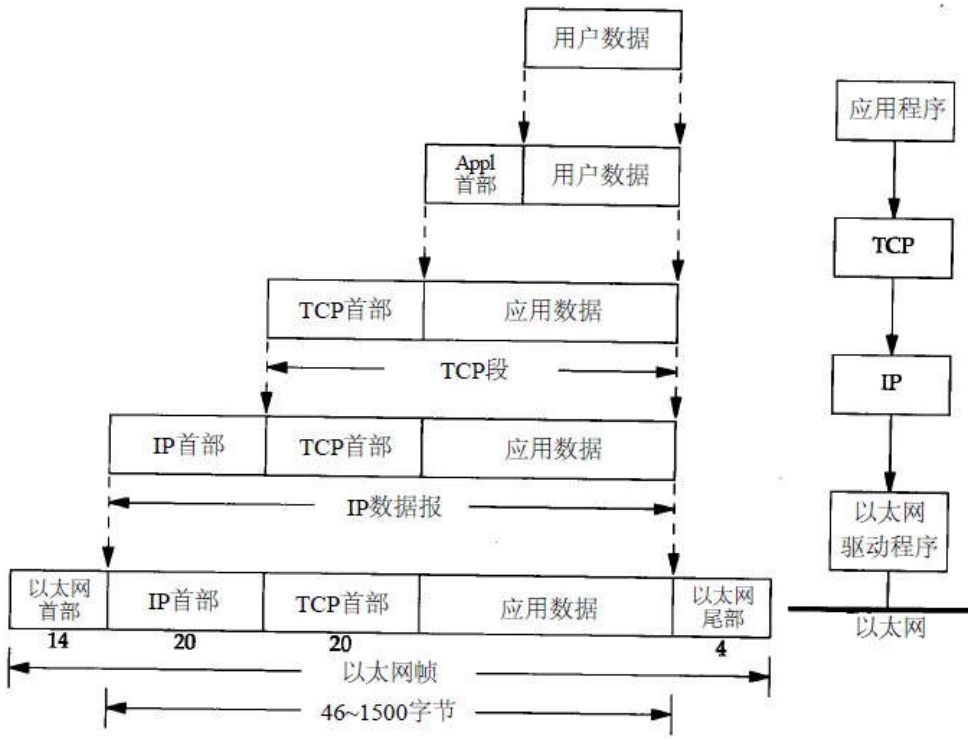
数据包结构分析

通过 wireshark 抓取在不同链路上的数据包，分析数据在网上传输过程。首先要有下面基础知识。

1,网络数据封装过程，数据包发送的时候从上往下封装的，解封装反过来。



从下往上看

最下面是以太网帧，位于 osi 参考模型的数据链路层。该层帧格式有以太网帧（常用），802.2/802.3 帧和 ppp 帧。

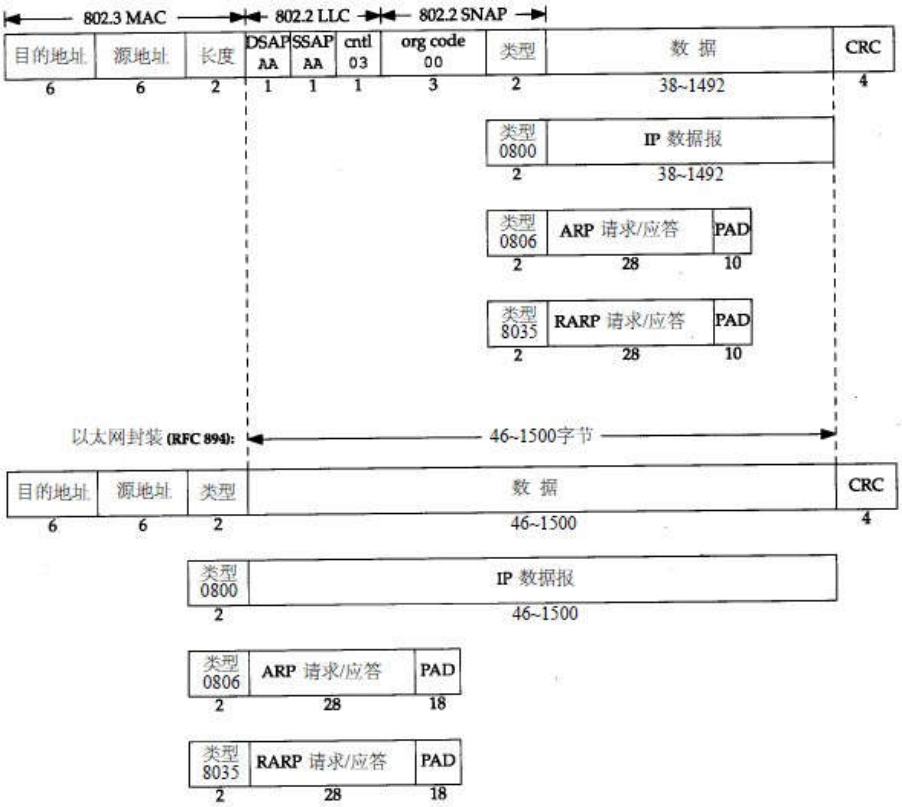
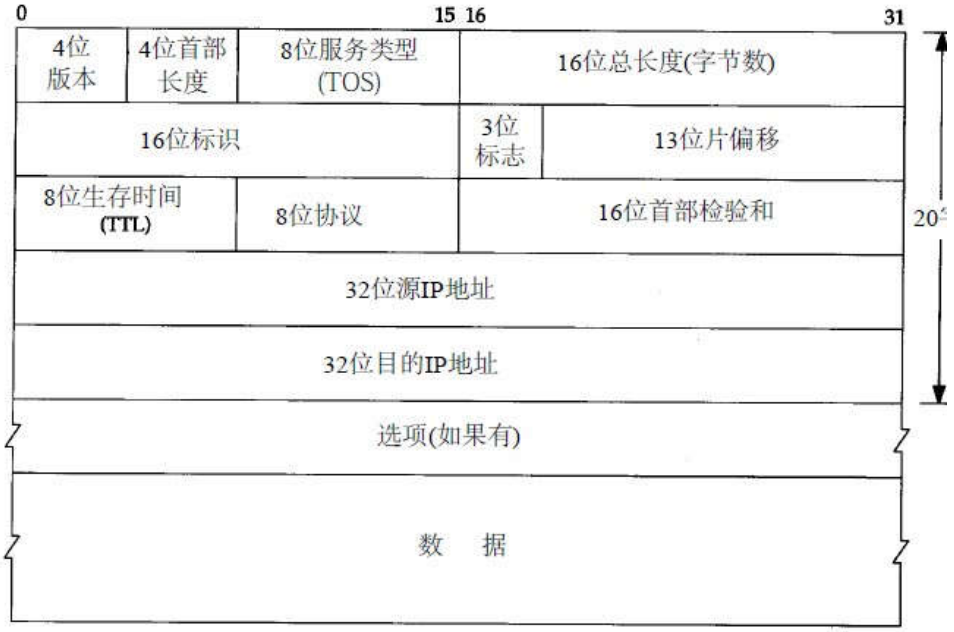
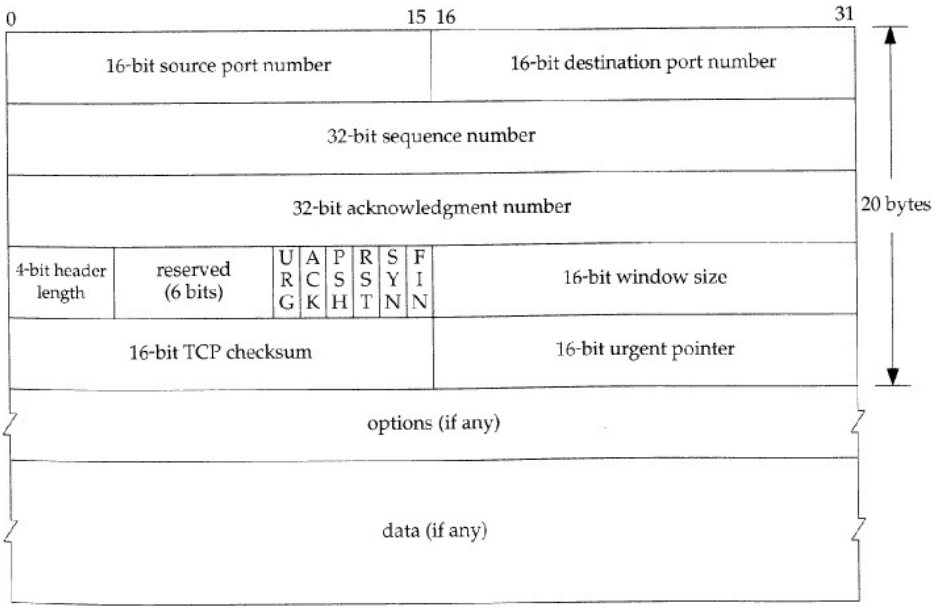


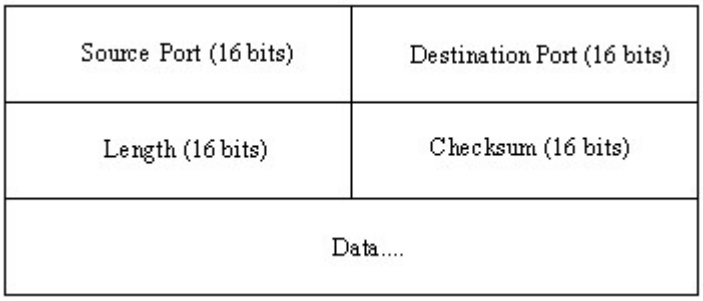
图2-1 IEEE 802.2/802.3（RFC 1042）和以太网的封装格式（RFC 894）

