3.2 IPSec

IPSec 是 IP Security 的缩写，顾名思义，就是为了 IP 网络的安全而服务的。按道理来说，我们得首先知道网络上有哪些不安全的行为，才能评估 IPSec 这项技术能在多大程度上保护网络的安全。不过这个超出本书/本章节的范围（也超出了笔者的知识范围，^_^）。

IPSec 比较复杂（内容比较多），笔者纠结了一番，还是决定采用直奔主题的方式来描述。

首先 IPSec 协议不是一个协议，而是一个协议族，或者说是一套 IP Security 体系架构。它包括如下几部分：

l 安全服务协议，包括 AH, ESP（不必纠结是什么，后面会讲述）

l 密钥交换协议，密钥是与加密算法相关的一个概念，本文不会再涉及加密算法相关内容（这个不是超出笔者的知识范围，而是超出笔者的能力范围），所以关于密钥的相关概念，请您自行查阅

l 安全相关的一些加密算法，IPSec 自己应该是不发明加密算法，它只是引用相关算法

IPSec 提供网络安全的方法有两种：认证和加密。下面我们就从这个主题切入，逐步剖析 IPSec。

3.2.1 AH/Authentication Header

3.2.1.1 概述

AH，Authentication Header，验证头部，是一个协议。验证谁的头部？验证 IP 报文的头部。那么，AH 是如何验证 IP 报文头部，又是如何解决相应的安全问题呢？先不用着急，我们先来看看报文头，有一个直观的认识。

下图是一个传统的 IP 报文格式：

图 1 传统 IP 报文格式

IPSec 的 AH，有两种方式，一个是传输模式，一个是隧道模式。这里呢，我们暂时先不要管这个概念。我们暂时先以比较简单的“传输模式”为例进行讲解。

所谓传输模式，我们暂时不管其他细节，就是先如此理解：在 IP Head 后面，嵌入一个 AH Head（注意，这里的 AH Header 中的 Header，与 AH/Authentication Header，不是同一个 Header），如下图所示：

图 2 IP/AH 报文格式（传输模式）

【说明：图中在 AH Header 后面以 TCP/UDP Header 为例画的图，其实也可以其他所有传统 IP 报文中的所有合法 Header，比如 ICMP Header。】

关于 AH Header 里面的内容，笔者后面再说，这里只须指明 AH Header 里面的一个关键字段，如下图所示：

图 3 AH Header

我们只须关注“验证数据”这个字段。这个验证数据就是为了保证 IP 报文的安全性（保证某些方面的安全性）。

3.2.1.2 AH 的基本原理

AH 保证 IP 报文的安全性（部分安全性，不是全部）的最核心的原理是通过 AH Header 中的“验证数据”这个字段来完成的。（下文是以传输模式为例讲述。隧道模式，会在后面的章节中补充说明）

【说明：以下是一个原理性的概述，不代表实际的软件实现就是这么一个过程】

发送方：

1) 在传统的 IP 报文格式中直接插入 AH Header

2) 相应字段赋值（不关键）

3) 计算 HMAC，填入 AH Header 中的“验证数据”字段（这个非常重要，下文会继续详细描述）

4) 发送报文

接收方：

1) 接收报文

2) 计算接收到报文的 HMAC

3) 比较计算的 HMAC 与发送报文中的 AH Header 中的“验证数据”字段：如果相同，则说明报文是完整的，没有被篡改（同样，下文会向西描述 HMAC 的计算）；如果不同，则说明报文被篡改（后续是报文丢弃，还是通知发送方，本文不涉及）

