安全协议

1 安全协议概述

TCP/IP 协议在最初设计时是基于一种可信网络环境来考虑设计的，并没有考虑通信时的安全问题。随着 Internet 的广泛应用导致网络中信息资源的重要性逐渐提高，但是 Internet 中的通信基石 TCP/IP 的安全性先天不足，引发加剧了网络安全问题。

为了弥补 TCP/IP 体系结构中各层协议的安全缺陷，安全人员和机构为每一层设计了相应的安全协议，这些安全协议与对应层的通信协议一起，实现该层对等层之间安全的数据通信。

安全协议主要为了解决以下几个信息安全问题：

1. 真实性：对信息的来源进行判断，能对伪造来源的信息予以鉴别。
2. 保密性：保证机密信息不被窃听，或窃听者不能了解信息的真实含义。
3. 完整性：保证数据的一致性，防止数据被非法用户篡改。
4. 可用性：保证合法用户对信息和资源的使用不会被不正当地拒绝。
5. 不可抵赖性：建立有效的责任机制，防止用户否认其行为，这一点在电子商务中是极其重要的。
6. 可控制性：对信息的传播及内容具有控制能力。
7. 可审查性：对出现的网络安全问题提供调查的依据和手段

下面简单介绍一下安全协议目前的主要框架：

安全协议体系结构

HTTPS PGP SET	应用层
SSL/TLS	TCP UDP
IPSec	IP
	以太网 ATM SDH

网络层目前普遍使用 Internet 安全协议（Internet Protocol Security, IPSec）,IPSec 由一组安全协议组成实现 IP 报文端到端安全传输。

传输层使用安全套接层（Secure Socket Layer,SSL）或者传输层安全（Transport Layer Security），传输层的安全协议实现了两端口之间 TCP/UDP 报文的传输安全。

应用层对应不同应用存在多种应用层安全协议，如用于 Web 应用安全的基于 SSL 的 HTTP 协议（Hyper Text Transfer Protocol over Secure Socket Layer,HTTPS），用于电子邮件应用安全的 PGP 协议（Pretty Good Privacy,PGP），用于安全支付安全电子交易（Secure Electronic Transaction,SET）等。

