OpenVPN协议

1. 公钥认证加密技术

网络通信中，信息安全的三个核心问题，一是验证通信对方身份，二是防止信息被泄露，三是防止信息被篡改。

数字签名用于验证身份，加密算法用于加密数据，摘要算法用于验证数据。

加密算法，简单划分来说，主要分两种，对称算法，非对称算法。

对称算法，加解密密钥相同，基本上只能用于加密和解密。

非对称算法，分公钥和私钥，可以用于签名和加解密。

（1）公钥和私钥

密钥有公钥和私钥之分，公钥，顾名思义，是可以公开的密钥。私钥，需要安全保存，不能被泄露。

（2）摘要算法

摘要算法基本原理是将一段数据输入进行运算，产生一段固定长度的数据，然后将数据附加到原始数据上，一起传输出去，接收方收到数据后，对数据进行同样的操作，根据结果可以验证数据是否发生了变化，原则上，原始数据发生了改变，接收方都能够通过计算结果知道数据有变化。

摘要算法也就是一种检错算法（也可能是被篡改的），目前常用的有以下几种：

校验和算法（check sum），循环冗余校验算法（crc），信息摘要算法（MD），哈希算法（HASH），check\_sum和crc经常用于数据通信中，用来检错，比如tcp/udp中使用check\_sum检错，chenk\_sum的检错能力比较弱，crc要强很多。

MD和SHA算法长被用做数据签名算法中，二者的检错能力更强，相对check\_sum和crc。

（3）数字签名

数字签名跟日常生活中的签名起的作用是一样的，用来对被签名的事物进行认证。

数字签名是一串数据，基本原理如下：

被签名数据T，选择摘要算法S1，计算摘要值t，然后选择签名算法S2，对t进行签名运算得到数字签名。

以rsa-sha256算法为例讲解，T计算哈希值t，然后对t使用rsa私钥加密，得到数字签名，因为rsa算法的的公钥和私钥都可以用作加密，所以签名时，直接使用私钥加密数据即可。其他的一些加密算法比如ecc，签名算法相对复杂，不是简单的加密运算。

（4）数字证书

数字证书是一项伟大的发明，其作用跟我们平时使用的证件是类似的，用来证明持有人的身份。数字证书主要有三部分组成，证书基本信息，公钥，数字签名。

证书基本信息，主要包含证书持有人的一些信息，比如组织机构名等。

公钥，是证书使用的公钥，明文

证书签名，证书的签名数据，用来验证证书的有效性（是否是伪造的，是否被篡改过）

（5）公钥基础设施（public key infrastruction）

CA（certificate authority），发证机构，签发数字证书。对外公开签名用的根证书，根证书一般是自签名证书，即数字证书的数字签名是使用证书中公钥对应的私钥产生的。

CA会使用根证书对应的私钥，给用户签发数字证书，其他的用户可以使用根证书对用户证书进行验证。证书签发和验证过程如下图所示：



首先，CA的根证书公开，所有用户对其都默认为是可以信赖的，而且各个用户可以使用证书中的公钥对根证书进行验证，防止信息被篡改。

然后，CA发证中心会使用根证书中的私钥给用户A发证，即根据用户A的发证请求，给用户生成数字签名，用户A要在发证请求中提交证书的基本信息和用户公钥，用户A的私钥单独保存，不能被泄露。

用户A拿到数字证书后就可以使用，其他用户B接收到该证书后，会使用公开的根证书对该证书进行验证，可以有效防止证书的伪造和篡改。

到此还有一个问题，虽然根证书的公钥可以保证用户的数字证书是真实有效的，但是因为数字证书是公开的，如何才能判断持有数字证书的人是真正的证书拥有者，比如黑客通过监听网络数据包，截取到数字证书，如何确定用户身份。

身份认证是通过数字证书对应的私钥去实现的。比如，用户B接收到A的数字证书后，先验证证书的真实有效性（通过根证书公钥），然后使用数字证书中的公钥加密一段数据发送给A，A收到数据后使用私钥解密数据，然后应答B，因为虽然数字证书公开，但是私钥却只有A有，加密数据只能A解密，通过这种方法来验证A的身份。简单来说，身份认证的核心思想是，使用被验证方的公钥加密一段数据发送给对方，来验证对方身份。

1. TLS/SSL协议
   1. TLS记录协议格式

记录协议格式如下：



记录协议主要有四个核心子协议：握手协议（handshake protocol），密钥规格变更协议（change cipher spec protocol），应用协议（application data protocol），警报协议（alert protocol）。



上图中列举了每个子协议主要的协议内容。

* 1. TLS/SSL协议流程

TLS/SSL基本的握手流程如下。



上图是一个相对完整的双向认证的TLS/SSL握手过程。

TLS握手主要做三件事情：加密套件协商；服务端，客户端身份认证；会话密钥交换。

**加密套件协商**

客户端通过ClientHello发送支持的加密套件，服务端通过ServerHello指定使用的加密套件。

主要包含四个方面：密钥协商算法：数字证书算法（非对称）：摘要算法（哈希算法）：加密算法（对称）

例如：

ECDHE-SM2-SM3-SM4

**身份认证**

TLS1.2中，可以单向认证server，可以双向认证server和client，也可以双方都不认证。

在双向认证中，双方采用如下的方法认证。

在server和client互发证书后，client会发送一条ClientKeyExchange来将pre-master key发送给server，pre-master key使用server的公钥进行签名和加密，因为只有server持有对应的私钥，所以原则上只有server可以解密数据，并正确应答client，达到认证server的目的。