对应网络层，主要协议有 ip，ICMP，IGMP。如下面的 ip 数据包头

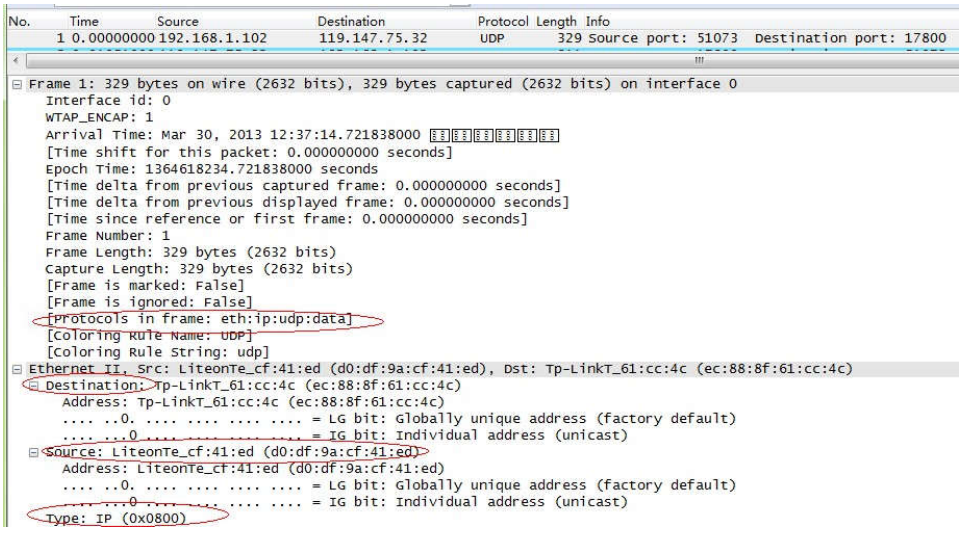


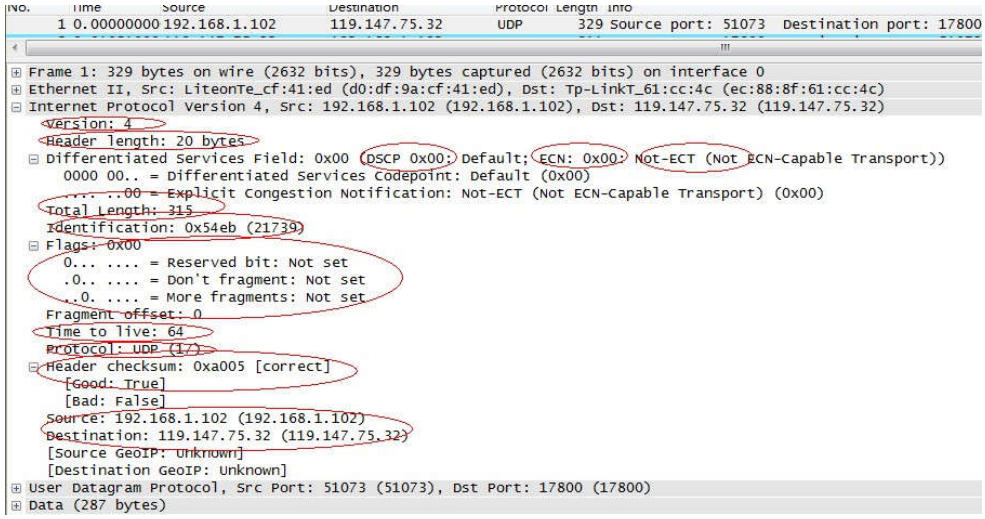
对于传输层，主要的协议就是 TCP（6）和 UDP（17）。

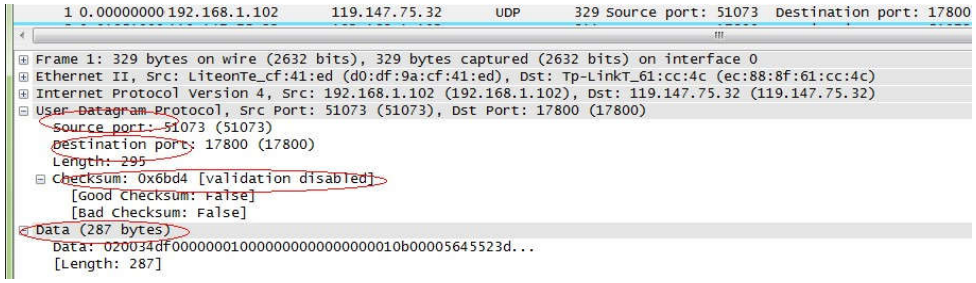

0
15 16
31
16-bit source port number
16-bit destination port number
32-bit sequence number
32-bit acknowledgment number
20 bytes
4-bit header length
reserved (6 bits)
URG
ACK
PSH
RST
SYN
FIN
16-bit window size
16-bit TCP checksum
16-bit urgent pointer
options (if any)
data (if any)


Source Port (16 bits)
Destination Port (16 bits)
Length (16 bits)
Checksum (16 bits)
Data....

第一个包(udp 包)


No. Time Source Destination Protocol Length Info
1 0.00000000 192.168.1.102 119.147.75.32 UDP 329 Source port: 51073 Destination port: 17800
Frame 1: 329 bytes on wire (2632 bits), 329 bytes captured (2632 bits) on interface 0
Interface id: 0
WTAP_ENCAP: 1
Arrival Time: Mar 30, 2013 12:37:14.721838000
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1364618234.721838000 seconds
[Time delta from previous captured frame: 0.000000000 seconds]
[Time delta from previous displayed frame: 0.000000000 seconds]
[Time since reference or first frame: 0.000000000 seconds]
Frame Number: 1
Frame Length: 329 bytes (2632 bits)
Capture Length: 329 bytes (2632 bits)
[Frame is marked: False]
[Frame is ignored: False]
[Protocols in frame: eth:ip:udp:data]
[Coloring Rule Name: UDP]
[Coloring Rule String: udp]
Ethernet II, Src: LiteonTe_cf:41:ed (d0:df:9a:cf:41:ed), Dst: Tp-LinkT_61:cc:4c (ec:88:8f:61:cc:4c)
Destination: Tp-LinkT_61:cc:4c (ec:88:8f:61:cc:4c)
Address: Tp-LinkT_61:cc:4c (ec:88:8f:61:cc:4c)
.... ..0. .... = LG bit: Globally unique address (factory default)
....0 .... = IG bit: Individual address (unicast)
Source: LiteonTe_cf:41:ed (d0:df:9a:cf:41:ed)
Address: LiteonTe_cf:41:ed (d0:df:9a:cf:41:ed)
.... ..0. .... = LG bit: Globally unique address (factory default)
....0 .... = IG bit: Individual address (unicast)
Type: IP (0x0800)





这是有我本机发出的 upd 包

附：ip 头字段说明

版本号（Version）：长度 4 比特。标识目前采用的 IP 协议的版本号。一般的值为 0100（IPv4），0110（IPv6）

IP 包头长度（HeaderLength）：长度 4 比特。这个字段的作用是为了描述 IP 包头的长度，因为在 IP 包头中有变长的可选部分。该部分占 4 个 bit 位，单位为 32bit（4 个字节），即本区域值= IP 头部长度（单位为 bit）/(8*4)，因此，一个 IP 包头的长度最长为“1111”，即 15*4 = 60 个字节。IP 包头最小长度为 20 字节。

服务类型（Type of Service）：长度 8 比特。8 位 按位被如下定义

PPP D T R C 0

但是 TOS 字段已经作为区分服务(Diffsrv)架构一部分被重新定义了，开始的 6 位构成区分服务代码点（DiffServ Code Piont，DSCP)，利用这 6 位可以定义 64 个不同的服务类别。

ECN 显式拥塞通知

IP 包总长（Total Length)：长度 16 比特。 以字节为单位计算的 IP 包的长度 (包括头部和数据)，所以 IP 包最大长度 65535 字节。

标识符（Identifier）(数据报 ID）:长度 16 比特。该字段和 Flags 和 Fragment Offest 字段联合使用，对大的上层数据包进行分段（fragment）操作。路由器将一个包拆分后，所有拆分开的小包被标记相同的值，以便目的端设备能够区分哪个包属于被拆分开的包的一部分。

标记（Flags)：长度 3 比特。该字段第一位不使用。第二位是 DF（Don't Fragment) 位，DF 位设为 1 时表明路由器不能对该上层数据包分段。如果一个上层数据包无法在不分段的情况下进行转发，则路由器会丢弃该上层数据包并返回一个错误信息。第三位是 MF（More Fragments）位，当路由器对一个上层数据包分段，则路由器会在除了最后一个分段的 IP 包的包头中将 MF 位设为 1。

片偏移（Fragment Offset)：长度 13 比特。表示该 IP 包在该组分片包中位置，接收端靠此来组装还原 IP 包。

生存时间（TTL)：长度 8 比特。当 IP 包进行传送时，先会对该字段赋予某个特定的值。当 IP 包经过每一个沿途的路由器的时候，每个沿途的路由器会将 IP 包的 TTL 值减少 1。如果 TTL 减少为 0，则该 IP 包会被丢弃。这个字段可以防止由于路由环路而导致 IP 包在网络中不停被转发。

协议（Protocol）：长度 8 比特。标识了上层所使用的协议。

以下是比较常用的协议号：

1 ICMP

2 IGMP

6 TCP

17 UDP

88 IGRP

89 OSPF

头部校验（Header Checksum）：长度 16 位。用来做 IP 头部的正确性检测，但不包含数据部分。 因为每个路由器要改变 TTL 的值,所以路由器会为每个通过的数据包重新计算这个值。

起源和目标地址（Source and Destination Addresses）:这两个地段都是 32 比特。标识了这个 IP 包的起源和目标地址。要注意除非使用 NAT，否则整个传输的过程中，这两个地址不会改变。

可选项（Options）：这是一个可变长的字段。该字段属于可选项，主要用于测试，由起源设备根据需要改写。可选项目包含以下内容：

松散源路由（Loose source routing）：给出一连串路由器接口的 IP 地址。IP 包必须沿着这些 IP 地址传送，但是允许在相继的两个 IP 地址之间跳过多个路由器。

严格源路由（Strict source routing）：给出一连串路由器接口的 IP 地址。IP 包必须沿着这些 IP 地址传送，如果下一跳不在 IP 地址表中则表示发生错误。

路由记录（Record route）：当 IP 包离开每个路由器的时候记录路由器的出

站接口的 IP 地址。

时间戳（Timestamps）：当 IP 包离开每个路由器的时候记录时间。

填充(Padding)：因为 IP 包头长度(Header Length)部分的单位为 32bit，所以 IP 包头的长度必须为 32bit 的整数倍。因此，在可选项后面，IP 协议会填充若干个 0，以达到 32bit 的整数倍。

tcp 三次握手

附 TCP 字段

源端口和目的端口字段——各占 2 字节。端口是传输层与应用层的服务接口。传输层的复用和分用功能都要通过端口才能实现。

序号字段——占 4 字节。TCP 连接中传送的数据流中的每一个字节都编上一个序号。序号字段的值则指的是本报文段所发送的数据的第一个字节的序号。

确认号字段——占 4 字节，是期望收到对方的下一个报文段的数据的第一个字节的序号。

数据偏移——占 4 bit，它指出 TCP 报文段的数据起始处距离 TCP 报文段的起始处有多远。“数据偏移”的单位不是字节而是 32 bit 字（4 字节为计算单位)。

保留字段——占 6 bit，保留为今后使用，但目前应置为 0。

紧急比特 URG —— 当 URG = 1 时，表明紧急指针字段有效。它告诉系统此报文段中有紧急数据，应尽快传送(相当于高优先级的数据)。

确认比特 ACK —— 只有当 ACK = 1 时确认号字段才有效。当 ACK = 0 时，确认号无效。

推送比特 PSH (Push) —— 接收 TCP 收到推送比特置 1 的报文段，就尽快地交付给接收应用进程，而不再等到整个缓存都填满了后再向上交付。

复位比特 RST (ReSet) —— 当 RST = 1 时，表明 TCP 连接中出现严重差错 (如由于主机崩溃或其他原因)，必须释放连接，然后再重新建立传输连接。