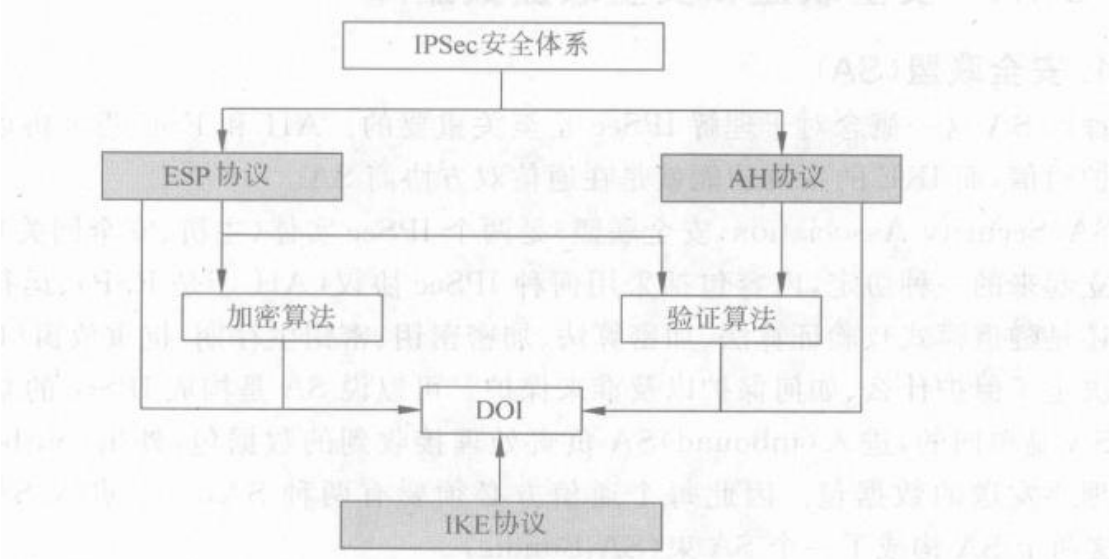
本节主要介绍 IPSec、SSL/TLS、HTTPS 这三种安全协议。

2 IPSec

Internet 协议安全 (IPSec)是一种开放标准的框架结构，是 IETF 以 RFC 形式公布的一组通过使用加密的安全服务以确保在 Internet 协议 (IP) 网络上进行保密而安全的通讯的安全协议标准。IPSec 将几种安全技术结合形成一个比较完整的安全体系结构,它通过在 IP 协议中增加两个基于密码的安全机制——认证头(AH)和封装安全有效负载(ESP),来支持 IP 数据项的认证、完整性和机密性。通过 IP 安全协议和密钥管理协议构建起 IP 层安全体系结构的框架,能保护所有基于 IP 的服务或应用。

IPSec 可以适用于 IPv4 和 IPv6，在 IPv4 中作为一个建议的可选服务，在 IPv6 中是一项必须支持的服务。

下面简单介绍 IPSec 的安全结构——安全协议、安全联盟、安全策略、密钥管理。



IPSec 的安全协议定义了如何通过在 IP 数据包中确保 IP 数据报的机密性、完整性和可认证性。IPSec 使用一种称为安全联盟(Security Associations, SA)的概念性实体集中存放所有需要记录的协商细节。因此,在 SA 中包含了安全通信所需的所有信息,可以将 SA 看做是一个由通信双方共同签署的有关安全通信的“合同”。 IPSec 通过安全策略(Security Policy, SP)为用户提供了一种描述安全需求的方法,允许用户使用安全策略来定义所保护的对象、安全措施以及密码算法等。安全策略由安全策略数据库(SecurityPolicy Database, SPD)来维护和管理。其密钥管理考虑了 IPSec 协议的独立性,将 SA 和密钥的管理分开,采用 IKE 协议定义通信实体间的身份认证、创建安全联盟、协商加密算法以及生成共享会话密钥。

IPSec 具有两种通信模式：传输模式、隧道模式。

传输模式（Transport Mode）只对上层协议数据和选择的 IP 首部字段提供认证保护，仅适用与主机实现。

隧道模式（Tunnel Mode）对整个 IP 数据项提供认证保护，可用于主机也可以用于网关。

一般的，当数据包最终目的地不是安全终点或者使用了 BITS 或 BITW 实施方案的情况下，需要在隧道模式下使用 IPSec。

下面分别介绍 IPSec 的安全协议——AH、ESP，密钥交换协商框架——ISAKMP。

1. AH

AH(AuthenticationHeader，验证头部协议）由 RFC2402 定义，是用于增强 IP 层安全的一个 IPSec 协议，该协议可以提供无连接的数据完整性、数据来源验证和抗重放攻击服务，

不提供机密性服务。下面介绍 AH 的报文格式和主要运行过程。

在传输模式下，在 IP 首部和净荷之间插入鉴别首部 AH；在隧道模式下，整个 IP 分组作为净荷插在外部 IP 首部和 AH 之后。

传输模式

隧道模式

AH 的格式如下：

8 位	8 位	16 位
下一个首部	鉴别首部长度	保留
安全参数索引（SPI）		
序号		
鉴别数据（变长）		

下一个首部（NextHeader）：表示跟在 AH 后面的数据类型，在传榆模式下，该字段是处于保护中的传输层协议的值，比如 6（TCP）17(UDP)或者 5（ESP）。在隧道模式下，AH 所保护的是整个 IP 包，该值是 4，表示 IP-in-IP 协议。

载荷长度（Payloadl-ength)：其值是以 32 位为单位的整个 AH 数据（包拈头部和变长的认证数据）的长度再减 2。

保留(reserved)：作为保留用．实现中应全部设置为 0。

SPI(SecurityParameterIndex，安全参数索引）：SPI 是一个 32 位整数，与源/目的地址 IPSec 协议一起组成的三元组可以为该 IP 包唯一地确定一个 SA。[1，255]保留为将来使用，0 保留本地的特定实现使用。

