# Tema 6: Protocolos de red seguros y redes privadas virtuales: Parte 1. Redes privadas virtuales

## Seguridad en Redes de Ordenadores

Departamento de Teoría de la Señal y Comunicaciones y Sistemas Telemáticos y Computación (GSyC)

Marzo de 2018

# Contenidos

# Contenidos

# Introducción

- Una VPN (Virtual Private Network) conecta dos subredes seguras a través de una red insegura, como Internet. Permite que las empresas puedan usar Internet para conectar sus oficinas remotas o sus trabajadores remotos de forma segura.
- Propiedades:
  - Encapsulado: los datos son encapsulados en otro protocolo que incluye una nueva cabecera de encaminamiento que permita atravesar la red insegura. El encapsulado en otro protocolo se denomina tunneling (tunelización).
  - Utilización de direccionamiento privado sobre una red pública insegura.
  - Autenticación: garantiza la identidad del origen del mensaje.
  - Confidencialidad: garantiza la confidencialidad de los datos que viajan a través de una red insegura.
  - Integridad: garantiza que los datos no han sido alterados por el camino.
  - Antirreproducción: detección de un ataque de reproducción.

# Topologías basadas en VPN

- **Host-host**: la VPN ofrece una conexión directa entre las dos máquinas.
- **Host-network**: la VPN ofrece una conexión directa entre una máquina y una subred. Resuelve la conexión de un trabajador remoto con la red de la empresa.
- **Network-network**: la VPN ofrece una conexión directa entre dos subredes. Resuelve la conexión entre 2 oficinas remotas.

# Ventajas/Inconvenientes

- Ventajas
  - Seguridad
  - Transparencia para los usuarios
  - Reducción de costes
- Inconvenientes
  - Configuración y gestión
  - Generación de claves
  - Problemas con NAT y firewalls.
  - Interoperabilidad de las implementaciones.

# Protocolos

- PPTP (Point-to-point Tunneling Protocol), RFC-2637. Desarrollado por un consorcio formado por Microsoft. Protocolo no seguro (2 días).
- L2TP (Layer 2 Tunneling Protocol), RFC 2661, incluye todas las características de PPTP y Cisco L2F (Layer 2 Forward). Nueva versión L2TPv3 (RFC-3931)
- IPSec es un protocolo de nivel de red creado por el IETF que puede enviar datos cifrados para redes IP. Requiere modificaciones en el nivel IP. Muy flexible, muy complejo.
- SSLv3/TLSv1 (Secure Sockets Layer/Transport Layer Security). Protocolo que se encuentra entre el nivel de aplicación y el nivel de transporte. Lo usan las aplicaciones para asegurar sus comunicaciones:
  - HTTPS, SMTPS, FTPS, OpenVPN, etc.

# Contenidos

# OpenVPN

- Open Source creado en 2002 para construir VPN entre subredes que puede utilizar dos modelos de seguridad diferentes:
  - protocolos SSL/TLS (negociación de clave compartida, dinámicamente)
  - claves compartidas previamente (*pre-shared-key*).
- Envía los datos cifrados a través de una red insegura, utilizando para ello una comunicación UDP o TCP, puerto por defecto 1194.
  - TCP es conveniente en ciertos entornos pero los algoritmos de control de congestión pueden afectar a la eficiencia de la transmisión.
- Muy fácil de instalar.
- Funciona con el paradigma cliente/servidor.
- Puede crear túneles de nivel de enlace o de nivel de red.