同步比特 SYN —— 同步比特 SYN 置为 1，就表示这是一个连接请求或连接接受报文。

终止比特 FIN (Final) —— 用来释放一个连接。当 FIN = 1 时，表明此报文段的发送端的数据已发送完毕，并要求释放运输连接。

窗口字段 —— 占 2 字节。窗口字段用来控制对方发送的数据量，单位为字节。TCP

连接的一端根据设置的缓存空间大小确定自己的接收窗口大小，然后通知对方以确定对方的发送窗口的上限。

检验和 —— 占 2 字节。检验和字段检验的范围包括首部和数据这两部分。在计算检验和时，要在 TCP 报文段的前面加上 12 字节的伪首部。

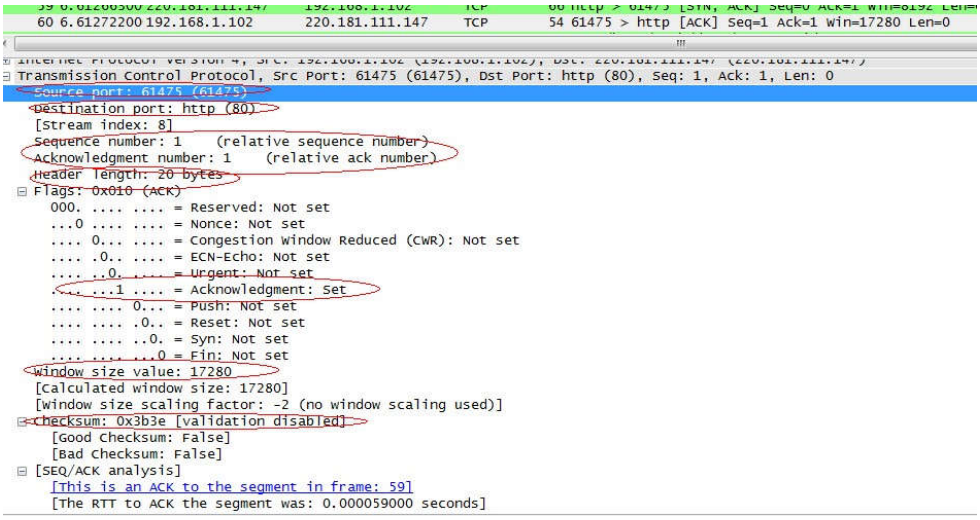
紧急指针字段 —— 占 16 bit。紧急指针指出在本报文段中的紧急数据的最后一个字节的序号。

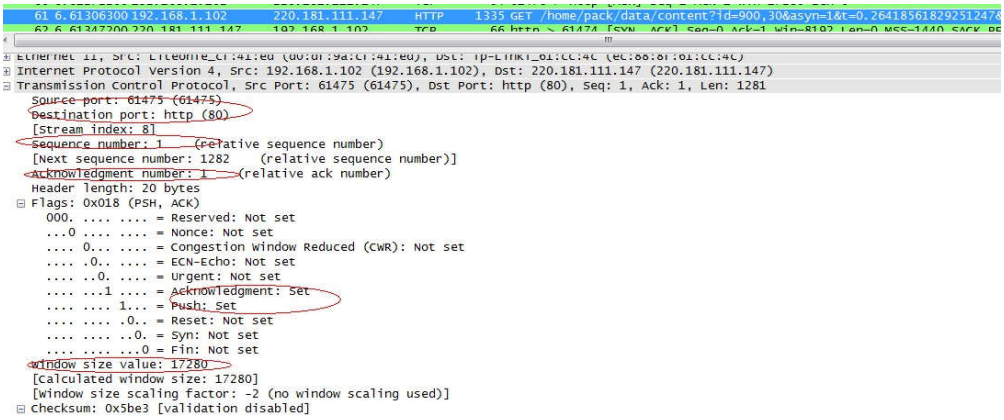
选项字段 —— 长度可变。TCP 最常用的选项字段是 MSS(Maximum Segment Size)，即最大报文段长度 MSS 告诉对方 TCP：“我的缓存所能接收的报文段的数据字段的最大长度是 MSS 个字节。填充字段 —— 这是为了使整个首部长度是 4 字节的整数倍。


58 6.60833000 192.168.1.102 220.181.111.147 TCP 66 61477 > http [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
59 6.61266300 220.181.111.147 192.168.1.102 TCP 66 http > 61475 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1440 SACK_PERM=1
60 6.61272200 192.168.1.102 220.181.111.147 TCP 54 61475 > http [ACK] Seq=1 Ack=1 Win=17280 Len=0
Frame 58: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
Ethernet II, Src: LiteonTe_cf:41:ed (d0:df:9a:cf:41:ed), Dst: Tp-LinkT_61:cc:4c (ec:88:8f:61:cc:4c)
Internet Protocol Version 4, Src: 192.168.1.102 (192.168.1.102), Dst: 220.181.111.147 (220.181.111.147)
Transmission Control Protocol, Src Port: 61477 (61477), Dst Port: http (80), Seq: 0, Len: 0
Source port: 61477 (61477)
Destination port: http (80)
[Stream index: 11]
Sequence number: 0 (relative sequence number)
Header length: 32 bytes
Flags: 0x002 (SYN)
000. .... = Reserved: Not set
...0 .... = Nonce: Not set
.... 0... .... = Congestion Window Reduced (CWR): Not set
.... .0.. .... = ECN-Echo: Not set
.... ..0. .... = Urgent: Not set
....0 .... = Acknowledgment: Not set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
....1. = Syn: Set
[Expert Info (Chat/Sequence): Connection establish request (SYN): server port http]
[Message: Connection establish request (SYN): server port http]
[Severity level: Chat]
[Group: Sequence]
....0 = Fin: Not set
Window size value: 8192
[Calculated window size: 8192]
Window size value: 8192
[Calculated window size: 8192]
Checksum: 0x900c [validation disabled]
[Good Checksum: False]
[Bad Checksum: False]
Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted
Maximum segment size: 1460 bytes
Kind: MSS size (2)
Length: 4
MSS Value: 1460
No-Operation (NOP)
Type: 1
0... .... = Copy on fragmentation: No
.00. .... = Class: Control (0)
...0 0001 = Number: No-Operation (NOP) (1)
Window scale: 2 (multiply by 4)
Kind: Window Scale (3)
Length: 3
Shift count: 2
[Multiplier: 4]
No-Operation (NOP)
No-Operation (NOP)
TCP SACK Permitted Option: True
Kind: SACK Permission (4)
Length: 2

58 6.60833000 192.168.1.102 220.181.111.147 TCP 66 61477 > http [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
59 6.61266300 220.181.111.147 192.168.1.102 TCP 66 http > 61475 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1440 SACK_PERM=1
60 6.61272200 192.168.1.102 220.181.111.147 TCP 54 61475 > http [ACK] Seq=1 Ack=1 Win=17280 Len=0
Transmission Control Protocol, Src Port: http (80), Dst Port: 61475 (61475), Seq: 0, Ack: 1, Len: 0
Source port: http (80)
Destination port: 61475 (61475)
[Stream index: 8]
Sequence number: 0 (relative sequence number)
Acknowledgment number: 1 (relative ack number)
Header length: 32 bytes
Flags: 0x012 (SYN, ACK)
000. .... = Reserved: Not set
...0 .... = Nonce: Not set
.... 0... .... = Congestion Window Reduced (CWR): Not set
.... .0.. .... = ECN-Echo: Not set
.... ..0. .... = Urgent: Not set
....1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
....1. = Syn: Set
[Expert Info (Chat/Sequence): Connection establish acknowledge (SYN+ACK): server port http]
[Message: Connection establish acknowledge (SYN+ACK): server port http]
[Severity level: Chat]
[Group: Sequence]
....0 = Fin: Not set
Window size value: 8192
[Calculated window size: 8192]
Checksum: 0x2008 [validation disabled]

Checksum: 0x2008 [validation disabled]
[Good Checksum: False]
[Bad Checksum: False]
Options: (12 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), No-Operation (NOP), No-Operation (NOP), No-Operation (NOP)
Maximum segment size: 1440 bytes
Kind: MSS size (2)
Length: 4
MSS Value: 1440
No-Operation (NOP)
No-Operation (NOP)
No-Operation (NOP)
No-Operation (NOP)
[Expert Info (Warn/Protocol): 4 NOP in a row - a router may have removed some options]
[Message: 4 NOP in a row - a router may have removed some options]
[Severity level: Warn]
[Group: Protocol]
No-Operation (NOP)
No-Operation (NOP)
TCP SACK Permitted Option: True
Kind: SACK Permission (4)
Length: 2
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 53]
[The RTT to ACK the segment was: 0.065597000 seconds]





到这三次握手的过程就结束了，下面就开始数据传输。

pptp 包

首先要知道 ppp 链路协议，其帧格式如下

每一帧都以标志字符 0 x 7 e 开始和结束。紧接着是一个地址(A)字节，值始

终是 0 x ff，然后是一个值为 0 x 0 3 的控制(C)字节。

协议的两个字段，表示后面信息部分的数据协议是什么，包括：

0x0021——信息字段是 IP 数据报

0xC021——信息字段是链路控制数据 LCP

0x8021——信息字段是网络控制数据 NCP

0xC023——信息字段是安全性认证 PAP

0xC025——信息字段是 LQR

0xC223——信息字段是安全性认证 CHAP

PPP 协议中提供了一整套方案来解决链路建立、维护、拆除、上层协议协商、认证等问题。具体包含这样几个部分：链路控制协议 LCP（Link Control Protocol)；网络控制协议 NCP(Network Control Protocol)；认证协议，最常用的包括口令验证协议 PAP（Password Authentication Protocol）和挑战握手验证协议 CHAP（Challenge-Handshake Authentication Protocol)。

典型的 PPP 链路协商过程分为三个阶段：链路建立阶段、认证阶段(可选)、网络控制协商阶段。PPP 的协商过程主要经过如下图所示五个状态：

链路死亡状态（Dead）——链路一定开始和结束于这个状态。当一个外部事件（例如载波侦听或网络管理员设定）指出物理层已经准备就绪时，PPP 将进入链路建立阶段。

链路建立状态(Establish)——在这个状态 PPP 通过发送和接收链路配置报文(Configuration)，协商具体的参数选项，当收到并发送 Configurations ACK 后，该状态结束，即打开链路。如果线路中断或者配置失效将返回链路死亡状态

认证状态(Authenticate)——在这个状态协商具体的认证参数，是否认证？进行什么认证？认证的参数交换等，当认证通过或不需认证将开始网络层协议的协商，进入网络层协议配置状态，否则链路终止，最后回到链路死亡阶段

网络层协议配置状态(Network)——LCP 协商成功将进入 NCP 的协商阶段，在这个阶段将进行网络层协议的协商，每一种网络层协议（如 IP、IPX 或 AppleTalk）需要单独建立和配置一个 NCP，如果任一的 NCP 协商不成功将随时关闭该 NCP。NCP 协商通后将可以进行网络报文的通信。如果不成功将关闭链路并进入链路终止状态，最后返回初始的链路死亡状态

链路终止状态(Terminate)——因为链路失效、认证失败、链路质量状态失败、链路空闲时间超时以及管理员关闭链路等原因，可随时进入链路终止状态。PPP 协议通过发送 Terminate-Request 并接收到 Terminate-ACK 以后进入该状态

LCP(链路控制协议)