序列号（SequenceNumber)：序列号是一个 2 位数，作为一个单调递增的计数器，为每个 AH 包赋予一个序号。当通信双方建立 SA 时，计数器初始化为的 SA 是单向的，每发送一个包，外出 SA 的计数器增 1；每接收一个包，进人 SA 的计数器增 1。该字段可以用于抵抗重放攻击。

验证数据（AuthenticationData）：可变长部分，包含了验证数据，也就是 HMAC 算法的结果，称为 ICV 完整性校验值。该字段必须为位的整数倍，如果 ICV 不是 32 位的整数倍，必须进行填充．用于生成 ICV 的算法由 SA 指定。

SA 建立后应用 AH 对报文进行处理，AH 所用到的 HMAC 算法和密钥由 SA 确定。从上面的传输模式和隧道模式可以看出，AH 协议验证的范围包括整个 IP 包，验证过程概括如下：在发送方，整个 IP 包和验证密钥被作为输入，经过 HMAC 算法计算后得到的结果被填充到 AH 头部的“验证数据”字段中；在接收方，整个 IP 包和校验算法所用的密钥也被作为输人，经过 HMAC 算法计算的结果和 AH 头部的“验证数据”字段进行比较，如果一致，说明该 IP 包数据没有被篡改，内容是真实可信的。

在应用 HMAC 算法时，有一些因素需要考虑。在 IP 字段中，有一是可变，而且在传输过程中被修改也是合理的，不能说明该数据包是被非法篡改的。这些字段在计算 HMAC 时被临时用 0 充，具体包括如下。

1.ToS(TypeofService)：8 位的服务类型字段指出了延时、吞吐量和可靠性方面的要求。某些路由器会改该字段以达到特定的 QoS 服务质量。

2.标志字段：这是指用于表示分片的 3 位标志，路由器可能会修改这 3 个标志。

3.分片偏移字段：标志字段后面的偏移字段。

4.TTL:生命期，为了防止 IP 包的无限次路由，每经过一个路由器，该字段减 1 当 TTL 变为 0 时，被路由器抛弃。

5.头部校验和：中间路由器对 IP 包头部作了任何修改之后．必须重新计算头部校验和，因此该字段也是可变的。

6.选项字段。

2. ESP

封装安全载荷（Encapsulate Security Payload，ESP）和 AH 提供的安全能力不同，处理开销也不同。AH 只提供了数据完整性和抗重放性服务，处理开销小；ESP 同时提供了数据完整性认证、抗重放和数据加密传输机制，处理开销大。AH 和 ESP 协议可以分别单独使用，也可以联合使用。

下面介绍 ESP 在两种模式下的报文格式：

IP 首部	ESP 首部	净荷	ESP 尾部	ESP 校验数据

传输模式

外层 IP 首部	ESP 首部	IP 首部	净荷	ESP 尾部	ESP 校验数据

隧道模式

在传输模式下，IP 首部和净荷之间插入 ESP 首部，净荷之后插入 ESP 尾部和 ESP 校验数据（MAC）；在隧道模式下，在 IP 首部和净荷的前面插入外部 IP 首部和 ESP 首部，净荷之后插入 ESP 尾部和 ESP 校验数据。

ESP 首部中包含 SPI 和序号，ESP 尾部包含填充数据、填充字段长度和下一个首部字段。这些字段与 AH 中的字段作用相同。

ESP 采用和 AH 相同的 MAC 算法计算验证数据，但是不同于 AH，ESP 鉴别字段不包括外层 IP 首部中的固定字段。

在传输模式下 ESP 加密运算覆盖的字段是净荷和 ESP 尾部，在隧道模式下加密了包括 IP 首部在内的整个 IP 分组，最常使用的加密算法是 DES 和 AES。