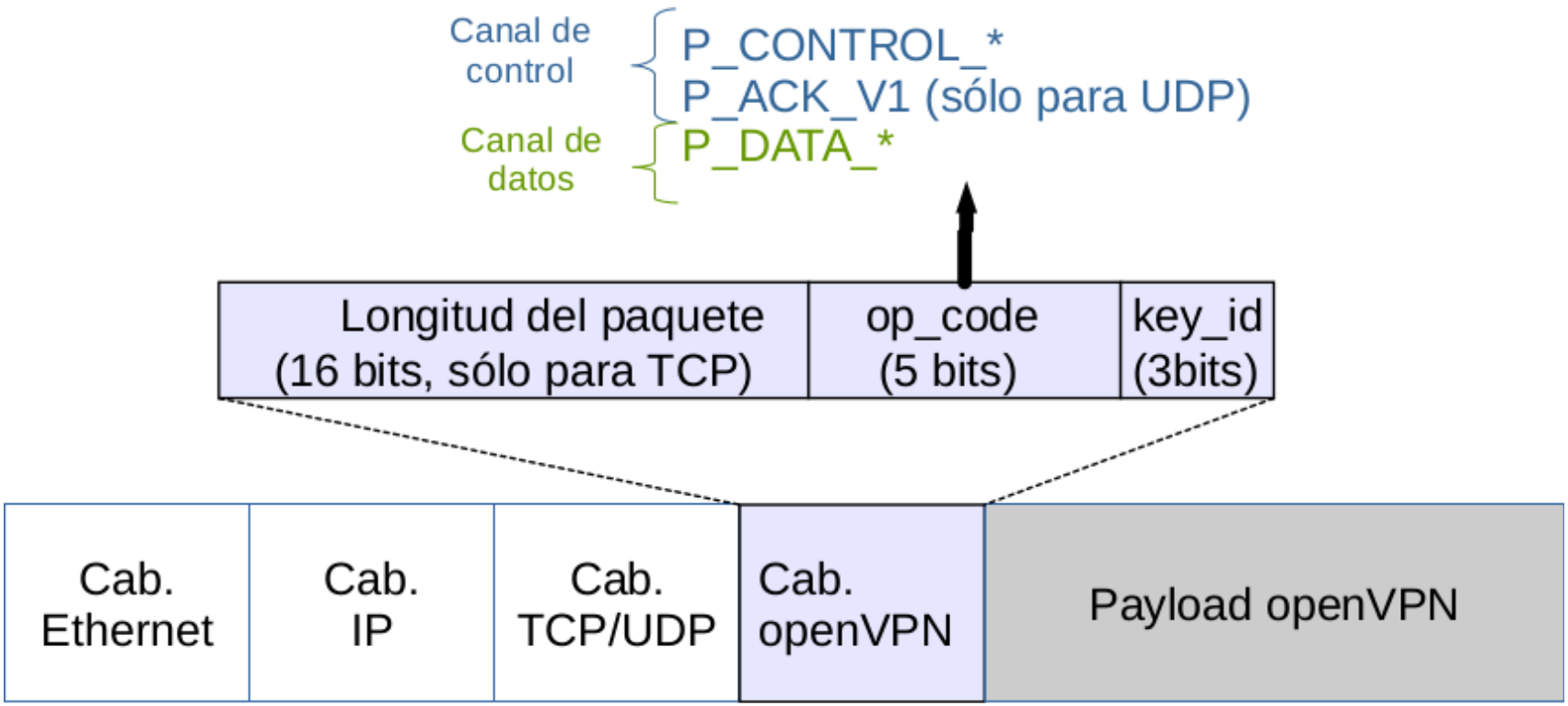
# Sesión OpenVPN

- Dentro de la misma sesión OpenVPN establece dos canales diferentes de comunicación:
  - **Canal de control**:
    - **Establecimiento de la comunicación SSL/TLS**, intercambio de claves, establecimiento de los algoritmos de seguridad, autenticación, etc.
    - El establecimiento de la comunicación SSL/TLS requiere un protocolo **fiable**. Si openVPN usa UDP, se introducen mensajes ACK para confirmar la recepción de los mensajes.
    - Los mensajes utilizados en el canal de control son `P_CONTROL_*` (en el caso de usar UDP, además hay asentimientos `P_ACK`).
  - **Canal de datos**:
    - Se utiliza una vez establecida la comunicación SSL/TLS, usará mensajes UDP sin asentimientos.
    - Los mensajes utilizados son `P_DATA` que van cifrados.