配置、建立、维护和测试数据链路的连接，有如下协商参数：

1 Maximum-Receive-Unit（最大接收单元）

3 Authentication-Protocol（认证协议）

4 Quality-Protocol（质量协议）

5 Magic-Number（魔术数）

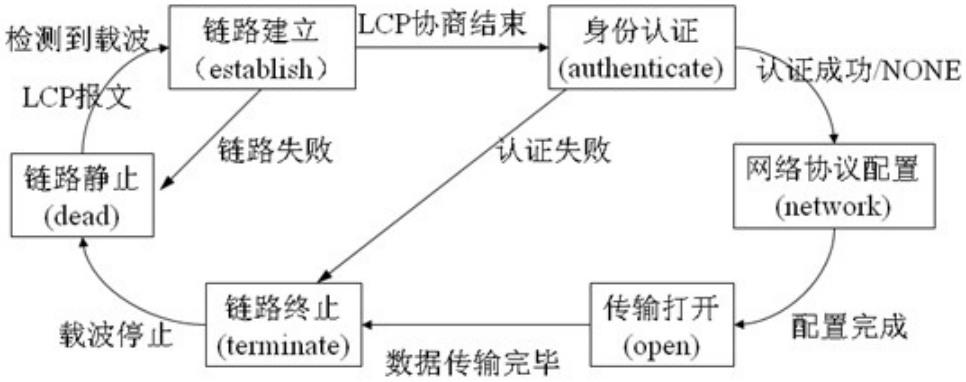
7 Protocol-Field-Compression（协议域压缩）

8 Address-and-Control-Field-Compression（地址和控制域压缩）

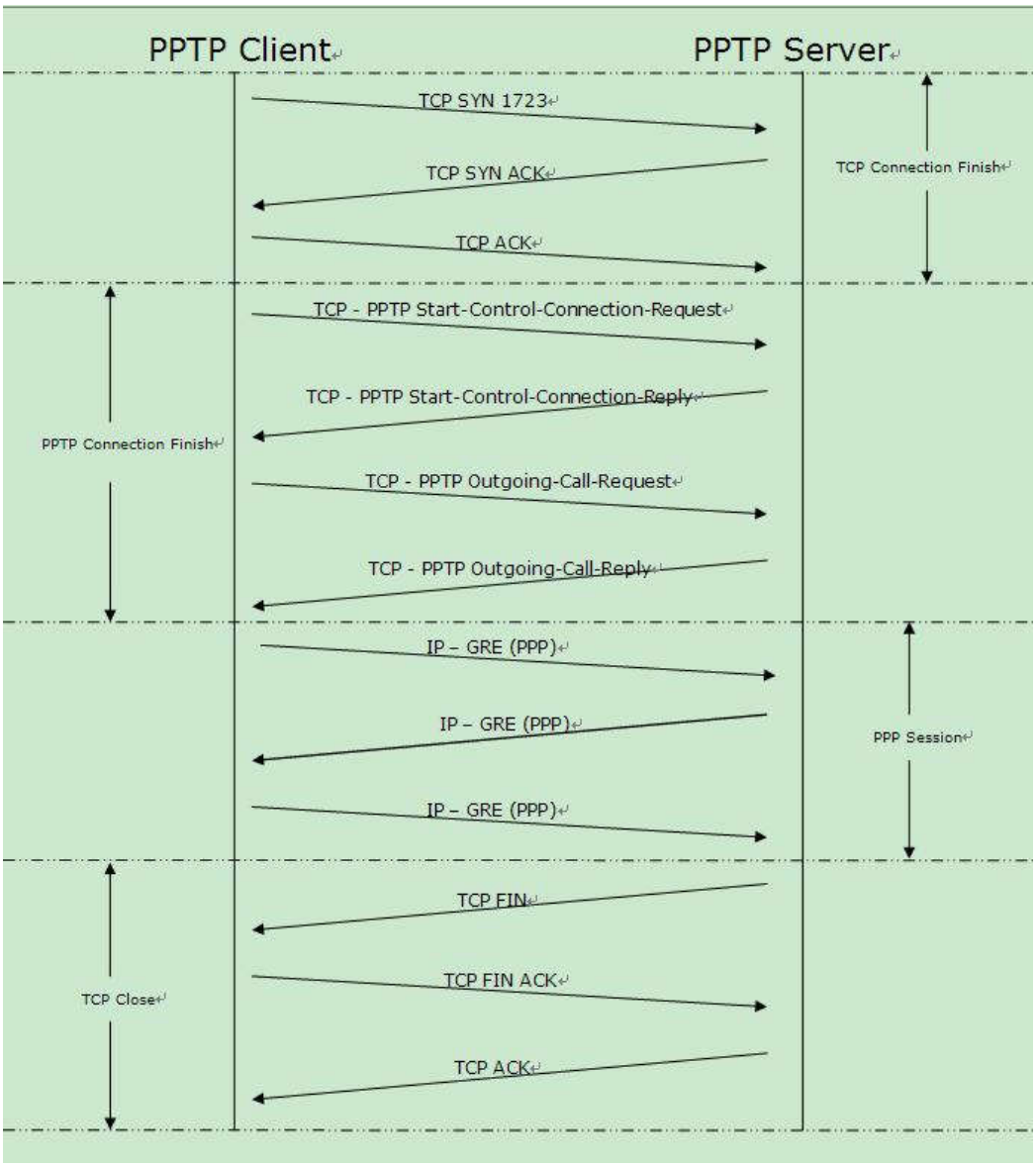
身份认证、压缩、多链路绑定、回拨等功能均在 LCP 子层实现。

NCP(网络控制协议)

建立和配置不同的网络层协议，比如选择 IPCP，协商 IP 及 IP 数据压缩等。IP 地址协商就是在这层实现。



接下来可以看 pptp 连接过程：



PPTP 控制连接通过以下步骤建立：

TCP 连接由 PPTP 客户机上的一个动态分配的 TCP 端口到 PPTP 服务器上的 TCP 端口 1723 建立一个连接（三次握手)。

PPTP 客户端发送一条 PPTP Start-Control-Connection-Request (开始控制连接请求）消息，后者将用于建立一个 PPTP 控制连接。

PPTP 服务器使用一条 PPTP Start-Control-Connection-Reply (开始控制连接应答）消息予以响应。

PPTP 客户端发送一条 PPTP Outgoing-Call-Request（传出调用请求）消息，并选择一个调用 ID，识别用于将数据从 PPTP 客户端发送到 PPTP 服务器的 PPTP 隧道。PPTP 客户端使用 PPTP

Outgoing-Call-Request 消息从 PPTP 服务器请求一个 PPTP 隧道（也称为调用)。

PPTP 服务器发送一条 PPTP Outgoing-Call-Reply (传出调用应答) 消息，并选择自身的调用 ID，识别将数据从 PPTP 服务器发送到 PPTP 客户端的 PPTP 隧道。

PPTP 客户端发送一条 PPTP Set-Link-Info（设置链路信息）消息来指定 PPTP 协商选项。

PPTP 控制连接创建过程的最终结果如下：

PPTP 服务器已允许创建一个 PPTP 隧道。

PPTP 客户端已确定了在通过 PPTP 隧道向 PPTP 服务器发送数据时在 GRE 报头中使用的调用 ID。

PPTP 服务器已确定了在通过 PPTP 隧道向 PPTP 客户端发送数据时在 GRE

报头中使用的调用 ID。

在建立 PPTP 控制连接之后，数据就可以在 PPTP 客户端和 PPTP 服务器之间发送了。 通过 PPTP 连接发送的第一个数据包将用于建立 PPP 连接。

数据包首先被加密并使用一个 PPP 报头进行封装。所得到 PPP 帧将使用一个通用路由封装（GRE）报头进行封装，该报头已针对 PPTP 修改过。 然后，GRE 封装的 PPP 帧使用一个 IP 报头进行封装，这个报头包含对应于 PPTP 隧道端点的源和目标 IP 地址。

为了维持 PPTP 控制连接，PPTP 客户端每隔 60 秒发送一条 PPTP Echo Request（回送请求）消息，不管 PPTP 客户端和服务器之间是否正在发送 GRE 封装的数据。在接收到 PPTP Echo Request 消息时，PPTP 服务器将发送一条 PPTP Echo Reply（回送应答）消息。PPTP Echo Request 消息包含一个 Identifier 字段，该字段的值将在 PPTP Echo Reply 消息中回显，以便 PPTP 客户端能够将 PPTP Echo Request 与其应答相匹配。

为了终止 PPTP 连接，PPP 连接、PPTP 协议连接和 TCP 连接必须全部终止。 当 PPTP 客户端终止 PPTP 连接时，将会交换如下数据包：

PPTP 客户端发送一条 PPTP Set-Link-Info 消息来指定链路的 PPP 参数。

PPTP 客户端发送一条 Link Control Protocol (LCP) Terminate-Request 消息来终止 PPP 连接。

PPTP 服务器发送一条 PPTP Set-Link-Info 消息来指定链路的 PPP 参数。

PPTP 服务器发送 LCP Terminate-Ack 消息来响应 LCP Terminate-Request 消息，从而终止 PPP 连接。

PPTP 客户端发送一条 PPTP Clear-Call-Request 消息，向 PPTP 服务器表

示 PPTP 控制连接即将终止。

PPTP 服务器使用一条 PPTP Call-Disconnected-Notify 消息进行响应。

PPTP 客户端发送一条 PPTP Stop-Control-Connection-Request 消息来终止 PPTP 控制连接.

PPTP 服务器使用一条 PPTP Stop-Control-Connection-Reply 消息进行响应。

TCP 连接终止。

如果 PPTP 服务器要终止连接，所交换的消息是相同的，只要将上述过程中的 PPTP 客户端替换成了 PPTP 服务器即可

附 GRE 报文格式

gre 报文格式(GRE 头部的长度为 4～20 字节)

1,前 4 字节是必须出现的

2,第 5～20 字节将根据第 1 字节的相关 bit 位信息，可选出现

3,GRE 头部的长度将影响 Tunnel 口的 mtu 值

0bit C:校验和标志位。如配置了 checksun 则该位置为 1，同时校验和（可选)、偏离（可选）部分的共 4 bytes 出现在 GRE 头部。

如不配置 checksun 则该位置为 0，同时校验和（可选)、偏离（可选）部分不出现在 GRE 头部。

1bit R:路由标志位。 如 R 为 1，校验和（可选）、偏离（可选）、路由（可选）部分的共 8 bytes 出现在 GRE 头部。

如 R 为 0, 校验和（可选）、偏离（可选）、路由（可选）部分不出现在 GRE 头部。

2bit K:密钥标志位。 如配置了 KEY 则该位置为 1，同时密钥（可选）部分的共 4 bytes 出现在 GRE 头部。

如不配置 KEY 则该位置为 0，同时密钥（可选）部分不出现在 GRE 头部。

3bit S:序列好同步标志位。如配置了 sequence-datagrams 则该位置为 1，同时序列号（可选）部分的共 4 bytes 出现在 GRE 头部。

如不配置 sequence-datagrams 则该位置为 0，同时序列号（可选）部分不出现在 GRE 头部。

4bit s:严格源路由标志位。 除非所有的路由都符合严格源路由，该 bit 位为 1。通常该 bit 为 0。

5~7bit：递归控制：该位置需为 0

8~12bit: 未定义，需为 0

13~15 版本：需为 0

16~31 协议类型：常用的协议，例如 IP 协议为 0800

针对 PPTP 修改过的 GRE 报头具有如下所示的结构

0bit C:校验和标志位，对于 PPTP，该标志总被设为 0。

1bit R:路由标志位，对于 PPTP，该标志总被设为 0。

2bit K:密钥标志位对于 PPTP，该标志总被设为 1。Key 字段是 Protocol Type、Payload Length 和 Call ID 字段的组合。