3. ISAKMP

Internet 安全关联密钥管理协议(InternetSecurityAssociationKeyManagementProtocol，ISAKMP）由 RFC2408 定义，定义了协商、建立、修改和删除 SA 的过程和包格式。ISAKMP 只是为 SA 的属性和协商、修改、删除 SA 的方法提供了一个通用的框架，并没有定义具体的 SA 格式和任何密钥交换协议的细节，也没有定义任何具体的加密算法、密钥生成技术或者认证机制。这个通用的框架是与密钥交换独立的，可以被不同的密钥交换协议使用。

ISAKMP 报文可以利用 UDP 或者 TCP，端口都是 500，一般情况下常用 UDP 协议。

ISAKMP 双方交换的内容称为载荷（payload），ISAKMP 目前定义了 13 种载荷，载荷

添加上 ISAKMP 头部，这样就组成了一个 ISAKMP 报文，这些报文按照一定的模式进行交换，从而完成 SA 的协商、修改和删除等功能。

下图为 ISAKMP 报文的首部格式：

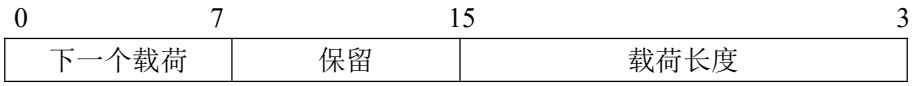
发起方 Cookie：长 64bit，用于表示 SA 的建立、发布和删除；
应答方 Cookie：长 64bit，对发送方做出应答的一方的 Cookie，首次会话该字段为空；
下一个载荷：长 8bit，表示第一个负载的类型；
主版本号：长 4bit，表示 ISAKMP 的主版本；
次版本号：长 4bit，表示 ISAKMP 的次版本；
交换类型：长 8bit，类型如下表所示；
标志：长 8bit，表示对 ISAKMP 交换所做的特定选择设置；
消息的 ID：长 32bit，唯一标识一个消息；
长度：长 32bit，表示以 8bit 为一个单位的消息总长

交换类型	值
None	0
基本交换	1
身份包含交换	2
纯认证交换	3
积极交换	4
信息交换	5
ISAKMP 将来使用	6-31
DOI 专用	32-239
私有用途	240-255

交换类型定义了在通信双方传送的载荷的类型和顺序，这些交换模式对于传输信息的保护程度不同，传输载荷数量也不同。

下面介绍载荷的首部格式和各个载荷类型：

首部格式如下



下一个载荷：长 8bit，表示跟在本载荷后面的下一个载荷的类型。最后一个载荷的下一载荷字段为 0；

保留：长 8bit，全为 0；

载荷长度：长 16bit，以字节为单位表示载荷的长度。

ISAKAP 的负载类型有：

1. SA 负载（SA）：用于 SA 的建立
2. 提议负载（P）：包含 SA 协商信息
3. 变换负载（T）：定义指定协议的安全变换
4. 密钥交换负载（KE）：用于不同的密钥交换技术
5. 识别负载（ID）：身份信息的认证
6. 证书负载（CERT）：用于传输公钥证书
7. 证书请求负载（CR）：用于请求另一方的公钥证书
8. 散列函数负载（HASH）：用于通信双方的认证和验证消息数据完整性
9. 签字负载（SIG）：用于验证消息中数据完整性和不可否认性
10. 一次性随机数负载（NOUNCE）：用于保证交换过程的实时性和抗重放攻击
11. 通告负载（N）：包含于此 SA 相关的错误信息和状态信息
12. 删除负载（D）：表示发送者以及在数据库中删除某个 SA

下面简单介绍 ISAKMP 的协商过程：

ISAKMP 的协商过程分为两个阶段，两个阶段所协商的对象不同，协商过程的交换方式是由 ISAKMP 定义的或者由密交换协议（IKE)定义的。

第一阶段要协商的 SA 可以称为 ISAKMP SA(在 IKE 中可以称为 IKESA〕，该 SA 是为了保证第二阶段的安全通信。

第二阶段要协商的 SA 是交换协议最终要协商的 SA.当 IKE 为 IPSec 协商时可以称为 IPSecSA，保证 AH 或者 ESP 的安全通信。第二阶段的安全由第一阶段的协商结果来保证。