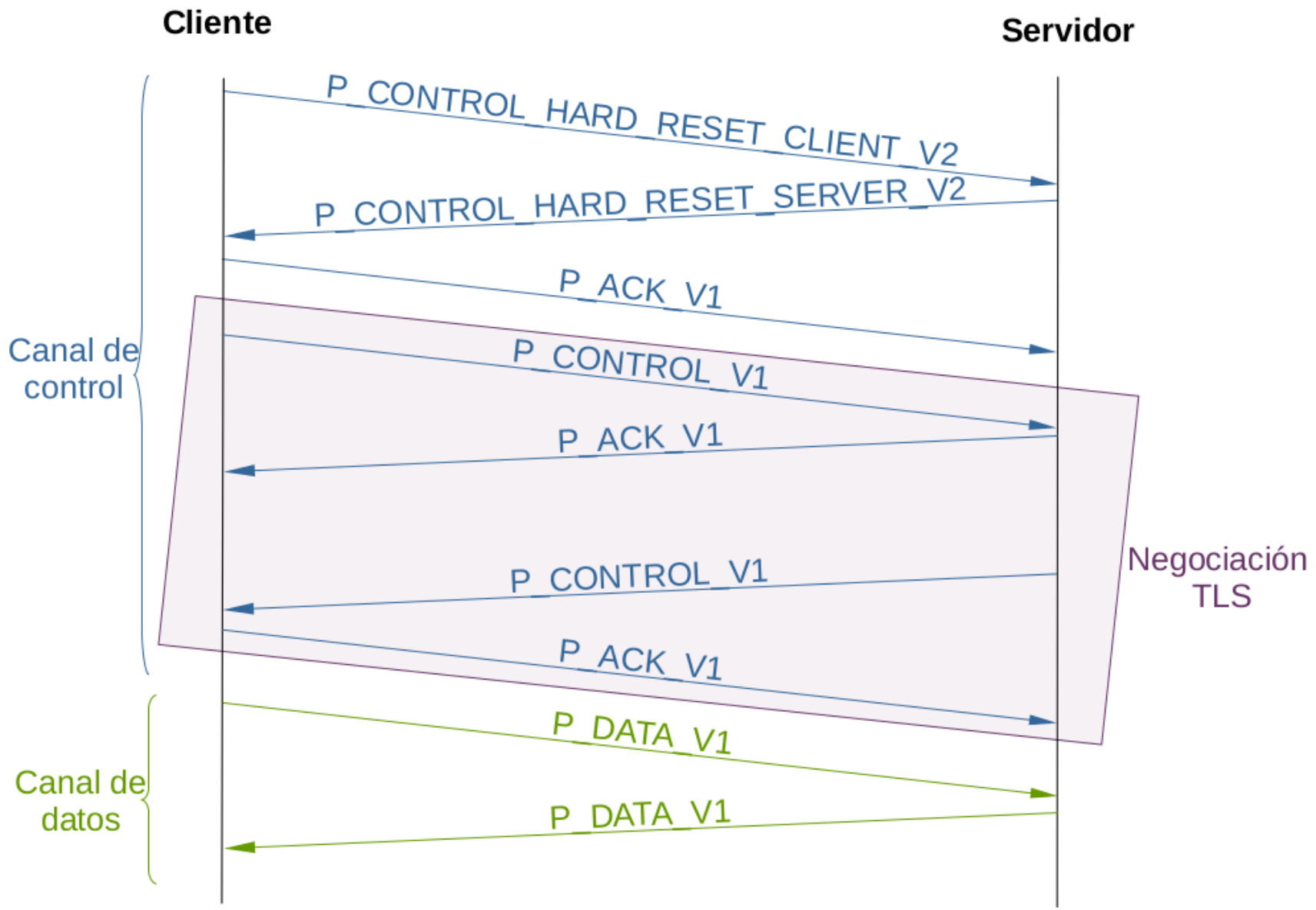
# Contenidos

# Formato de mensajes OpenVPN

- OpenVPN introduce una cabecera en todos los mensajes que contiene los siguientes campos:
  - Longitud del paquete (16 bits) sólo en TCP.
  - `opcode` (5 bits) sólo en el modo TLS. Es el tipo de mensaje.
  - `key_id` (3 bits) sólo en el modo TLS. Código para la sesión TLS negociada. La sesión TLS se renegocia usando un nuevo `key_id`.



# Ejemplo de establecimiento de sesión openVPN



# Tipo de mensajes OpenVPN (opcode)

- Tipos de mensajes en el **canal de control**:

| | |
|---|---|
| P_CONTROL_HARD_RESET_CLIENT_V1/ P_CONTROL_HARD_RESET_CLIENT_V2 | Método 1/Método 2 de clave, establecimiento de identificador de sesión TLS desde el cliente |
| P_CONTROL_HARD_RESET_SERVER_V1/ P_CONTROL_HARD_RESET_SERVER_V2 | Método 1/Método 2 de clave, establecimiento de identificador de sesión TLS desde el servidor |
| P_CONTROL_SOFT_RESET_V1 | Establecimiento de una renovación de sesión TLS, hay una ventana en la que se usarán tanto la antigua como la nueva |
| P_CONTROL_V1 | Establecimiento de una sesión TLS |
| P_ACK_V1 | Asentimiento de los mensajes de `P_CONTROL_V1`, si se utiliza como protocolo de transporte UDP. |

- Tipos de mensajes en el **canal de datos**:

| | |
|---|---|
| P_DATA_V1 | Datos cifrados en el túnel |
| P_DATA_V2 | Datos cifrados en el túnel y la identificación de los extremos |

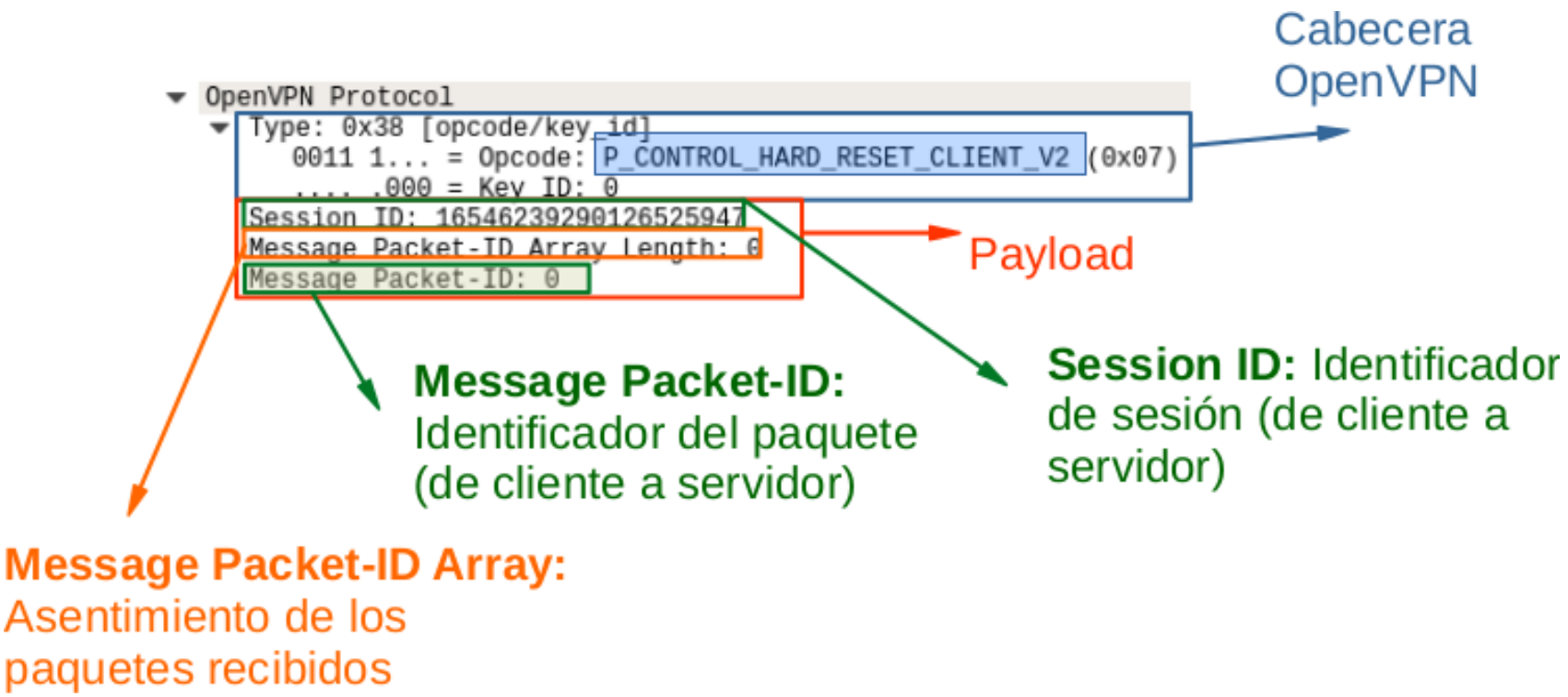
# Intercambio de mensajes OpenVPN

- Inicialmente se intercambian los mensajes para establecer el identificador de sesión TLS desde el cliente y el servidor.
  - P_CONTROL_HARD_RESET_CLIENT_V1/_V2
  - P_CONTROL_HARD_RESET_SERVER_V1/_V2