3bit S:序列好同步标志位,当设置为 1 时,表示提供了 Sequence Number 字段。

4bit s:严格源路由标志位对于 PPTP，该标志总被设置为 0。

Recursion Control（递归控制） 一个用于递归的 3 位标志，对于 PPTP，该字段总被设为 0。

Flags 一个用于 GRE 标志的 4 位字段。对于 PPTP，该字段总被设为 0。

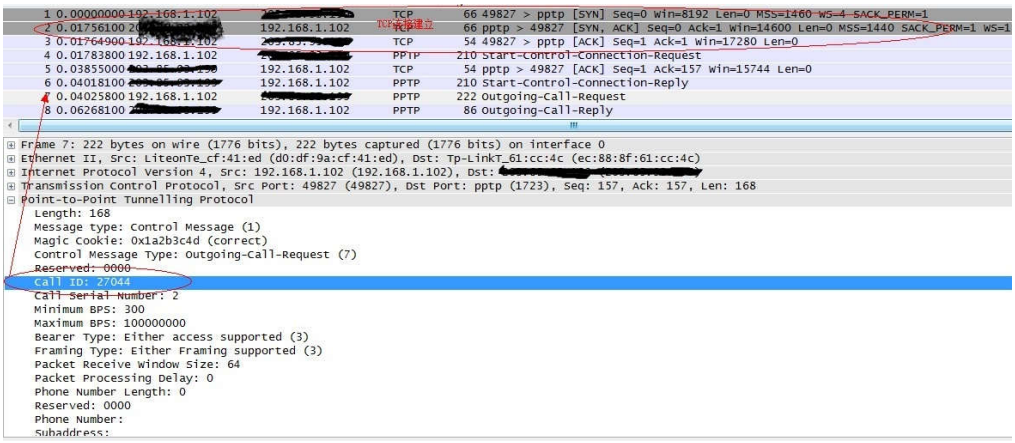
Version 一个用于表示 GRE 报头版本的 3 位字段。对于 PPTP，该字段总被设为 1。

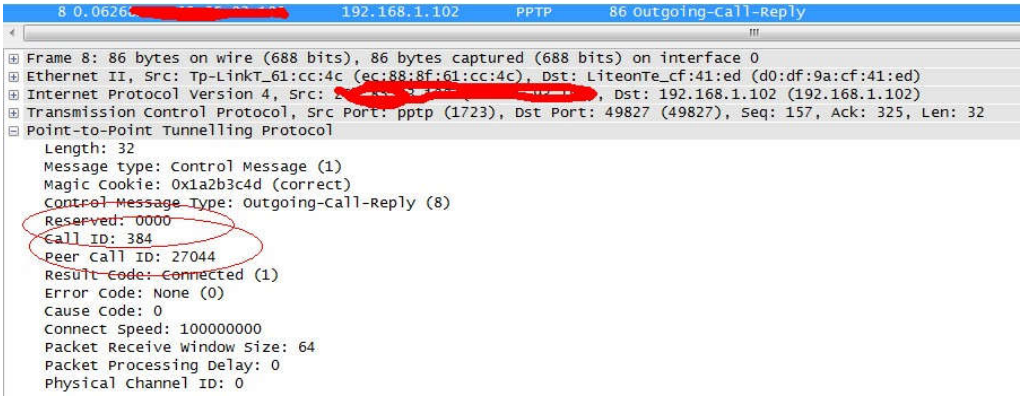
Protocol Type 一个用于存储 GRE 有效负载（payload）的 EtherType 值的 16 位字段。对于 PPTP，该字段总被设为 0x880B，即 PPP 帧的 EtherType 值。

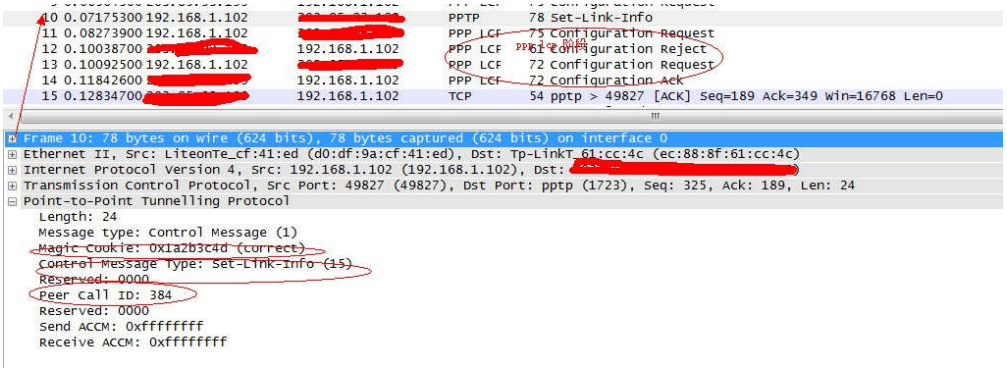
Call ID 一个用于表示这个包的 PPTP 隧道的 16 位字段。对于 PPTP 连接，Call ID 字段有两个不同的值。 一个值用于 PPTP 客户端发送的数据，另一个值用于 PPTP 服务器发送的数据。

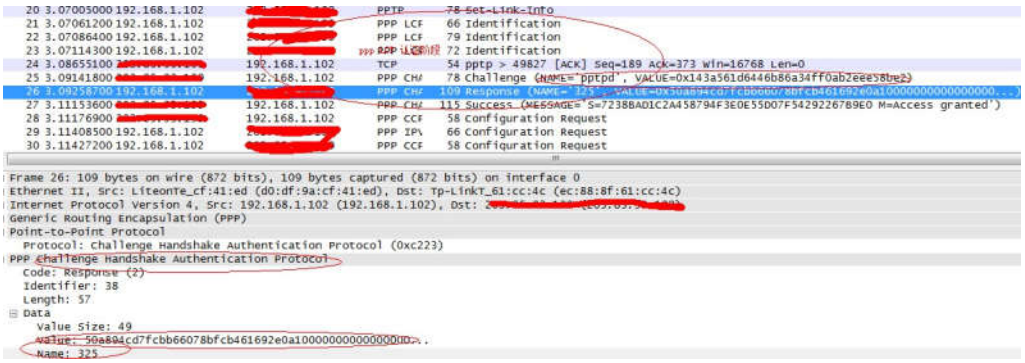
Sequence Number 一个用于表示这个数据包的序列号的 32 位字段。该字段仅在 Sequence Number Present 标志被设置为 1 时才提供。

下面是我通过 wiresharke 抓的包来验证



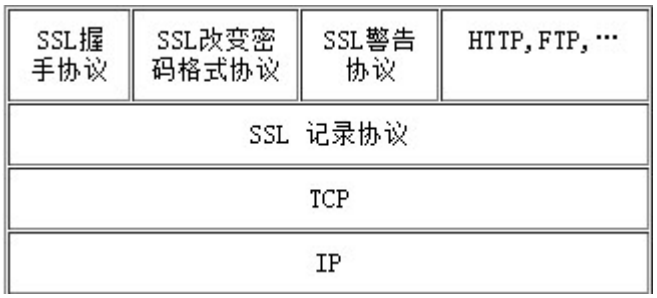






下面我想总结下 openvpn 的实现，但是先要了解 ssl。

ssl 在 TCP/IP 模型位置如下图



SSL：（Secure Socket Layer，安全套接字层） ：位于可靠的面向连接的网络层协议和应用层协议之间的一种协议层。SSL 通过互相认证、使用数字签名确保完整性、使用加密确保私密性，以实现客户端和服务器之间的安全通讯。 该协议由两层组成：SSL 记录协议和 S S L 握手协议。

ssl 记录协议:

SSL 记录协议为 SSL 连接提供两种服务：机密性和报文完整性。在 SSL 协议中，所有的传输数据都被封装在记录中,记录是由记录头和长度不为 0 的记录数据组成的。所有的 SSL 通信都使用 SSL 记录层，记录协议封装上层的握手协议、警告协议、改变密码格式协议和应用数据协议。其包涵如下字段：

内容类型(8 位):封装的高层协议.已经定义的内容类型是握手协议、警告协议、改变密码格式协议和应用数据协议.

主要版本（8 位)：使用的 SSL 主要版本.对于 SSLv3.0，值为 3.

次要版本（8 位)：使用的 SSL 次要版本.对于 SSLv3.0，值为 0.

压缩长度（16 位)：明文数据（如果选用压缩则是压缩数据）以字节为单位的

长度.

SSL 握手协议

SSL 握手协议被封装在记录协议中，该协议允许服务器与客户机在应用程序传输和接收数据之前互相认证、协商加密算法和密钥。在初次建立 SSL 连接时服务器与客户机交换一系列消息。

SSL 握手协议报文头包括三个字段：

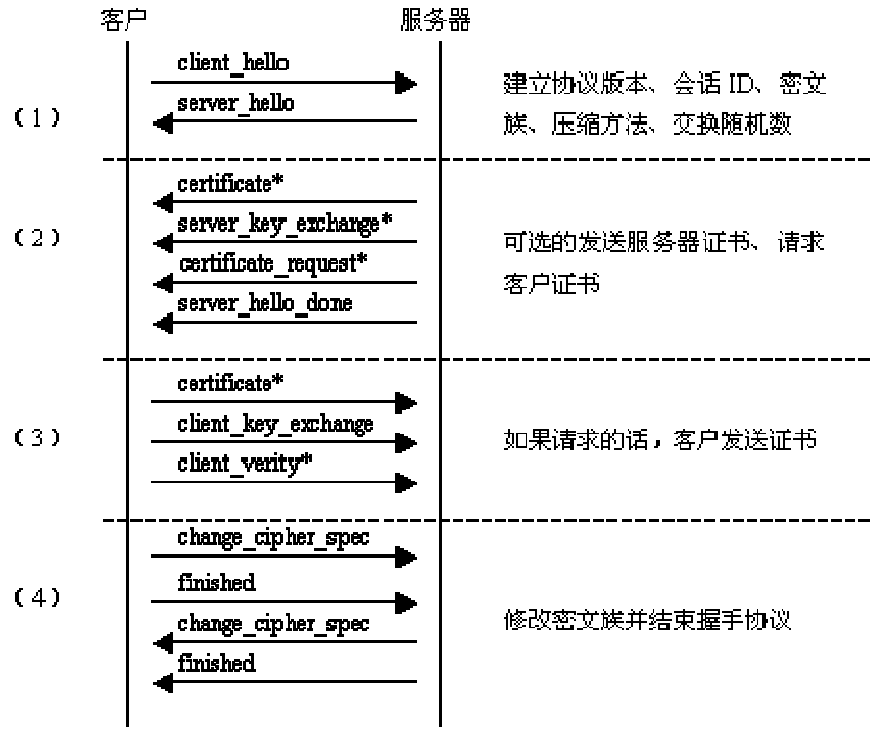
类型（1 字节）：该字段指明使用的 SSL 握手协议报文类型。SSL 握手协议报文包括 10 种类型。报文类型见下图。

长度（3 字节）：以字节为单位的报文长度。

内容（≥1 字节）：使用的报文的有关参数。

报文类型	参数
hello_request	空
client_hello	版本、随机数、会话ID、密文族、压缩方法
server_hello	版本、随机数、会话ID、密文族、压缩方法
certificate	X.509v3证书链
server_key_exchange	参数、签名
certificate_request	类型、授权
server_done	空
certificate_verify	签名
client_key_exchange	参数、签名
finished	Hash值

ssl 握手协议协商过程:



（1）建立安全能力。客户机向服务器发送 client_hello 报文，服务器向客户机回应 server_hello 报文，建立如下的安全属性：协议版本，会话 ID，密文族，压缩方法，同时生成并交换用于防止重放攻击的随机数。密文族参数包括密钥交换方法（Deffie-Hellman 密钥交换算法、基于 RSA 的密钥交换和另一种实现在 Fortezza chip 上的密钥交换)、加密算法（DES、RC4、RC2、3DES 等)、MAC 算法（MD5 或 SHA-1)、加密类型（流或分组）等内容。

(2) 认证服务器和密钥交换。在 hello 报文之后，如果服务器需要被认证，服务器将发送其证书。如果需要，服务器还要发送 server_key_exchange。然后，服务器可以向客户发送 certificate_request 请求证书。服务器总是发送 server_hello_done 报文，指示服务器的 hello 阶段结束。

（3）认证客户和密钥交换。客户一旦收到服务器的 server_hello_done 报文，客户将检查服务器证书的合法性（如果服务器要求)，如果服务器向客户请求了证书，客户必须发送客户证书，然后发送 client_key_exchange 报文，报

文的内容依赖于 client_hello 与 server_hello 定义的密钥交换的类型。最后，客户可能发送 client__verify 报文来校验客户发送的证书，这个报文只能在具有签名作用的客户证书之后发送。

（4）结束。客户发送 change_cipher_spec 报文并将挂起的 CipherSpec 复制到当前的 CipherSpec。这个报文使用的是改变密码格式协议。然后，客户在新的算法、对称密钥和 MAC 秘密之下立即发送 finished 报文。finished 报文验证密钥交换和鉴别过程是成功的。服务器对这两个报文响应，发送自己的 change_cipher_spec 报文、finished 报文。握手结束，客户与服务器可以发送应用层数据了。 当客户从服务器端传送的证书中获得相关信息时，需要检查以下内容来完成对服务器的认证：时间是否在证书的合法期限内；签发证书的机关是否客户端信任的；签发证书的公钥是否符合签发者的数字签名；证书中的服务器域名是否符合服务器自己真正的域名。服务器被验证成功后，客户继续进行握手过程。

同样的，服务器从客户传送的证书中获得相关信息认证客户的身份，需要检查：用户的公钥是否符合用户的数字签名；时间是否在证书的合法期限内；签发证书的机关是否服务器信任的；用户的证书是否被列在服务器的 LDAP 里用户的信息中；得到验证的用户是否仍然又权限访问请求的服务器资源。

SSL 报警协议是用来为对等实体传递 SSL 的相关警告。如果在通信过程中某一方发现任何异常,就需要给对方发送一条警示消息通告。警示消息有两种：

Fatal 错误,如传递数据过程中发现错误的 MAC,双方就需要立即中断会话,同时消除自己缓冲区相应的会话记录.

Warning 消息,这种情况,通信双方通常都只是记录日志,而对通信过程不造成

任何影.

SSL 修改密文协议

为了保障 SSL 传输过程的安全性,客户端和服务器双方应该每隔一段时间改变加密规范。所以有了 SSL 修改密文协议。SSL 修改密文协议是 3 个高层的特定协议之一,也是其中最简单的一个。在客服端和服务器完成握手协议之后,它需要向对方发送相关消息(该消息只包含一个值为 1 的单字节)，通知对方随后的数据将用刚刚协商的密码规范算法和关联的密钥处理，并负责协调本方模块按照协商的算法和密钥工作.

应用数据协议：将应用数据直接传递给记录协议 SSL 报警协议是用来为对等实体传递 SSL 的相关警告。如果在通信过程中某一方发现任何异常,就需要给对方发送一条警示消息通告。警示消息有两种：

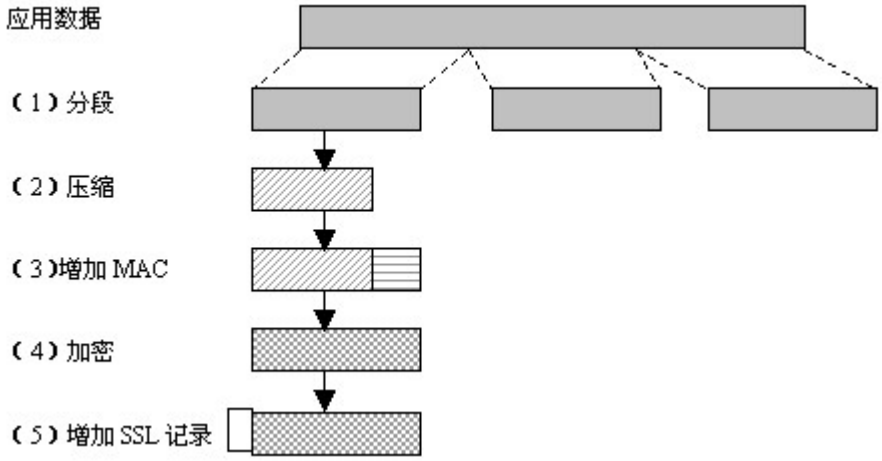
Fatal 错误：如传递数据过程中发现错误的 MAC,双方就需要立即中断会话,同时消除自己缓冲区相应的会话记录.

Warning 消息：这种情况,通信双方通常都只是记录日志,而对通信过程不造成任何影.

SSL 修改密文协议

为了保障 SSL 传输过程的安全性,客户端和服务器双方应该每隔一段时间改变加密规范。所以有了 SSL 修改密文协议。SSL 修改密文协议是 3 个高层的特定协议之一,也是其中最简单的一个。在客服端和服务器完成握手协议之后,它需要向对方发送相关消息(该消息只包含一个值为 1 的单字节)，通知对方随后的数据将用刚刚协商的密码规范算法和关联的密钥处理，并负责协调本方模块按照协商的算法和密钥工作.

应用数据协议：将应用数据直接传递给记录协议应用数据的传输过程为：



（1）应用程序把应用数据提交给本地的 SSL；

（2）发送端根据需要，使用指定的压缩算法，压缩应用数据；

（3）发送端使用散列算法对压缩后的数据进行散列，得到数据的散列值；

（4）发送端把散列值和压缩后的应用数据一起用加密算法加密；

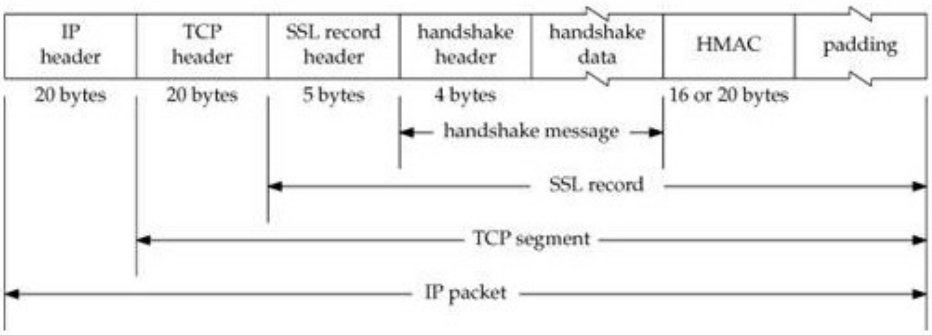
（5）密文通过网络传给对方；

（6）接收方用相同的加密算法对密文解密，得到明文；

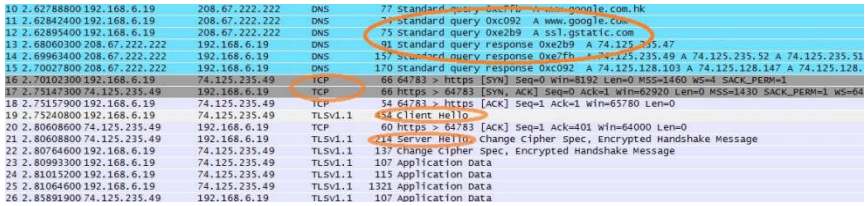
（7）接收方用相同的散列算法对明文中的应用数据散列；

（8）计算得到的散列值与明文中的散列值比较；

完整数据包格式如下：



本地访问 https://www.google.com，通过抓包可以清晰看出上述过程！



貌似该说 openvpn 了，答案不是！下面还要说一个内容就是 ipsec。

IPsec(缩写 Internet Protocol Security)是保护 IP 协议安全通信的标准，它主要对 IP 协议(因此它工作在网络层)分组进行加密和认证。

IPSec 是一个用于保证通过 IP 网络进行安全（保密性、完整性、真实性）的秘密通信的开放式标准框架

IPSec 实现了网络层的加密和认证，在网络体系结构中提供了一种端到端的安全解决方案

IPSec 加密的数据包可以通过任何 IP 网络，而不需要对中间的网络互联设备做任何的改变

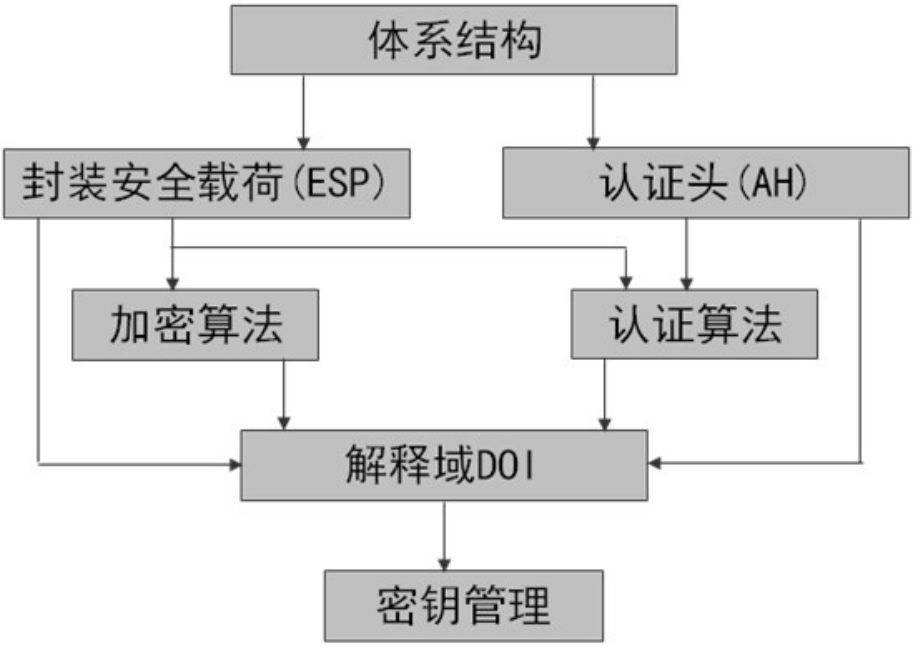
需要知道加密的惟一设备是端点，大大降低了实现和管理的成本

IPsec 作为一个协议族（即一系列相互关联的协议）由以下部分组成：

(1)保护分组流的协议；如加密分组流的封装安全载荷(ESP)或者认证头(AH)（AH 和 ESP 主要用于数据的封装）

(2)用来建立这些安全分组流的密钥交换协议。IKE 协议是唯一已经制定的密钥交换协议。

IPSec 协议主要由 Internet 密钥交换协议（IKE）、认证头（AH）及封装安全载荷(ESP)等 3 个子协议组成，还涉及认证和加密算法以及安全关联 SA 等内容，关系图如下：



IKE Internet 密钥交换是一种协商和交换安全参数和认证密钥的体系框架，用于建立 IPSEC VPN 特性的安全参数协商功能由 IKE 来完成.

