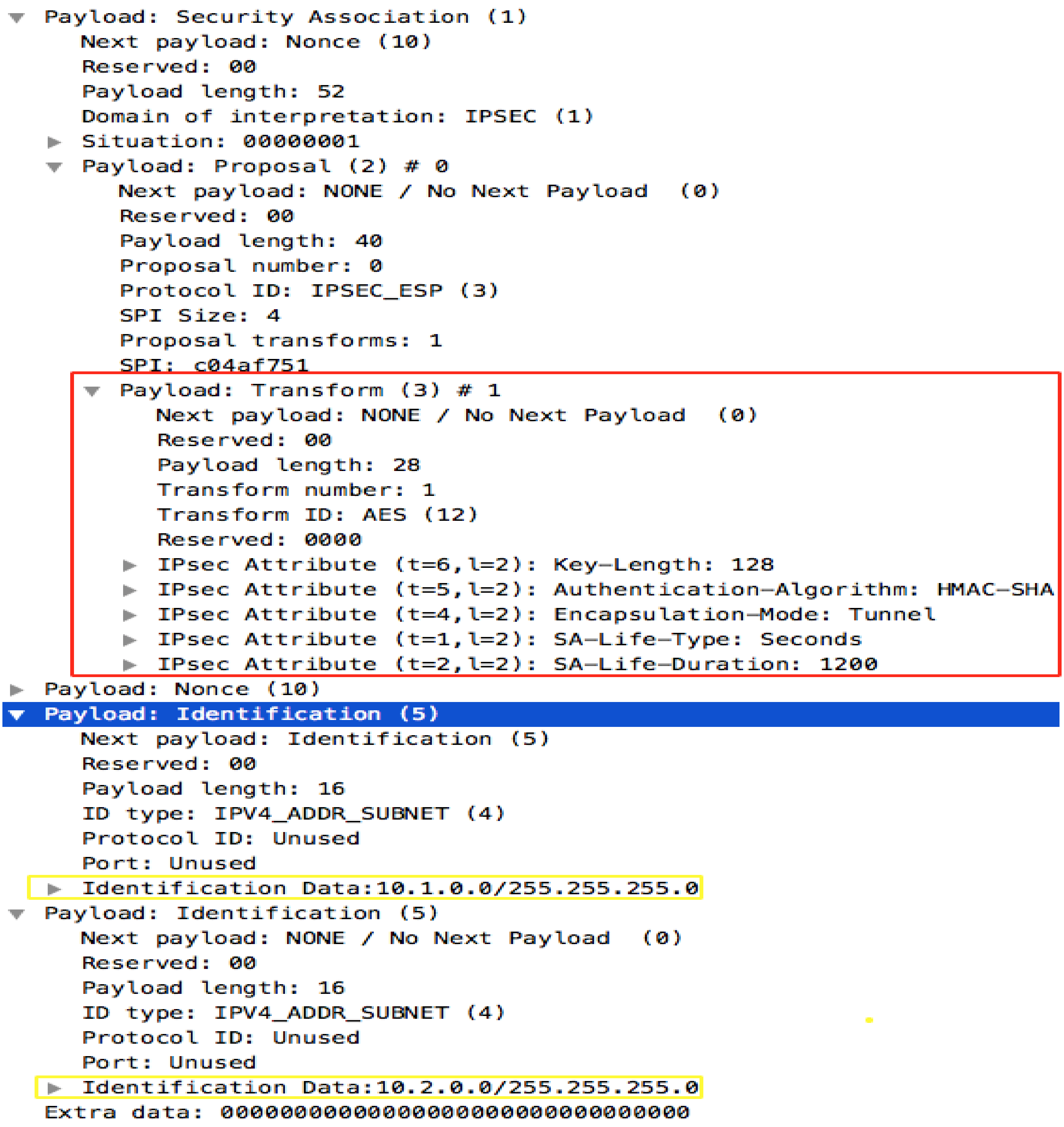
Now, Initiator sends its proposals to negotiate the security parameters for production traffic as mentioned (the highlighted yellow proposal is just a sample as the rest is collapsed - **this is frame #7**):