  OpenVPN implementa **2 métodos de claves**:
  - Método 1: deriva claves aleatorias a partir de la función `rand_bytes()`.
  - Método 2: mezcla información aleatoria de los dos extremos de la conexión usando la función TLS PRF. Éste es el método por defecto para OpenVPN 2.0.
- Estos mensajes deben ir asentidos desde cada extremo: P_ACK_V1
- A continuación se establecen los parámetros de seguridad en el establecimiento de la sesión TLS a través de los mensajes P_CONTROL_V1 que deberán ir asentidos P_ACK_V1.
  - Los mensajes `P_CONTROL_V1` contendrán la información de la sesión TLS donde se intercambiarán los mensajes `Client Hello`, `Server Hello`, intercambio de claves y certificados, especificación de algoritmos de cifrado.
- Una vez establecida la configuración de seguridad a través de TLS se pueden enviar/recibir datos cifrados a través del túnel, utilizando el canal de datos: P_DATA

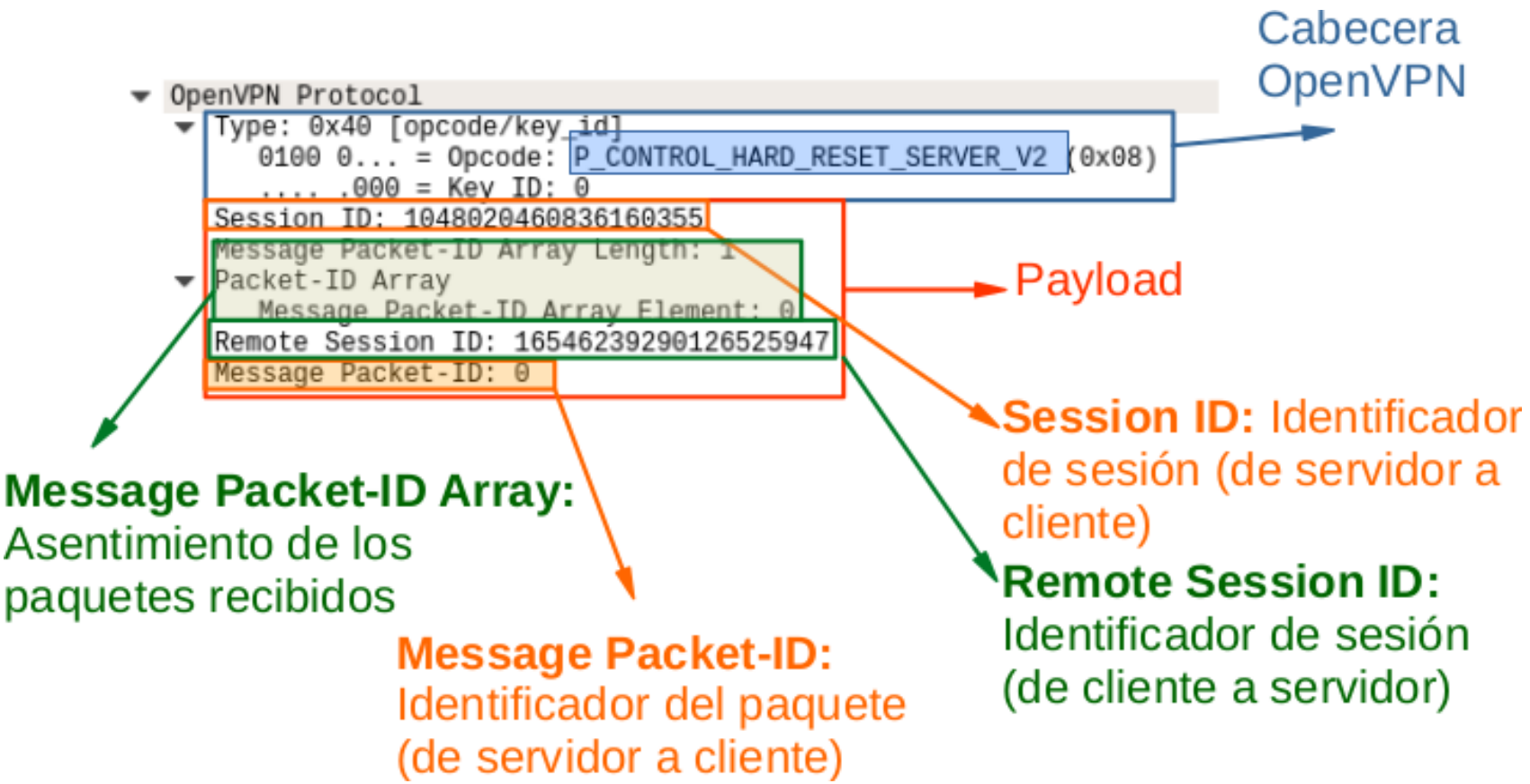
# Formato de mensajes P_CONTROL_HARD_RESET_CLIENT_V2

- Mensaje del cliente al servidor para iniciar el canal de control OpenVPN.



