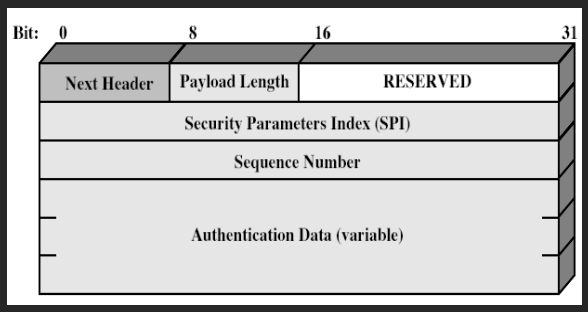
AH

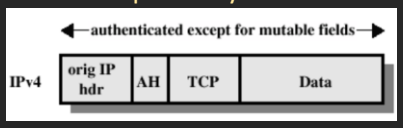


Next Header identifies the higher-level protocol that is using this protocol. *Payload Length* is self explaining. *RESERVED* bits. Then there is the *SPI (Security Parameter Index)*, we have already discussed it, about it I want to add a further consideration, if you remember we defined the SPI like something that was describing some technical details about the keys and other techniques used in order to implement the cryptography needed to ensure security. The way such bits are interpreted is not officially defined, so you can setup your scheme. *Sequence Numbers* is self explaining. *Authentication Data* is explaining informations about authentication, for example which hashing function and so on.

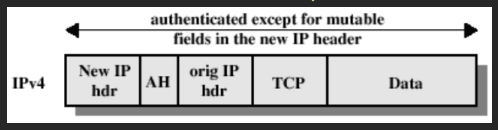
This protocol is protecting the payload, I mean it applies the MAC function on the whole payload, but also some part of the IP header. In particular only the non-mutable fields (we know that there are mutable and non-mutable fields, for example fields like the one that informs about fragmentation can vary packet to packet) are authenticated and secured. In this way AH can provide data integrity and data origin authentication (the IP address cannot be modified as this always invalidates the hash value). However, it is incompatible with NAT, because once the packet reacher the gateway, the gateway cannot change IP of destination in order to exchange it with the private one within its network, because this will invalidate the authentication.

AH operates on top of IP, using the protocol number *51*. Therefore when the packet is going over the internet it is seen as a normal datagram, when a generic router receives a packet in which in the IP Header in the IP Protocol field there is the number *51* it means that the router will have to pass the packet to the AH layer upperlying it, which will extract the AH header and that can process the real payload.

When AH is used on packets with *Transport Mode*, the outcome is a packet that has a *original IP header*, the *AH header* and finally the *payload* (which is composed by *TCP* header and the *Data*).



When AH is used in the *Tunnel Mode*, the original datagram (original *IP header, TCP header* and *Data)* becomes the new payload, then there is the *New IP header*, and between them there is the *AH header*.

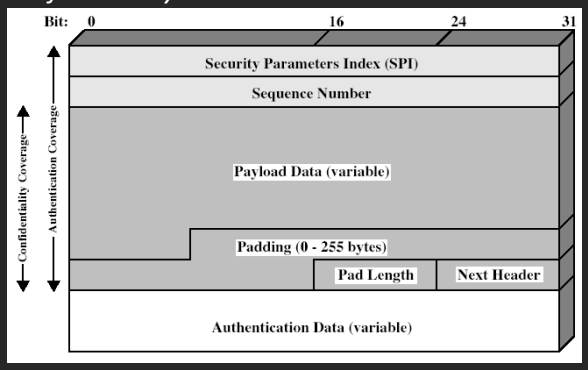


What happens is that here the whole original datagram is protected, so also the mutable fields of the IP header. But then there is the New IP header for which only the non-mutable fields are protected. So it can be used in such a way that I specify as destination in the new IP header the IP address of the gateway of my enterprise, and once the gateway checks the integrity of the whole packet, it can remove the new IP header and the AH header and use normally the original packet within the private network of the enterprise.

ESP

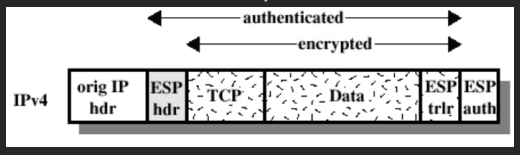
ESP is providing encryption, it is supporting all modern encrypting algorithms like AES, Triple-DES and so on, and often it is using CBC. If you are using ESP_a, the difference from AH is that when you use the authentication feature of ESP the authentication / data integrity is made only on the payload, not even the non-mutable fields of the IP header are included. This allows the ESP to be compatible with NAT. It means that AH is somewhat more inclusive then ESP_a for what concerns authentication. In some cases ESP provides also limited *traffic flow confidentiality*, traffic flow is something an attacker is interested to understand, he wants

to see the type of traffic outgoing your computer, you want to avoid this so you need *traffic flow confidentiality.*

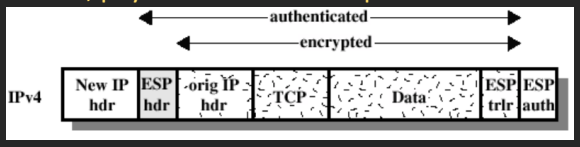


ESP header structure, again you can see SPI and Sequence Number, then there is some variable-length *Data* and the padding after it, then we find the pad length and the *next header* which describes what is the protocol that will elaborate these packets at higher level. At the end there is the *Authentication Data*. Notice that if you see this scheme you are reading a scheme composed by *header* and by *trailer.*