我们以一个图来标示上述过程：

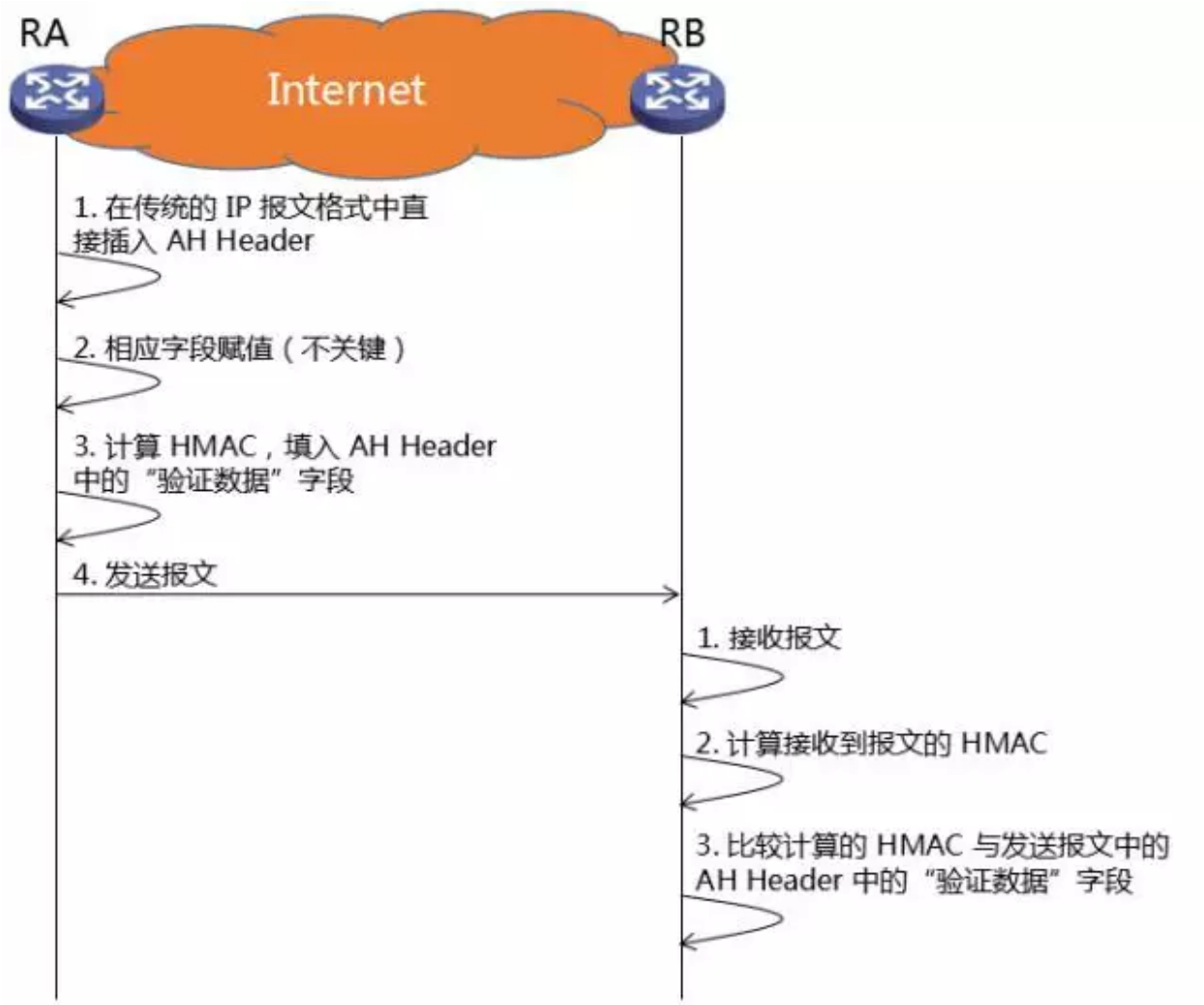

图 4 IP/AH 的发送与验证过程

那么，上文中提到的 HMAC 是什么意思呢？它又是如何计算的？又是如何保证报文的完整性（防止篡改）的呢?

MAC 不是 TCP/IP 中的 MAC 地址中的那个 MAC，它是 Message Authentication Codes 的缩写，即“报文验证码”。如果它的算法采用 HASH 来构造，那么就称作 HMAC。HMAC 的算法有：HMAC-MD5、HMAC-SHA1、HMAC-RIPEMD-160 等。本文不描述这些算法，请读者自行查阅。不过这些算法可以简单这样理解：

HMAC = Hash（data， key）

l Hash 代表一个算法，比如：HMAC-MD5、HMAC-SHA1、HMAC-RIPEMD-160。它将数据通过一个密钥（key），通过一个算法，就算出一个 hash 值。