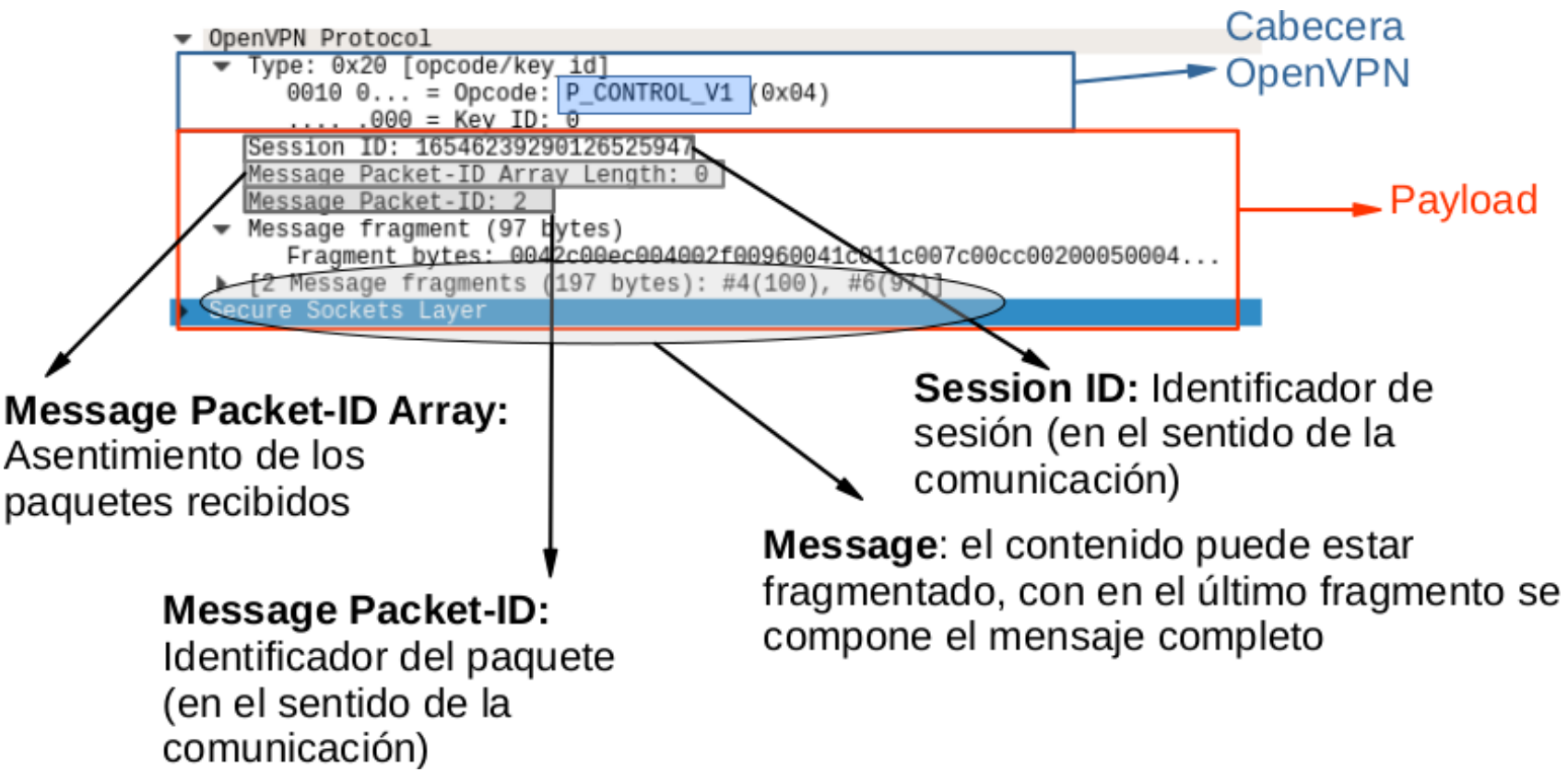
# Formato de mensajes P_CONTROL_HARD_RESET_SERVER_V2

- Mensaje del servidor al cliente como respuesta a `P_CONTROL_HARD_RESET_CLIENT_V2`.