1. 协商安全参数： 身份验证方式 封装方式 加密方式 完整性检测方式...

2. 动态产生主密钥和会话密钥.

3. 进行双方的身份验证.(通过身份认证可以保证数据的真实性。常用的身份认证方式包括：Pre-shared key，预共享

IKE 是一种混合型协议，由 RFC2409 定义，包含了 3 个不同协议的有关部分：ISAKMP、Oakley 和 SKEME。

ISAKMP/Oakley/SKEME 是为 IKE 的协商提供服务的，它提供了实现 IKE 的框架、密钥交换模式和方法、密钥的更新方法。

ISAKMP 是“Internet 安全关联和密钥管理协议”的简称，ISAKMP 对验证和密钥交换提出了结构框架，但没有具体定义，在该筐架以内，它定义了每一次交换的包结构，每次需要几个包交换，主模式 6 个包交换和主动（积极）模式

3 个包交换，它由美国国家安全处开发，在配置 IPSEC VPN 的时候，只能设置它，后两个协议不能被设置。ISAKMP 被设计用来独立的进行密钥交换，即被设计用于支持多种不同的密钥交换。

Oakley 描述了一系列被称为“模式”的密钥交换，并详述了每一种提供的服务。

SKEME 描述了一种提供匿名，否认，和快速密钥更新的通用密钥交换技术。

IKE 是使用部分 Oakley，部分 SKEME，并结合 ISAKMP 的一种协议，它使用 ISAKMP 来得到已验证的用于生成密钥和其它安全联盟（如 AH，ESP）中用于 IETE IPsec DOI 的材料。

IKE 协议是 Oakley 和 SKEME 协议的一种混合，并在由 ISAKMP 规定的一个框架内运作。Oakley 和 SKEME 定义了通信双方建立一个共享的验证密钥所必须采取的步骤。IKE 利用 ISAKMP 语言对

这些步骤以及其它信息交换措施进行表述。

IKE 是一个使用已知的 UDP 端口（端口号 500）的应用层协议

IKE 主要完成两个作用：安全关联的集中化管理，减少连接时间、密钥的生成和管理。

AH 协议　验证报头　可以提供身份验证和完整性的功能，但本身不能实现数据的加密，而且不能穿越 NAT. 不常用.

AH的格式

认证头（AH）协议

AH 的协议代号为 51

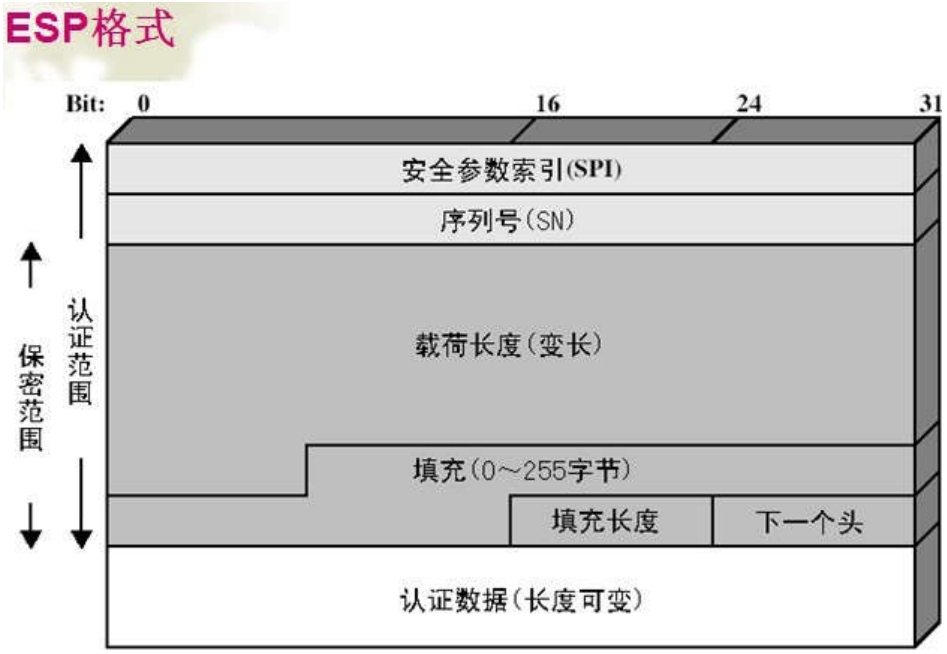
基于网络层的一个安全协议

IPSec 协议的重要组成部分

用于为 IP 数据包提供安全认证的一种安全协议

AH 的功能：为 IP 数据包提供强认证的一种安全机制，具有为 IP 数据包提供数据完整性（通过消息认证码产生的校验值保证）、数据源认证（通过在数据包中包含一个将要被认证的共享秘密或密钥来保证）、抗重放攻击（通过使用一个经认证的序列号来实现）等功能 .

ESP 协议 封装安全协议 可以实现身份验证 / 加密 / 完整性等一系列安全特性. 并可以穿越 NAT.



封装安全载荷（ESP）协议

AH 只能确保 IP 数据包的来源和完整性，不能为 IP 数据包提供机密性，即 IP 数据包在传输过程是可见的

引入能提供机密性服务的 ESP

协议代号为 50

ESP 的功能：主要支持 IP 数据包的机密性，将需要保护的数据进行加密后再封装到新的 IP 数据包中。此外，ESP 还提供认证服务，ESP 只认证 ESP 头之后的信息，比 AH 认证的范围小

IPSec 连接需要两个步骤：

1.建立管理链接 （IKE SA）

2.建立数据连接 （ipsec SA）

在没有详细说这两个过程的时候，先解释一下 SA

安全关联 （Security Association）一组用于保护信息安全的策略约定, 即如何利用相关的安全参数来保护数据流, (SA 可看作是一个通过公共网络到某一

特定个人、某一组人或某个网络资源的安全通道，允许构建不同类型的安全通道）.

安全关联——为了使通信双方的认证算法、加密算法保持一致，相互间建立的联系

安全关联 SA 是构成 IPSec 的基础，是两个通信实体经协商建立起来的一种协定，决定了用来保护数据报文安全的 IPSec 协议、转码方式、密钥及密钥的有效存在时间

IPSec 方案最终会构建一个 SA 的数据库 SADB，用于维护 IPSec 协议保障数据包安全的 SA 记录

在 IPSec 保护 IP 报文前，必须先建立一个安全联盟，可以手工或动态建立，Internet 密钥交换 IKE 用于动态建立安全联盟，IKE 代表 IPSec 对 SA 进行协商，并对 SADB 进行填充 .

SA 由 3 个参数唯一标识

安全参数索引（SPI）

目的 IP 地址

安全协议标识符：AH 或 ESP 安全关联

SPI (Security Parameter Index)，由 IKE 自动分配,发送数据包时，会把 SPI 插入到 IPSec 头中,接收到数据包后，根据 SPI 值查找 SAD 和 SPD，从而获知解密数据包所需的加解密算法、hash 算法等。

SA 是一个单向的逻辑连接，也就是说，在一次通信中，IPSec 需要建立两个 SA，一个用于入站通信，另一个用于出站通信.

使用 SPI 可以标识路由器与不同对象之间的连接.

建立安全联盟的方式有两种，一种是手工方式（Manual），一种是 IKE 自动协商（ISAKMP）方式。

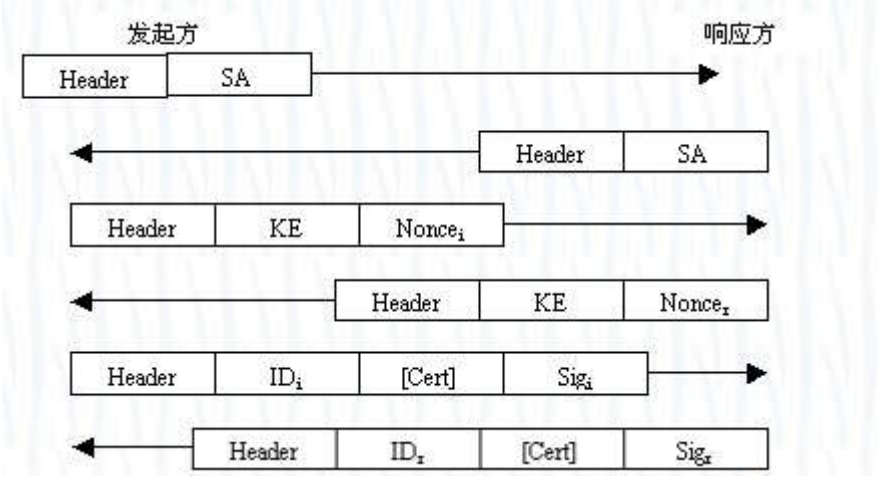
手工方式配置比较复杂，创建安全联盟所需的全部信息都必须手工配置，而且 IPSec 的一些高级特性（例如定时更新密钥）不能被支持，但优点是可以不依赖 IKE 而单独实现 IPSec 功能。 该方式适用于当与之进行通信的对等体设备数量较少的情况，或是在小型静态环境中。

IKE 自动协商方式相对比较简单，只需要配置好 IKE 协商安全策略的信息，由 IKE 自动协商来创建和维护安全联盟。该方式适用于中、大型的动态网络环境中。

该方式建立 SA 的过程分两个阶段。第一阶段，协商创建一个通信信道（ISAKMP SA），并对该信道进行认证，为双方进一步的 IKE 通信提供机密性、数据完整性以及数据源认证服务；第二阶段，使用已建立的 ISAKMP SA 建立 IPsec SA。分两个阶段来完成这些服务有助于提高密钥交换的速度。

第一阶段，使用 ISAKMP/IKE 建立管理连接时分为主模式和积极模式（只有 remote vpn 和 Easy vpn 是积极模式的,其他都是用主模式来协商的）

主模式交换提供了身份保护机制，经过三个步骤，六个消息，头两个消息协商策略；下两个消息交换 Diffie-Hellman 的公共值和必要的辅助数据；最后的两个消息验证 Diffie-Hellman 交换。



消息 1:发送方向对等体发送一条包含一组或多组策略提议,在策略提议中包括 5 元组(加密算法,散列算法,DH,认证方法,IKE SA 寿命)

1．策略协商，在这一步中，就四个强制性参数值进行协商:

1）加密算法：选择 DES 或 3DES

2）hash 算法：选择 MD5 或 SHA

3）认证方法：选择证书认证、预置共享密钥认证或 Kerberos v5 认证

4）Diffie-Hellman 组的选择

消息 2：接受方查看 IKE 策略消息,并尝试在本地寻找与之匹配的策略,找到后,则有一条消息去回应。

注意!!!发起者会将它的所有策略发送给接受者,接受者则在自己的策略中寻找与之匹配的策略(对比顺序从优先级号小的到大的)（默认策略实际就是个模版没作用，如果认证只配置预共享的话，其他参数就会 copy 默认策略里的）