为了认证client，client会发送一条certificate verify数据包给server，数据包中包含一条使用client私钥签名（加密）的数据，server收到数据后，使用client公钥验证（解密），来验证client。

实际情况中，身份认证和密钥协商一般是同时进行的。

**会话密钥交换**

如果安全需求不高，可以不使用密钥协商算法，由通信双方一端产生会话密钥（共享密钥），然后使用证书中的公钥加密传输给对端，对端使用私钥解密即可得到会话密钥（OpenVPN的tls key exchange method 1就是使用此方法）。



双方各自产生加密密钥，然后使用对方的公钥加密，然后再使用自己的私钥签名，对方收到后，使用对方公钥验签，然后使用自己私钥解密，既交换了密钥，同时也验证了对方身份。

目前常用的密钥交换算法是DHE和ECDHE算法，即迪非赫尔曼和椭圆曲线迪非赫尔曼算法。以TLS1.2版本，ECDHE算法为例进行讲解：



问：在TLS握手协议中，双方的数字证书需要在网络上传输，这样会造成证书的泄露，会不会有安全隐患？

答：原则上是不会有安全隐患的，因为虽然数字证书泄露了，只是泄露了公钥，与其对应的私钥并没有泄露，窃取了数字证书的不法用户，在进行身份认证时，没有数字证书私钥，无法解密相关数据，认证会失败。

1. OpenVPN协议

（1）协议格式

OpenVPN协议依托于TCP/UDP协议，根据协议头中操作码（OPCODE）的不同来区分包的种类，其与其他网络协议的组包关系如下图所示。



根据OpenVPN头中第一个字节中OPCODE的不同，OpenVPN主要分9种数据包，分别是是：

|  |  |  |
| --- | --- | --- |
| OPCODE | Message type | Role |
| 0x01 | P\_CONTROL\_HARD\_RESET\_CLINET\_V1 | 客户端会话初始化，使用tls key exchange method1 |
| 0x02 | P\_CONTROL\_HARD\_RESET\_SERVER\_V1 | 服务端对客户端会话初始化的回复, 使用tls key exchange method1 |
| 0x03 | P\_CONTROL\_SOFT\_RESET\_V1 | 申请密钥重新协商 |
| 0x04 | P\_CONTROL\_V1 | 控制包 |
| 0x05 | P\_ACK\_V1 | 控制数据包应答 |
| **0x06** | **P\_DATA\_V1** | **数据包** |
|  |  |  |
| 0x07 | P\_CONTROL\_HARD\_RESET\_CLINET\_V2 | 客户端会话初始化，使用tls key exchange method2 |
| 0x08 | P\_CONTROL\_HARD\_RESET\_SERVER\_V2 | 服务端对客户端会话初始化的回复, 使用tls key exchange method2 |
| **0x09** | **P\_DATA\_V2** | **数据包，相对于P\_DATA\_V1，增加了peer id域。** |

OpenVPN的包可以分为两种，control packet（0x01,0x02,0x03,0x04,0x05,0x07,0x08）和data packet（0x06，0x09），其格式如下所示：



**图中，红色字体的是必有的字段，其他字段可能是没有的**。

Control Packet

|  |  |
| --- | --- |
| Local session id | 本地的会话id |
| HAMC | 所有数据数据HMAC，当启动配置--tls-auth时，数据包中才包含此字段 |
| Packet id | 用于应答保护，包含一段序列码和时间戳，启动--tls-auth时，才包含此字段 |
| ack packet id size | 应答的packet id数组的大小，当此字段的值大于0时，才会有下面的ack packet id array和remote session id字段 |
| ack packet id array | Ack packet id size大于0时，才会有此字段，其值是收到的数据包的messa packet id字段 |
| remote session id | 远端的session id，Ack packet id size大于0时，才会有此字段，其值是收到的数据包的local session id字段 |
| message packet id | 本数据包的packe id |

每个control packet必须包含session id和packet id字段，当接收方收到数据后，根据收到的session id和packet id，将收据填充到ack packet id array和remote session id字段，回复对方。

Data Packet

|  |  |
| --- | --- |
| HMAC | 如果没有配置--auth-none，会包含此字段。 |
| IV | 密文，如果没有配置--no-iv，会包含此字段，IV值 |
| peer\_id | 密文，对端id |
| packet id | 密文，如果没有配--no-reply，会包含此字段 |

HMAC是所有密文的MAC值，Data packet 的payload全是密文。



（2）协议流程

一个完整的OpenVPN握手建立的流程如下：



OpenVPN加密套件协商、身份认证，密钥协商使用TLS/SSL协议，如下：



握手完毕后，使用P\_DATA协议传输加密数据，如下：



（3）OpenVPN如何管理密钥的？

通过ip，虚拟地址，还是端口号？

1. OpenVPN

OpenVPN可以理解为通过TLS协议和自定义协议建立安全的TCP/UDP连接，然后通过虚拟网卡技术，在客户端和服务端生成虚拟网卡，修改端设备的路由表，然后将端设备上其他应用的数据导入虚拟网卡，然后交给OpenVPN主进程，然后OpenVPN再打包数据，通过之前建立的TCP/UDP安全连接将数据发送出去。

那么问题来了，修改路由表将其他应用的数据导入虚拟网卡，OpenVPN的主进程是如何通过实际网卡发送数据的？不应该也被导入到虚拟网卡了吗？为什么没有形成死循环呢？