3 SSL/TLS

IPSec 通过安全关联实现了网络层 IP 分组两端的安全传输，本小节介绍的 SSL/TLS 通过建立安全连接实现数据在两个进程端口间的安全传输。

传输层安全协议（Transport Layer Security，TLS）的前身是安全套接层协议（Secure Sockets Layer，SSL）最初由网景公司设计和开发，目的是为电子商务应用提供一个安全的环境。由于 SSL 在网络安全中的重要性，1996 年互联网工作组（Internet Engineering Task Force，IETF）成立了专门的工作组并与 1999 年在 SSL v3 的基础上，推出了传输层安全协议 TLS v1，并与 2006，2008 年分别发布 TLS v1.1、TLS v1.2，被当前互联网普遍应用。

为了表示 TLS 和 SSL 的历史关系，常写作 SSL/TLS。

下面介绍 TLS 的协议结构：

TLS 握手协议	TLS 改变密码规范协议	TLS 报警协议	上层应用数据
TLS 记录协议			
TCP			
IP			

TLS 记录协议用于封装上层协议消息，实现上层通信的源端鉴别、保密性和完整性；

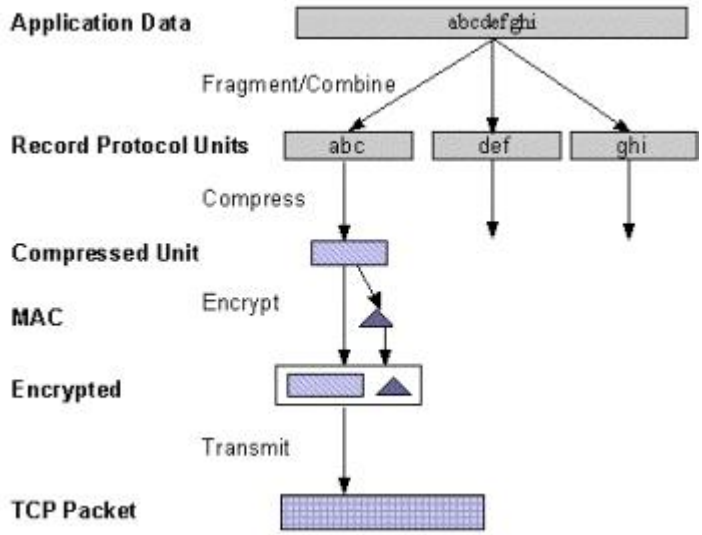
TLS 握手协议实现了身份鉴别和安全参数的协商，客户端和服务端通过 TLS 记录协议传输数据前，需要通过 TLS 握手协议完成双向身份鉴别，并协商 TLS 压缩算法、加密算法、完整性验证算法（MAC）、加密密钥、MAC 密钥等安全参数；