l key 代表一个密钥

l HMAC 就是计算结果

HMAC 计算中，其对应的 data 是什么呢？如下图所示：（以 AH 传输模式为例，隧道模式后面会有描述）

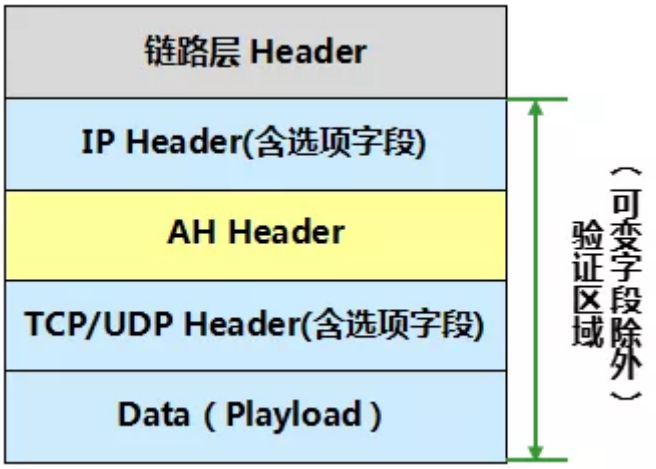

图 5 HMAC 算法中的 data

上图中标识的“验证区域”就是 HMAC 算法中的 data。而图中的可变字段除外的含义是，针对 IP Header， AH Header， TCP Header 中的可变（可选字段），HMAC 是忽略其值得，全是以0替代（不要问我为什么，我也不知道，^_^）。下图列出了 IP Header， AH Header，图中标识出了哪些是可变字段：