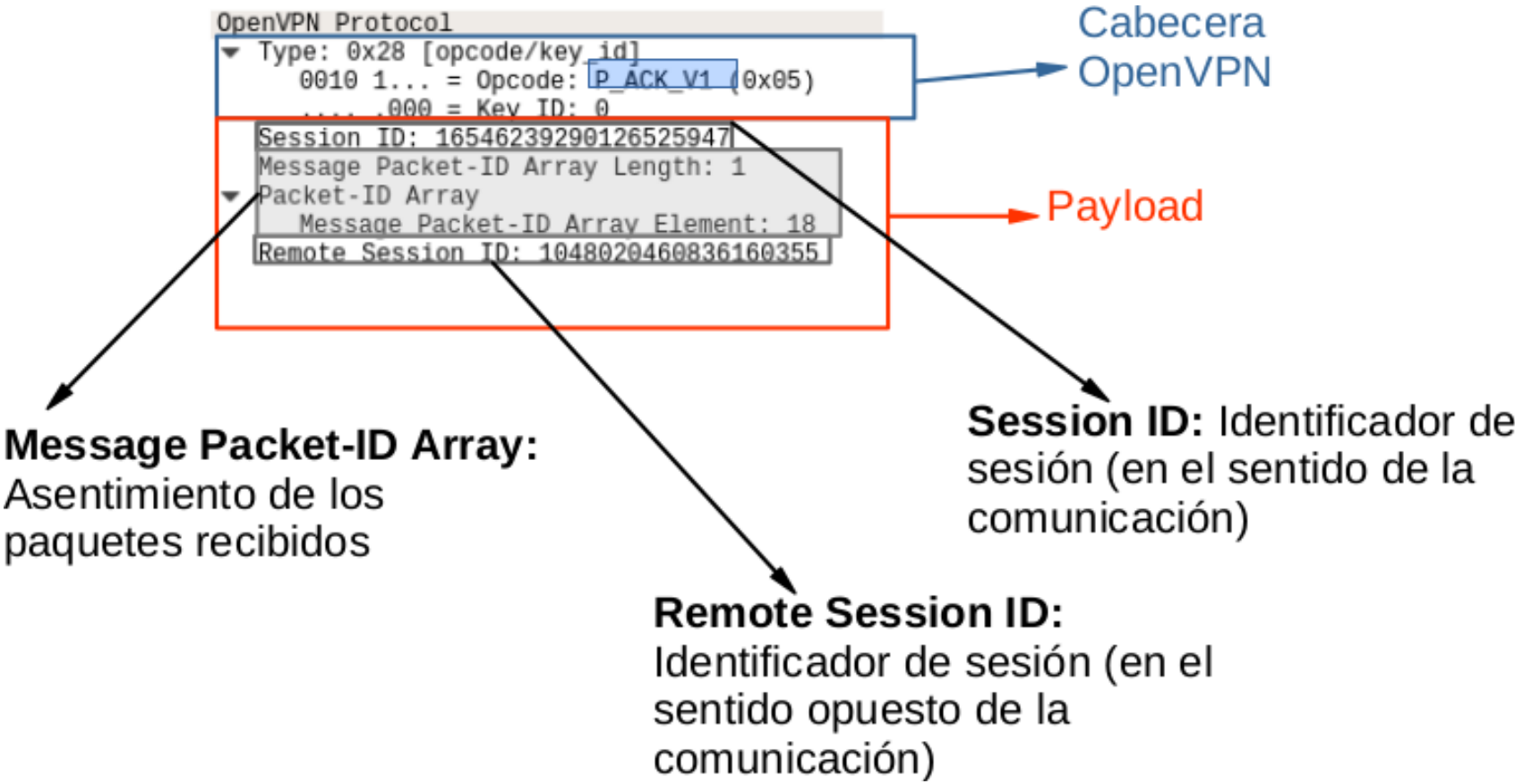
# Formato de mensajes P_CONTROL_V1

- Mensaje en ambos sentidos para establecer los parámetros de seguridad del canal de datos, en particular, establecer la sesinó TLS.



# Formato de mensajes P_ACK_V1

- Mensaje en ambos sentidos para asentir los mensajes `P_CONTROL_V1`.



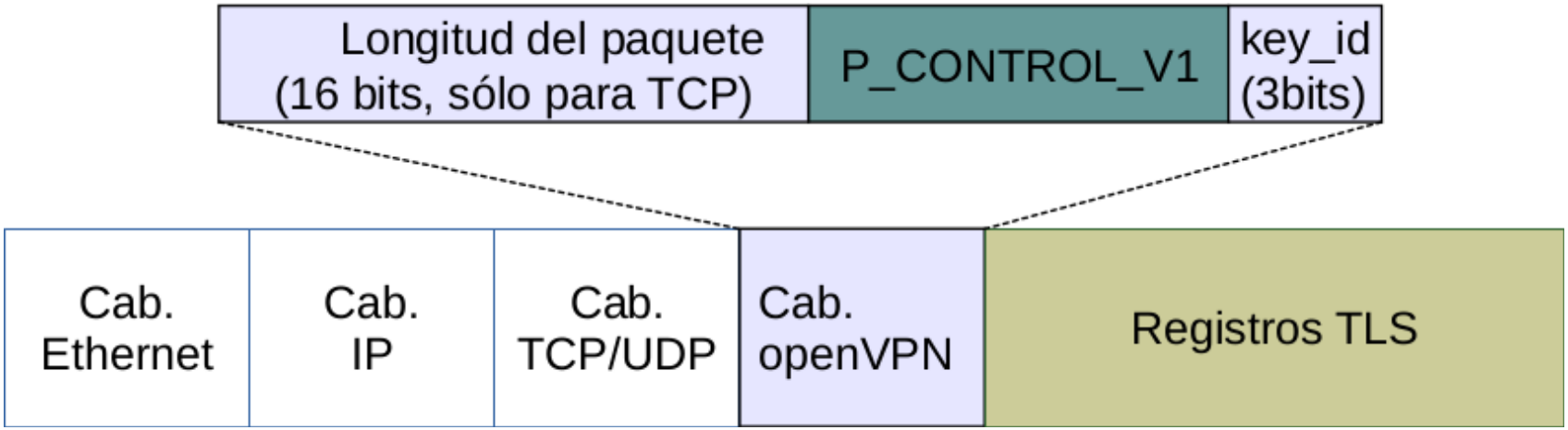
# Campos relevantes de los mensajes

- `Local session_id` (8 bytes):
  número aleatorio para identificar la sesión OpenVPN local.
- `Message packet-id array length` (1 byte):
  longitud del campo `Message packet-id array`.
- `Packet-id array`:
  identificadores de paquetes recibidos y que se están asintiendo.
- `Remote session_id` (8 bytes):
  número de identificador de la sesión OpenVPN remota.
- `Message packet-id` (4 bytes):
  Identificador de paquete, sólo en los mensajes de control (para que después puedan ser asentidos por el otro extremo).