TLS 改变密码规范协议实现协商完安全参数后通知对方使用新约定的安全参数；

TLS 报警协议用于传输错误信息；

下面介绍 TLS 记录协议的封装过程和格式。

TLS 记录协议封装过程如下图所示：



上层数据报文经过了分组、压缩、计算校验、加密、添加首部，5 个过程保证了 TLS 对上层数据的安全传输。

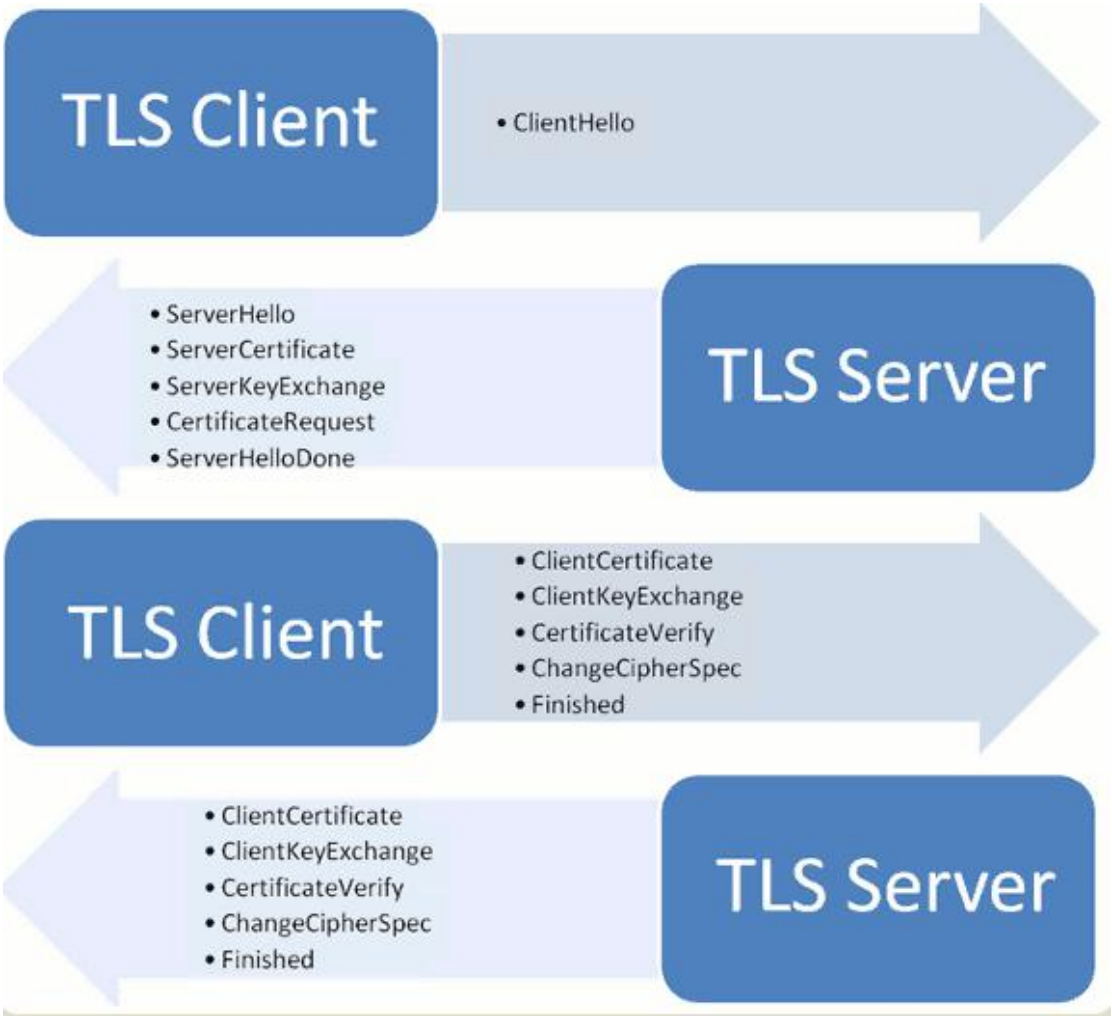
TLS 记录协议格式如下表：

内容类型	主版本号	次版本号	压缩数据长度
压缩后的数据			
MAC			

内容类型：上层消息类型，如 TLS 握手协议、HTTP 协议；

主版本号；
次版本号；
压缩数据长度：加密前上层消息的长度；
压缩后的数据；
MAC：源端鉴别、完整性验证。

下面介绍 TLS 握手过程：



"握手阶段"涉及四次通信，我们一个个来看。需要注意的是，"握手阶段"的所有通信都是明文的。

1. 客户端发出请求（ClientHello）

首先，客户端（通常是浏览器）先向服务器发出加密通信的请求，这被叫做 ClientHello 请求。

在这一步，客户端主要向服务器提供以下信息。

（1） 支持的协议版本，比如 TLS 1.0 版。

（2） 一个客户端生成的随机数，稍后用于生成"对话密钥"。

（3） 支持的加密方法，比如 RSA 公钥加密。

（4） 支持的压缩方法。

这里需要注意的是，客户端发送的信息之中不包括服务器的域名。也就是说，理论上服务器只能包含一个网站，否则会分不清应该向客户端提供哪一个网站的数字证书。这就是为

什么通常一台服务器只能有一张数字证书的原因。

对于虚拟主机的用户来说，这当然很不方便。2006 年，TLS 协议加入了一个 Server Name Indication 扩展，允许客户端向服务器提供它所请求的域名。