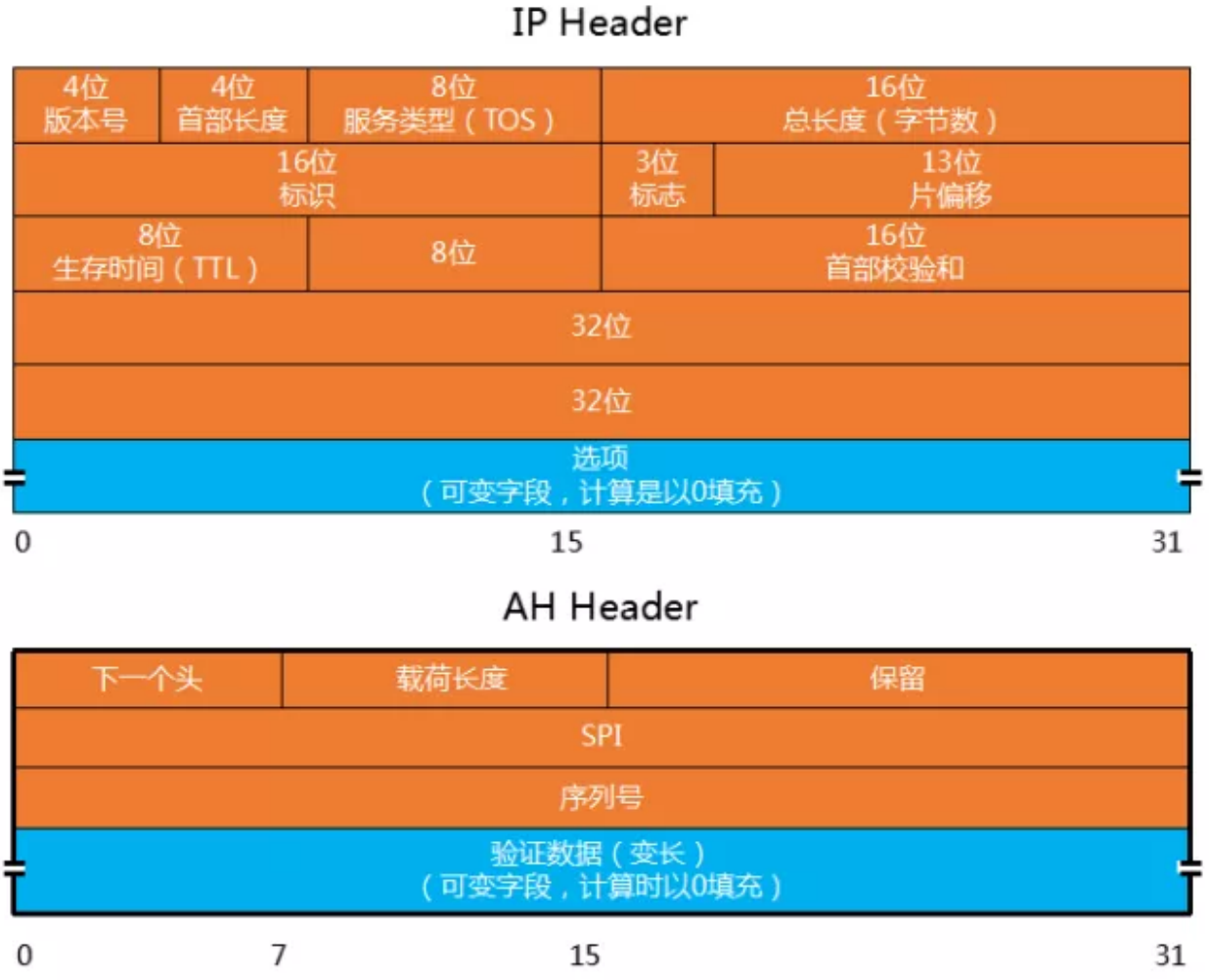

图 6 IP Header， AH Header 中的可变字段

综合以上描述，我们再以图示的方式，对以上做个总结，以更加好理解。

发送方：（再次强调，图中只是以 TCP/UDP 为例而已，实际上可以是任何合法协议）

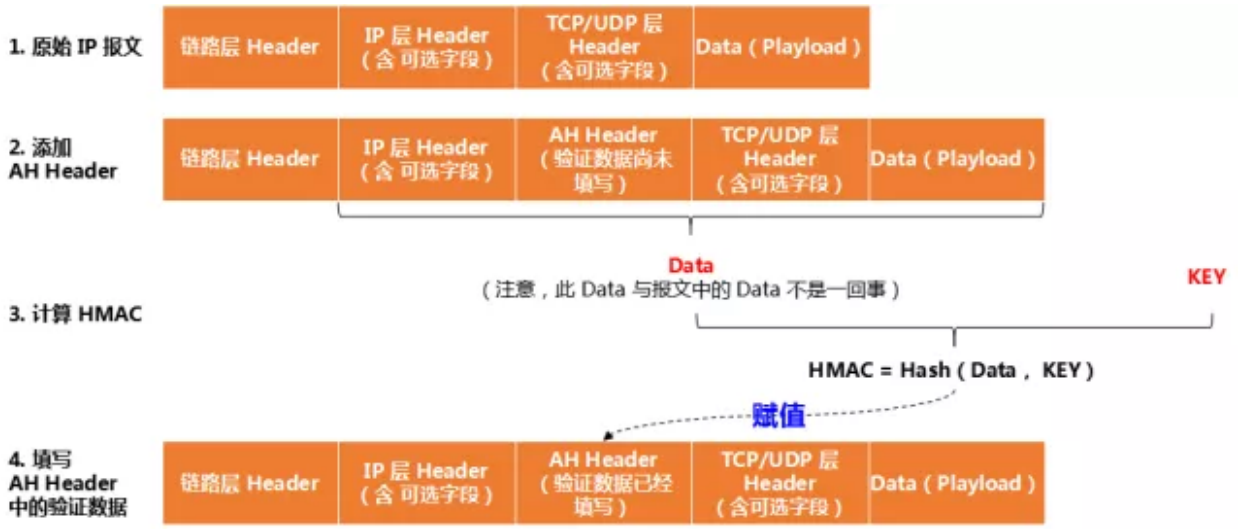

图 7 IP/AH 发送方的操作过程（传输模式）

接收方：

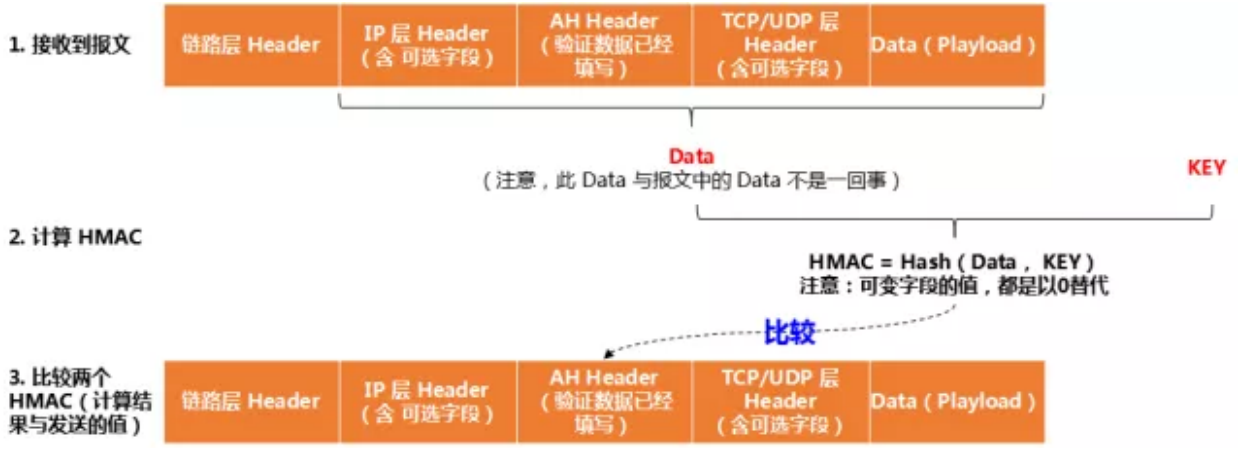

图 8 IP/AH 接收方的操作过程（传输模式）

以上讲述了 AH 的基本原理，那么 AH 是如何保证通信的安全的呢?

3.2.1.3 AH 如何保障通信的安全