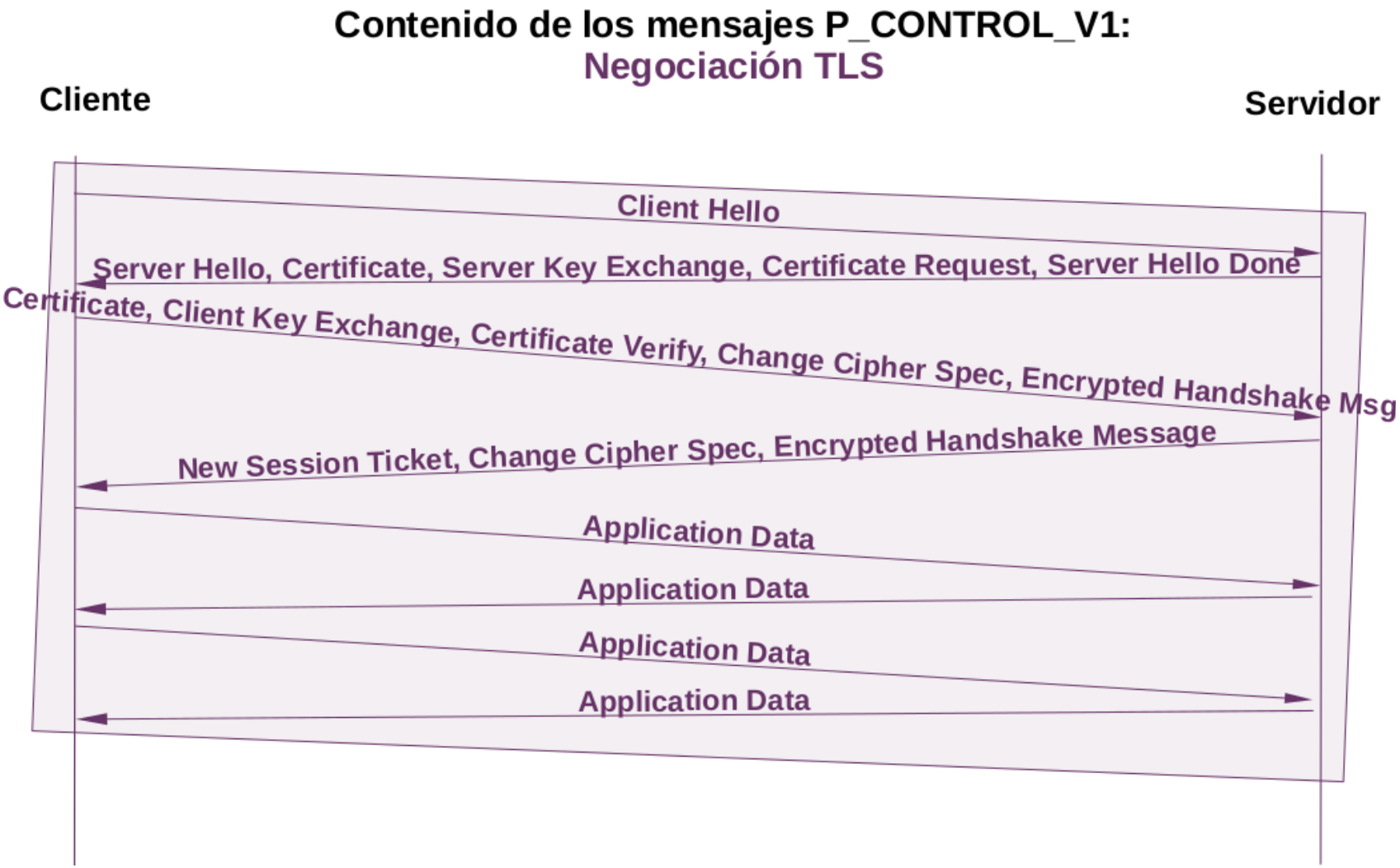
# Contenidos

# Establecimiento de la sesión TLS

- La sesión TLS se establece utilizando mensajes OpenVPN del tipo `P_CONTROL_V1` que contienen los registros TLS que permiten establecer los parámetros de seguridad entre cliente y servidor.



# Ejemplo de establecimiento de sesión TLS



# Registros TLS

- **Client Hello**: el cliente envía una lista de los algoritmos de cifrado soportados, lista de métodos de compresión y extensiones. Se adjunta un identificador de sesión `SessionId`.
- **Server Hello**: el servidor responde con los algoritmos de seguridad elegidos (cipher suite)
- **Certificate**: se envía la cadena de certificados necesarios para autenticar a los extremos.
- **ServerKeyExchange/ClientKeyExchange**: parámetros Diffie Hellman para calcular el secreto compartido.
- **Certificate Request**: el servidor solicita al cliente su certificado para autenticarlo.
- **Server Hello Done**: indica que el servidor ha enviado todos los datos de su negociación.
- **Certificate Verify**: el cliente demuestra que posee la clave privada de su certificado, contiene información firmada por el cliente.
- **Change Cipher Spec**: notifica al que lo recibe, que la siguiente información (Encrypted Handshake Message) será cifrada con las claves calculadas.
- **Encrypted Handshake Message**: mensaje cifrado con las claves calculadas previamente.