2. 服务器回应（SeverHello）

服务器收到客户端请求后，向客户端发出回应，这叫做 SeverHello。服务器的回应包含以下内容。

（1） 确认使用的加密通信协议版本，比如 TLS 1.0 版本。如果浏览器与服务器支持的版本不一致，服务器关闭加密通信。

（2） 一个服务器生成的随机数，稍后用于生成"对话密钥"。

（3） 确认使用的加密方法，比如 RSA 公钥加密。

（4） 服务器证书。

除了上面这些信息，如果服务器需要确认客户端的身份，就会再包含一项请求，要求客户端提供"客户端证书"。比如，金融机构往往只允许认证客户连入自己的网络，就会向正式客户提供 USB 密钥，里面就包含了一张客户端证书。

3. 客户端回应

客户端收到服务器回应以后，首先验证服务器证书。如果证书不是可信机构颁布、或者证书中的域名与实际域名不一致、或者证书已经过期，就会向访问者显示一个警告，由其选择是否还要继续通信。

如果证书没有问题，客户端就会从证书中取出服务器的公钥。然后，向服务器发送下面三项信息。

（1） 一个随机数。该随机数用服务器公钥加密，防止被窃听。

（2） 编码改变通知，表示随后的信息都将用双方商定的加密方法和密钥发送。

（3） 客户端握手结束通知，表示客户端的握手阶段已经结束。这一项同时也是前面发送的所有内容的 hash 值，用来供服务器校验。

上面第一项的随机数，是整个握手阶段出现的第三个随机数，又称"pre-master key"。有了它以后，客户端和服务器就同时有了三个随机数，接着双方就用事先商定的加密方法，各自生成本次会话所用的同一把"会话密钥"。

至于为什么一定要用三个随机数，来生成"会话密钥"：

不管是客户端还是服务器，都需要随机数，这样生成的密钥才不会每次都一样。由于 SSL 协议中证书是静态的，因此十分有必要引入一种随机因素来保证协商出来的密钥的随机性。

对于 RSA 密钥交换算法来说，pre-master-key 本身就是一个随机数，再加上 hello 消息中的随机，三个随机数通过一个密钥导出器最终导出一个对称密钥。

pre master 的存在在于 SSL 协议不信任每个主机都能产生完全随机的随机数，如果随机数不随机，那么 pre master secret 就有可能被猜出来，那么仅适用 pre master secret 作为密钥就不合适了，因此必须引入新的随机因素，那么客户端和服务器加上 pre master secret 三个随机数一同生成的密钥就不容易被猜出了，一个伪随机可能完全不随机，可是是三个伪随机就十分接近随机了，每增加一个自由度，随机性增加的可不是一。

此外，如果前一步，服务器要求客户端证书，客户端会在这一步发送证书及相关信息。

4. 服务器的最后回应

服务器收到客户端的第三个随机数 pre-master key 之后，计算生成本次会话所用的"会话密钥"。然后，向客户端最后发送下面信息。

（1）编码改变通知，表示随后的信息都将用双方商定的加密方法和密钥发送。

（2）服务器握手结束通知，表示服务器的握手阶段已经结束。这一项同时也是前面发

送的所有内容的 hash 值，用来供客户端校验。

至此，整个握手阶段全部结束。接下来，客户端与服务器进入加密通信，就完全是使用普通的 HTTP 等应用层协议，只不过用"会话密钥"加密内容。

4 HTTPS

超文本传输协议 HTTP 协议被用于在 Web 浏览器和网站服务器之间传递信息。HTTP 协议以明文方式发送内容，不提供任何方式的数据加密，如果攻击者截取了 Web 浏览器和网站服务器之间的传输报文，就可以直接读懂其中的信息，因此 HTTP 协议不适合传输一些敏感信息，比如信用卡号、密码等。