首先说明，AH 所谓**保障**通信的安全，是指在某些方面保障，并不是全面保障。而且我们注意这个词：保障！是的，是“保障”，而不是“保证”。没人能够“保证”它能100%的“保障”通信的安全!

AH 能够在如下几个方面保障通信的安全。

（一）数据的完整性验证

如下图所示，在数据的发送过程中，数据有可能被篡改（到底是如何篡改的，用了啥黑科技，呵呵，sorry，俺不知道，反正网上大神多的很，防不胜防！）：

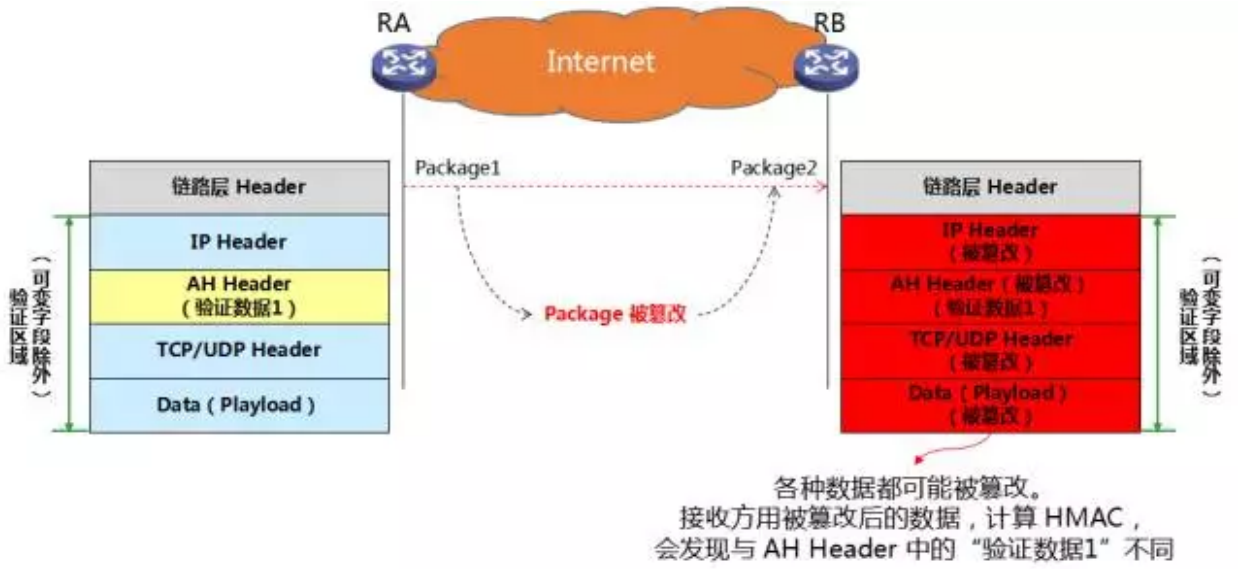

图 9 数据发送过程被篡改

根据上一小节的描述，我们知道，只要数据被篡改，那么接收方就会知道。AH 就是利用这个原理来保障数据的完整性。

这个时候，有一个问题，不知道您注意到没有：那位篡改数据的大神，为什么不把 AH Header 中的“验证数据”给篡改了——按照它篡改的其他数据，根据相应的算法，同时把“验证数据”一起给篡改了，那不就不会被发现了吗？