1,2 交换后的结果：由于通信双方决定了一个特定的策略组后，它们以后的通信便必须根据它进行，所以这种形式的协商是两个 IKE 通信实体第一步所需要做的。

消息 3 和消息 4，这 2 条消息,用于交换 DH 的公开信息和随机数。虽然名

为密钥交换，但事实上交换的只是一些 DH 算法生成共享密钥所需要的基本材料信息。

两个对等体根据 DH 的公开信息都算出了双方相等的密值后,两端主机可以各自生成出完全一样的共享"主密钥"，保护紧接其后的认证过程。

（D-H 算法　用于在不安全的环境中, 如 INTERNET, 交换只有双方才知道的保密密钥, 用于保护最初的安全协商.A 和 B , 公开交换一些数据, 然后各自都能根据这些数据通过复杂的数学运算，计算出一个相同的密钥, 但别的用户却不能.）

消息 5 和消息 6

这 2 条消息用于双方彼此验证,这个过程是受上面协商的主密钥加密保护的，DH 交换需要得到进一步认证，如果认证不成功，通信将无法继续下去."主密钥"结合在第一步中确定的协商算法，对通信实体和通信信道进行认证。在这一步中，整个待认证的实体载荷，包括实体类型、端口号和协议，均由前一步生成的"主密钥"提供机密性和完整性保证。

在野蛮模式下，总共三个信息被交换。

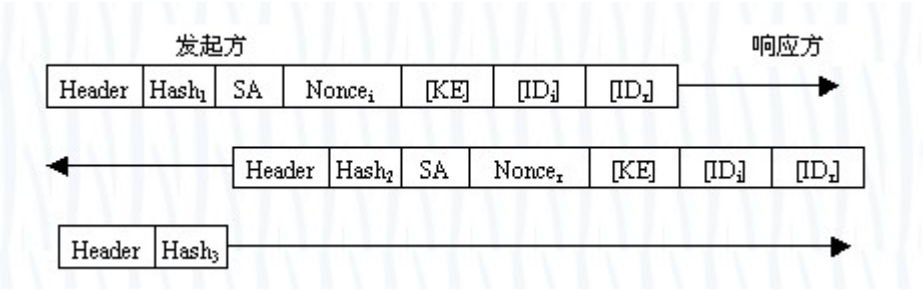
第一个信息由 SA、nonce 和身份组成。

第二个信息是，在验证发起方并接受 SA 后，应答方发送 nonce 和身份信息给发起方。

第三个信息是，发起方验证应答方的身份以及进行被提议的信息的交换。

在 Aggressive 模式下，两个在第一次交换发送的身份信息是没有加密的。Aggressive 模式的优点是信息交换快速，但加密被节省了。

第二阶段协商建立 IPsec SA，为数据交换提供 IPSec 服务。第二阶段协商消息受第一阶段 SA 保护，任何没有第一阶段 SA 保护的消息将被拒收。

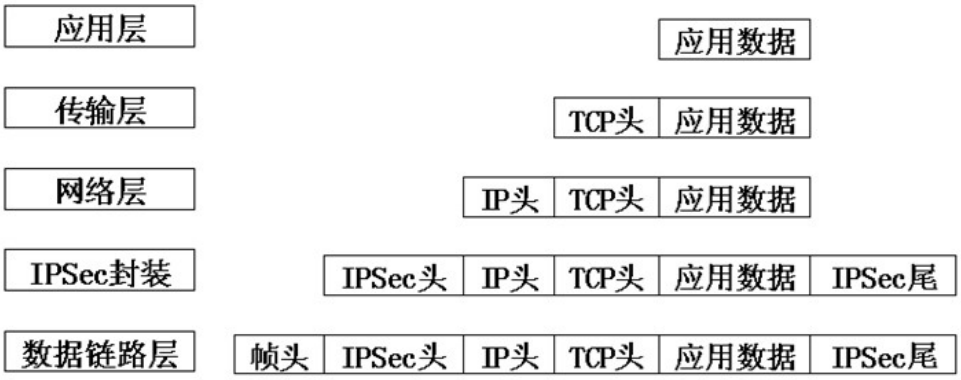


快速模式交换通过三条消息建立 IPsec SA：

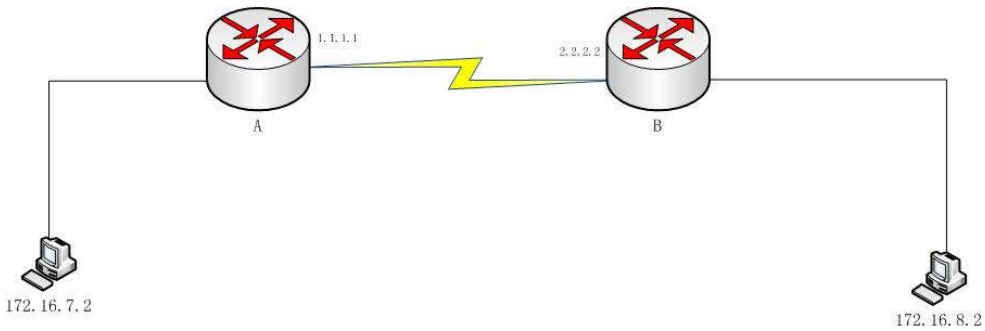
头两条消息协商 IPsec SA 的各项参数值，并生成 IPsec 使用的密钥。包括使用哪种 IPSec 协议（AH 或 ESP）、使用哪种 hash 算法（D5 或 SHA）、是否要求加密，若是，选择加密算法（DES 或 3DES）。在上述三方面达成一致后，将建立起两个 SA，分别用于入站和出站通信。第二条消息还为响应方提供在场的证据；

第三条消息为发起方提供在场的证据。

有 IPSec 机制的数据封装与传递



ipsec 完整的工作过程



如图两个公司通过路由器组成 ipsec vpn，如果 172.16.7.2 想给你 172.16.8.2 通信。

1,从 172.16.7.2 发送出标准的数据包，源 ip172.16.7.2，目的 ip172.16.8.2，通过网关送到路由器 A 上。

2,路由器 A 接受到数据包，拆包分析是到主机 172.16.8.2 的，查路由发现应该走 ipsec 通道，故检查安全策略库（SPD），过程如下图。

3,在主机 172.16.8.2 上，把上述过程反过来执行一遍。

附：

IPSec 包含有两个指定的数据库：

（1）安全策略数据库（SPD）：指定了决定所有输入或者输出的 IP 通信部署的策略，负责所有的 IP 通信流

SPD 条目：目的 IP 地址、源 IP 地址、名称、传输层协议、源端口和目的端口、数据敏感级

（2）安全关联数据库（SAD）：包含有与当前活动的安全关联相关的参数（序列号计数器、序列号计数器溢出、抗重放窗口、AH 信息、ESP 信息、SA 的生存期、IPSec 协议模式、路径最大传输单元 MTU）

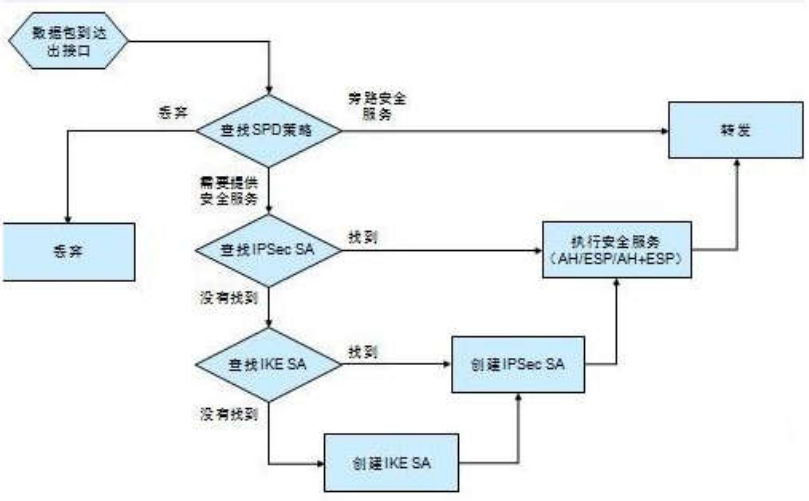
安全策略库（SPD）说明了对 IP 数据报提供何种保护，并以何种方式实施保护。SPD 中策略项的建立和维护应通过协商；而且对于进入和外出处理都应

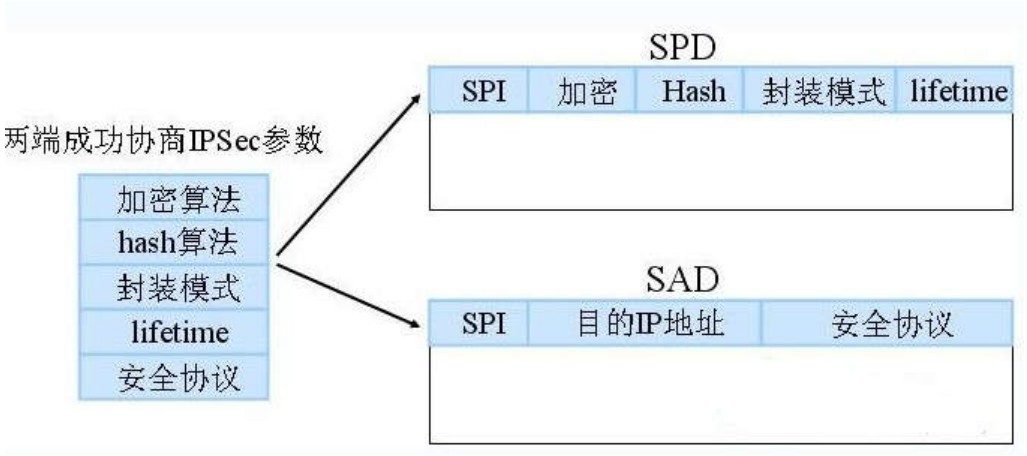
该有自己的策略库。对于进入或外出的每一份数据报，都可能有三种处理：丢弃、绕过或应用 IPSec。SPD 提供了便于用户或系统管理员进行维护的管理接口。可允许主机中的应用程序选择 IPSec 安全处理。SPD 中的策略项记录对 SA(SA 束)进行了规定，其字段包含了 IPsec 协议、模式、算法和嵌套等要求。SPD 还控制密钥管理(如 ISAKMP)的数据包，即对 ISAKMP 数据包的处理明确说明。

安全关联库（SAD）维护了 IPSec 协议用来保障数据保安全的 SA 记录。每个 SA 都在 SAD 中有一条记录相对应。对于外出处理，应 SPD 中查找指向 SAD 中 SA 的指针，如 SA 未建立，则应激活 IKE 建立 SA，并同 SPD 和 SAD 的记录关联起来。对于进入处理，SAD 的记录用目的 IP 地址、IPSec 协议类型和 SPI 标识。

SAD 的查找是通过一个三元组（SAID）：协议、目的地址、SPI 来进行的，三元组标识了唯一的 SA。通过对 SAID 的散列找到 SA 头，然后再进行详细匹配找到相应的 SA。

SAD 和 SPD 之间是通过 SAID 进行关联的。通过查看 SPD 中的 SAID 值，可对 SAD 进行查找，找到该策略项所应该实施的 SA。




SPD
SPI
加密
Hash
封装模式
lifetime
两端成功协商IPSec参数
加密算法
hash算法
封装模式
lifetime
安全协议
SAD
SPI
目的IP地址
安全协议