When ESP is used in *Transport Mode*, then the whole payload and the trailer is encrypted, and we have the orig IP hdr, the ESP hdr, the payload and the ESP auth portion in case of ESP_a:



As we can see, the ESP wraps the payload between an *header* and a *trailer*.
In the case of ESP_a the *ESP auth* will authenticate the datagram, in the case of ESP the *ESP auth* is missing.
When ESP is used in *Tunnel Mode*, the whole original datagram is encrypted, and so the final structure will be composed by New IP hdr, ESP hdr, payload and ESP auth portion in case of ESP_a:



The difference between Transport mode ESP and Tunnel mode ESP here is that with Transport mode we are encrypting only the payload, so the whole path of the data is visible, what is not visible is the information that our datagrams brings, while with Tunnel mode we are also encrypting the header, so the real destination and informations as offsets and message length will not be visible. We have already seen a usecase of ESP in Tunnel Mode (view previous pack of notes).

Anti-replay Service: it is a service that is contrasting the replay attacks. It is based ona sequence number, you have a counter that is initialized to 0, every time you send a packet the sender is incrementing the counter, and the value is stored on the sequence number field of the packet, technically speaking the counter is initialized to 1. Since the sequence number field is composed by 32 bit the sender must not allow the sequence number to cycle past $2^{32} - 1$ back to zero, otherwise replay attacks will be possible. Therefore there is a limit of the sequence number, when the connection is used for a long time maybe the limits is reached and what we need is to terminate the current SA and start a new one with new keys and starting again from 1. The standard implementation of this service is based on the fact that the receiver of the packet is using a windows of fixed size (there is also a proposal value given by the standard that is *W=64*), if the current packet falls out of the window on the left part it is old so it is wrong, if it falls out on the right the receiver just shift the window.

Combining Security Assotiations (SAs)

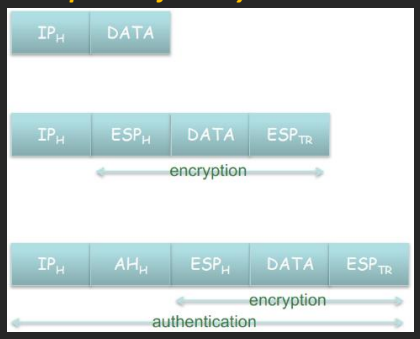
Every SA is implementing either AH or ESP, if you want to implement both of them you need to combine two SAs and the result of this combination is called **security bundle**. We have two type of security bundle:

- *transport adjacency*
- *iterating tunneling*

We will see these soon. Why we should want to combine security assotiations? Maybe we want authentication and confidentiality. Therefore we can think of using ESP_a or we can use ESP+AH. The first has

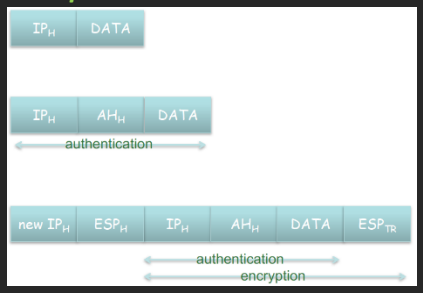
been already discussed. In the case we combine ESP and AH we can ask ourselves which should we apply first, and depending on this we are choosing *transport adjacency* or *iterating tunneling*.
The simplest case is ESP_a, which only uses one security assotiation (no combination of SAs). This is easy and we have already discussed this case. You can then setup a transport or a tunnel mode, and for both cases the authentication is applied on the ciphertext. But we can also have the two following bundles:

transport adjacency



here we are still assuming we want authentication after encryption. This is made by combining ESP and AH, so we combine two SAs. The inner SA is ESP, the outer SA is AH. This mean that the encryption is applied to the IP payload and so it is added an ESP header, and then all the result is authenticated, so an AH header is added. The advantage is that with this we are encrypting the data and we are authenticating the data but also some fields of the header.

transport tunnel



here the authentication is applied on the whole original datagram, so the AH header is added, and then we encrypt the whole outcoming datagram, so ESP header is added.

Does it make any sens to build bundles based on transport mode containing 3 security assotiations? Not at all. What is important is that there have been proposed 4 scenarios where we have to mix the security assotiations.

At one exam it has been asked the follwing question:
What security assotiation are best for the following cases:

- *Bob at home wants to print a pdf document on the LaserPrinter located in the private network of his corporate: in this case Bob needs only to connect to the Gateway of the corporate and he needs a tunnel of course (host-to-network communication), then the packet can travel in the private network of the corporate in an unprotected way. Indeed Bob can just ask for ESP_a in tunnel mode because (ESP guarantees that the document is encrypted till it reaches the gateway, and the authentication feature of ESP guarantees that the encrypted document will not be modified, then once the document reaches the gateway it can be sent in clear to Bob's printer since it will travel within the private network of Bob's corporate)*
- *Alice at home, she is the BOSS, and needs to save into her Personal Computer in the office a document that is confidential, so she needs more security assotiations, so once the packet reaches the gateway and the gateway extracts the payload, this outcome should be still protected, so she needs a SA between her and the gateway and a SA between her and her PC in the office, therefore she needs tunneling ESP for the first part (Alice → Gateway) and ESP for the second path (Gateway → PC). So she needs two level of encryption.*