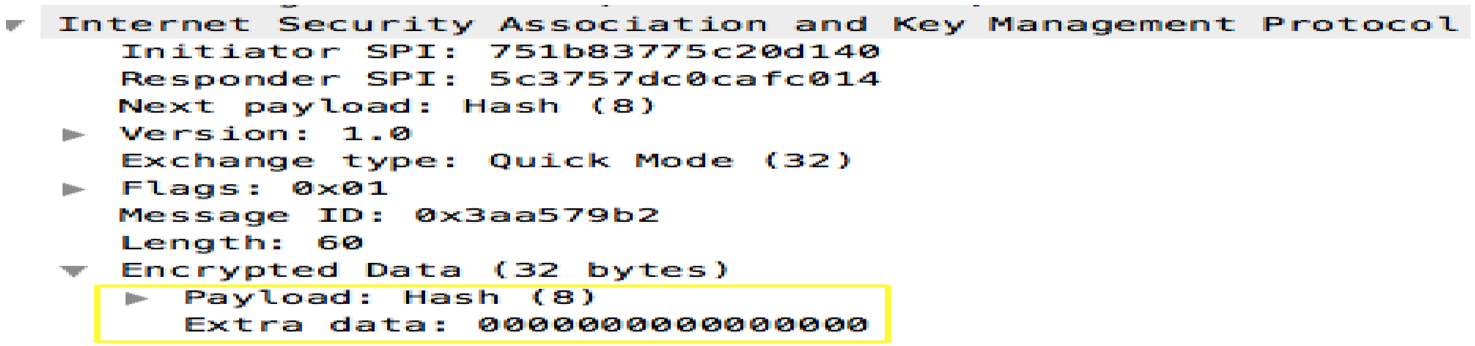
```
▼ Encrypted Data (208 bytes)
  ▶ Payload: Hash (8)
  ▼ Payload: Security Association (1)
      Next payload: Nonce (10)
      Reserved: 00
      Payload length: 104
      Domain of interpretation: IPSEC (1)
    ▶ Situation: 00000001
    ▼ Payload: Proposal (2) # 0
        Next payload: NONE / No Next Payload  (0)
        Reserved: 00
        Payload length: 92
        Proposal number: 0
        Protocol ID: IPSEC_ESP (3)
        SPI Size: 4
        Proposal transforms: 3
        SPI: ce38569e
      ▶ Payload: Transform (3) # 1
      ▶ Payload: Transform (3) # 2
      ▶ Payload: Transform (3) # 3
  ▶ Payload: Nonce (10)
  ▼ Payload: Identification (5)
      Next payload: Identification (5)
      Reserved: 00
      Payload length: 16
      ID type: IPV4_ADDR_SUBNET (4)
      Protocol ID: Unused
      Port: Unused
    ▶ Identification Data:10.1.0.0/255.255.255.0
  ▼ Payload: Identification (5)
      Next payload: NONE / No Next Payload  (0)
      Reserved: 00
      Payload length: 16
      ID type: IPV4_ADDR_SUBNET (4)
      Protocol ID: Unused
      Port: Unused
    ▶ Identification Data:10.2.0.0/255.255.255.0
    Extra data: 000000000000000000000000
```

```
▼ Payload: Transform (3) # 1
    Next payload: Transform (3)
    Reserved: 00
    Payload length: 28
    Transform number: 1
    Transform ID: AES (12)
    Reserved: 0000
  ▶ IPsec Attribute (t=6,l=2): Key-Length: 128
  ▶ IPsec Attribute (t=5,l=2): Authentication-Algorithm: HMAC-SHA
  ▶ IPsec Attribute (t=4,l=2): Encapsulation-Mode: Tunnel
  ▶ IPsec Attribute (t=1,l=2): SA-Life-Type: Seconds
  ▶ IPsec Attribute (t=2,l=2): SA-Life-Duration: 1200
```

Note: Identification payload carries source and destination tunnel IP addresses and if this doesn't match what is configured on both peers then IPSec negotiation will not proceed.
Then, in frame #8 we see that Responder picked one of the Proposals:



```
▼ Payload: Security Association (1)
    Next payload: Nonce (10)
    Reserved: 00
    Payload length: 52
    Domain of interpretation: IPSEC (1)
  ▶ Situation: 00000001
  ▼ Payload: Proposal (2) # 0
      Next payload: NONE / No Next Payload  (0)
      Reserved: 00
      Payload length: 40
      Proposal number: 0
      Protocol ID: IPSEC_ESP (3)
      SPI Size: 4
      Proposal transforms: 1
      SPI: c04af751
    ▼ Payload: Transform (3) # 1
        Next payload: NONE / No Next Payload  (0)
        Reserved: 00
        Payload length: 28
        Transform number: 1
        Transform ID: AES (12)
        Reserved: 0000
      ▶ IPsec Attribute (t=6,l=2): Key-Length: 128
      ▶ IPsec Attribute (t=5,l=2): Authentication-Algorithm: HMAC-SHA
      ▶ IPsec Attribute (t=4,l=2): Encapsulation-Mode: Tunnel
      ▶ IPsec Attribute (t=1,l=2): SA-Life-Type: Seconds
      ▶ IPsec Attribute (t=2,l=2): SA-Life-Duration: 1200
▶ Payload: Nonce (10)
▼ Payload: Identification (5)
    Next payload: Identification (5)
    Reserved: 00
    Payload length: 16
    ID type: IPV4_ADDR_SUBNET (4)
    Protocol ID: Unused
    Port: Unused
  ▶ Identification Data:10.1.0.0/255.255.255.0
▼ Payload: Identification (5)
    Next payload: NONE / No Next Payload  (0)
    Reserved: 00
    Payload length: 16
    ID type: IPV4_ADDR_SUBNET (4)
    Protocol ID: Unused
    Port: Unused
  ▶ Identification Data:10.2.0.0/255.255.255.0
  Extra data: 00000000000000000000000000000000
```



```
▼ Internet Security Association and Key Management Protocol
    Initiator SPI: 751b83775c20d140
    Responder SPI: 5c3757dc0cafc014
    Next payload: Hash (8)
  ▶ Version: 1.0
    Exchange type: Quick Mode (32)
  ▶ Flags: 0x01
    Message ID: 0x3aa579b2
    Length: 60
  ▼ Encrypted Data (32 bytes)
    ▶ Payload: Hash (8)
      Extra data: 0000000000000000
```

Frame #9 is just an ACK to the picked proposal confirming that Initiator accepted it:

Conclusion : Thus Ipsec protocol with AH and ESP header format is studied using Wireshark