# Cipher suite en TLS

- Es un conjunto de algoritmos que ayudan a proporcionar seguridad a una comunicación que usa TLS:
  - Algoritmo para intercambio de claves
  - Algoritmo para la autenticación de extremos
  - Algoritmo para el cifrado de datos
  - Algoritmo para la autenticación de mensajes (MAC, Message Authentication Code), es decir, verificar su integridad.
- Ejemplo: `TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA`
  - TLS: protocolo para el que se ha definido este cipher suite.
  - DHE: algoritmo para el intercambio de claves
  - RSA: algoritmo para autenticar al cliente y al servidor durante el establecimiento de la sesión TLS.
  - 3DES_EDE_CDC: algortimo de cifrado de datos, usando la clave acordada previamente.
  - SHA: algoritmo par verificar la integridad de los mensajes.

# Perfect Forward Secrecy (PFS)

- PFS es una propiedad del mecanismo de intercambio de claves, presente en el protocolo TLS.
- PFS es la independencia entre la clave de sesión generada durante el establecimiento de la sesión TLS y las claves RSA (clave pública/clave privada) que normalmente tienen un tiempo de vida largo.
- Con cada sesión TLS se generan claves de sesión diferentes para cifrar la información. Mayor seguridad ya que si se compromete una clave de sesión, sólo afecta a dicha sesión.
- Veremos como DHE (Diffie-Hellman Ephemeral) puede implementar PFS en TLS.

# Server Key Exchange con DHE

- El servidor genera x (parte privada) y calcula $X = g^x modp$, el servidor envía al cliente:
  - longitud de p y su valor
  - longitud de g y su valor
  - longitud de X (la clave pública) y su valor
  - longitud de la firma y su valor (utilizando la clave privada del servidor).

```
▼ Handshake Protocol: Server Key Exchange
    Handshake Type: Server Key Exchange (12)
    Length: 777
  ▼ Diffie-Hellman Server Params
      p Length: 256
      p: 924ebc428454890676d2a7fd547f63306bbd5e66a4a3481f...
      g Length: 1
      g: 02
      Pubkey Length: 256
      Pubkey: 148c8e719fd029a9bc0ed3a3242c673835b54d7ba61e17cc...
      Signature Length: 256
      Signature: 00317e6cd16f71855b0b678cfbf23839192562653f62b89a...
```

# Client Key Exchange con DHE

- El cliente genera y (parte privada) y calcula $Y = g^y \, mod p$, con los valores p y q que le ha enviado el servidor. El cliente envía:
  - longitud de Y (la clave pública) y su valor

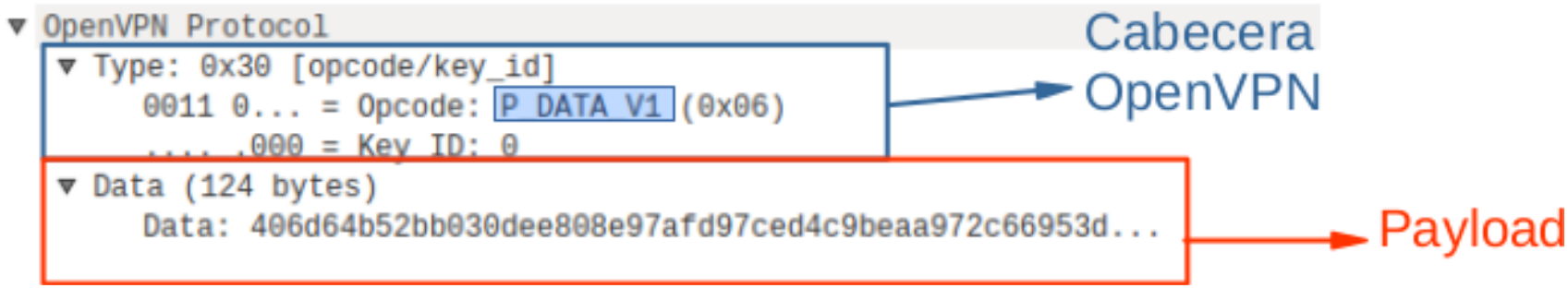
```
TLSv1 Record Layer: Handshake Protocol: Client Key Exchange
   Content Type: Handshake (22)
   Version: TLS 1.0 (0x0301)
   Length: 262
▼ Handshake Protocol: Client Key Exchange
      Handshake Type: Client Key Exchange (16)
      Length: 258
   ▼ Diffie-Hellman Client Params
         Pubkey Length: 256
         Pubkey: 6378efc0b9aa3230ebb175f2416f20d45b176dcbcd75f60a...
```

# Cliente y Servidor pueden calcular un secreto compartido

- **Pre-master secret**: Después del intercambio de los valores X e Y, cliente y servidor pueden calcular un secreto compartido.
  - El cliente calcula: $(X)^y \, modp = g^{xy} \, modp$
  - El servidor calcula: $(Y)^x \, modp = g^{xy} \, modp$
- **Master secret**: calculado a través de la función PRF (Pseudo Random Fuction) que toma como entrada: pre-master secret y números aleatorios generados por el cliente y el servidor.
- **Claves**: se calculan a partir de master secret: claves para cifrado de cliente y de servidor, claves para integridad de cliente y de servidor y claves IV de cliente y de servidor (para DES CBC).

# Contenidos

# Ejemplo de mensaje P_DATA_V1




- Algunos mensajes `P_DATA_V1` llevan información de control, se utilizan como mensajes `keepalive`, consulta de MTU, etc.
- OpenVPN establece que no puede pasar mucho tiempo sin que se envíen mensajes `P_DATA`. Dependiendo de la configuración, si pasa mucho tiempo puede desactivarse la configuración del túnel.

# Contenidos

# Bibliografía


- VPNs Illustrated, Tunnels, VPNs and IPSec, Jon C. Snader, Addison-Wesley.
- OpenVPN Security Overview (https://openvpn.net/index.php/open-source/documentation/security-overview.html)