这就要从我们前文说的算法（简化描述）来理解了：

HMAC = Hash（data， key）

是的，大神要想篡改 HMAC（验证数据），就得知道密钥：key！

只要保证密钥的安全，就能保证大神无法篡改“验证数据”，从而保证数据的完整性。（如何保证密钥的安全，这是另外一个课题。本文会在后面有所涉及。这里，请读者暂时不要纠结这个话题。）

【特别说明：没有什么算法能保证数据绝对安全！只是目前来说，IPSec 所采用的算法，比较安全而已！或者说已经是当前最安全的几种算法了！】

（二）数据源身份认证

这句话其实有点绕，其含义其实并**不**是：保障发送数据方的身份的真实性，即不是别人冒充你发送的。

网络上标识你的身份的，就是“用户名/密码”（广义的“用户名/密码”）。你的“用户名/密码”如果泄露了，对不起，这不是 IPSec 能解决的问题（所以，从这个角度来说，不要把 IPSec 神话，有时候别人冒充你，根本不需要什么所谓的黑科技）。

那么，“数据源身份认证”到底是什么意思？其实个人觉得，这一条不应该写，因为跟上一条重复。没办法，可能是当初 IPSec 处于宣传需要，硬加的一条优点，也可能是笔者水平差，没有理解。

暂时也不管我到底有没有真正理解，就先按照我的理解来吧，^_^。

根据第一条 AH 提供“数据的完整性验证”服务中的描述，因为：

HMAC = Hash（data， key）

它是综合了数据（data）和密钥（key）一起做的 Hash，所以，只要密钥没有被泄露，实际上你是无法篡改数据的（一篡改，就会被发现）。但是，当密钥泄露了，这一切都无从谈起。

这个时候，笔者觉得 IPSec 的宣传，就有点玩弄词汇了：

（1）假设密钥没有被泄露

（2）通过算法 HMAC = Hash（data， key），就能验证数据的完整性

（3）验证出数据的完整性后，就说明数据没有被篡改，也说明密钥没有被泄露

（4）因为密钥没有被泄露，就说明“你就是你”，因为别人无法知道密钥（因为一开始就假设密钥没有被泄露）

（5）因为证明了“你就是你”，所以就做到了“数据源身份认证”

Oh， My God！ 这是什么神逻辑啊！

暂时不吐槽，也许是我理解错了！！！

（三）防重放攻击

关于重放攻击，请允许笔者摘抄一下百度百科：

重放攻击（Replay Attacks）又称重播攻击、回放攻击或新鲜性攻击（Freshness Attacks），是指攻击者发送一个目的主机已接收过的包，特别是在认证的过程中，用于认证用户身份所接收的包，来达到欺骗系统的目的，主要用于身份认证过程，破坏认证的安全性。

它是一种攻击类型，这种攻击会不断恶意或欺诈性地重复一个有效的数据传输，重放攻击可以由发起者拦截并重复发该数据到目的主机进行。攻击者利用网络监听或者其他方式盗取认证凭据，一般是cookies或者一些的认证session会话，进行一定的处理后，再把它重新发给认证服务器。从这个解释上理解，加密可以有效防止明文数据被监听，但是却防止不了重放攻击。重放攻击任何网络通讯过程中都可能发生。重放攻击是计算机世界黑客常用的攻击方式之一。

……

防御方法之一就是“序号”法：

通信双方通过消息中的序列号来判断消息的新鲜性

要求通信双方必须事先协商一个初始序列号，并协商递增方法

【注：以上摘自http://baike.baidu.com/link?url=0MQzi4ijU8QVrrhQy1DNwXs5dozEIm6RSXnIlgOYtk6wAMaEGZHHz-Y5ioH6LDWzZKvNEVY9VInsYnHE9BsB5ZCwLsT8pQ7z0XLbtXk3VGB7LHYFJDEwNx7bGfZ0_oGh】

而 AH Header 中正好有一个字段：“序列号”！

AH 就是利用这个字段来做到防重放攻击。具体这个字段的初始值如何协商，后续如何协商递增，与本章节关系不大，笔者暂时不做描述（后续如果有时间，再补上）。

以上就是 AH 在保障通信安全方面所能提供的功能（服务）。我们再做一个总结：

（1）数据的完整性验证

（2）数据源身份认证

（3）防重放攻击

在以上的描述中，笔者一直有意忽略一个问题：发送方和接收方是如何知道对方的密钥的？或者是如何知道使用一个相同的密钥的（HMAC 就是共享密钥，即双方使用一个相同

的密钥）？