为了解决 HTTP 协议的这一缺陷，需要使用另一种协议：基于 SSL/TLS 的 HTTP（Hyper Text Transfer Protocol over Secure Socket Layer，HTTPS）是在 TCP 基础上建立 TLS 安全连接，通过 TLS 安全连接实现 HTTP 消息的保密性、完整性和源端鉴别。这个系统的最初研发由网景公司(Netscape)于 1994 年进行，并内置于其浏览器 Netscape Navigator 中，提供了身份验证与加密通讯方法。现在它被广泛用于万维网上安全敏感的通讯，例如交易支付方面。

HTTPS 和 HTTP 的区别主要为以下四点：

一、https 协议需要到 ca 申请证书，一般免费证书很少，需要交费。

二、http 是超文本传输协议，信息是明文传输，https 则是具有安全性的 ssl 加密传输协议。

三、http 和 https 使用的是完全不同的连接方式，用的端口也不一样，前者是 80，后者是 443。

四、http 的连接很简单，是无状态的；HTTPS 协议是由 SSL+HTTP 协议构建的可进行加密传输、身份认证的网络协议，比 http 协议安全。

HTTP
TLS
TCP
IP

下面介绍 HTTPS 的基本运行过程：

因为 HTTPS 是基于 SSL/TLS 的所以 HTTPS 运行过程与 TLS 运行过程基本相同：

1、客户端首先会将自己支持的加密算法，打包告诉服务器端。

2、服务器端从客户端发来的加密算法中，选出一组加密算法和 HASH 算法并将自己的身份信息以证书的形式发回给客户端。而证书中包含了网站的地址，加密用的公钥，以及证书的颁发机构等；

这里，服务器就将自己用来加密用的公钥一同发还给客户端，而私钥则服务器保存着，用户解密客户端加密过后的内容。

3、客户端收到了服务器发来的数据包后，会做这么几件事情：

1）验证一下证书是否合法。一般来说，证书是用来标示一个站点是否合法的标志。如果说该证书由权威的第三方颁发和签名的，则说明证书合法。

2）如果证书合法，或者客户端接受和信任了不合法的证书，则客户端就会随机产生一串序列号，使用服务器发来的公钥进行加密。这时候，一条返回的消息就基本就绪。

3）最后使用服务器挑选的 HASH 算法，将刚才的消息使用刚才的随机数进行加密，生成相应的消息校验值，与刚才的消息一同发还给服务器。

4、服务器接受到客户端发来的消息后，会做这么几件事情：

1）使用私钥解密上面第 2）中公钥加密的消息，得到客户端产生的随机序列号。

2）使用该随机序列号，对该消息进行加密，验证的到的校验值是否与客户端发来的一致。如果一致则说明消息未被篡改，可以信任。

3）最后，使用该随机序列号，加上之前第 2 步中选择的加密算法，加密一段握手消息，发还给客户端。同时 HASH 值也带上。

5、客户端收到服务器端的消息后，接着做这么几件事情：

1）计算 HASH 值是否与发回的消息一致

2）检查消息是否为握手消息

6、握手结束后，客户端和服务器端使用握手阶段产生的随机数以及挑选出来的算法进行对称加解密的传输。

因为非对称算法的效率对比起对称算法来说，要低得多得多；因此使用非对称加密算法进行数据传输往往只用在 HTTPS 的握手阶段。

参考资料：

1.曾湘黔.网络安全技术.北京：清华大学出版社，2013.

2.沈鑫剡，等.网络安全.北京：清华大学出版社，2017.

3.胡道元，闵京华.网络安全（第二版）.北京：清华大学出版社，2008.

4.阮一峰，SSL/TLS 协议运行机制的概述，2014.

本章小结

本章介绍了

习题

1.