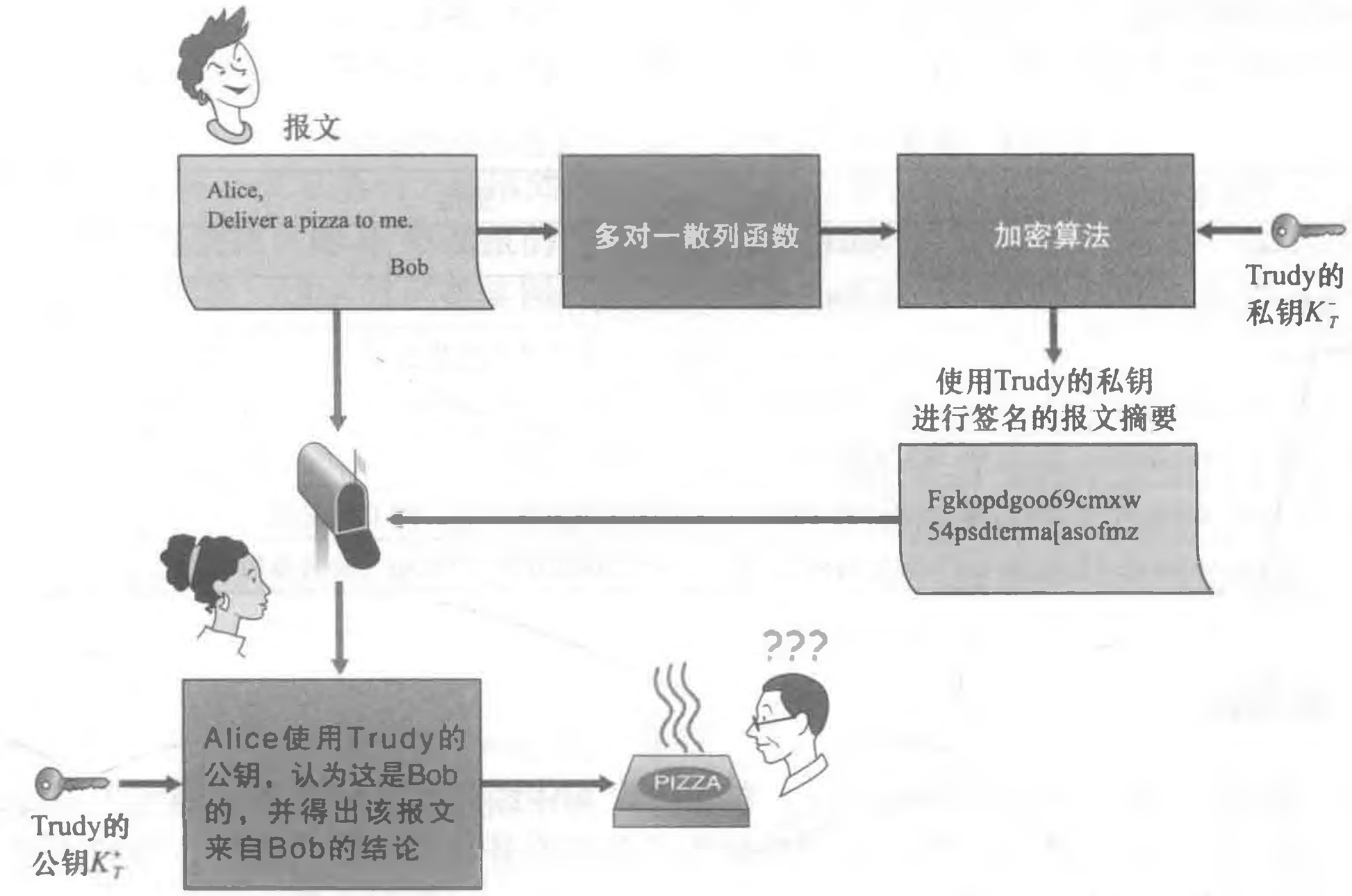

图 8-13　Trudy 用公钥密码冒充 Bob

1）CA 证实一个实体（一个人、一台路由器等）的真实身份。如何进行认证并没有强制的过程。当与一个 CA 打交道时，一方必须信任这个 CA 能够执行适当的严格身份验证。例如，如果 Trudy 走进名为 Fly-by-Night 的认证中心并只是宣称“我是Alice”，就可以得到该机构颁发的与 Alice 的身份相关联的证书的话，则人们不会对 Fly-by-Night 认证中心所签发的公钥证书有太多的信任。另一方面，人们可能愿意（或不愿意!）信任某个 CA，如果这个 CA 是联邦或州计划的一部分的话。你对与公钥相关联的身份的信任程度，仅能达到你对 CA 及其身份验证技术的信任程度。我们编织了多么混乱的信任关系网啊!

2）一旦 CA 验证了某个实体的身份，这个 CA 会生成一个将其身份和实体的公钥绑定起来的**证书**（certificate）。这个证书包含这个公钥和公钥所有者全局唯一的身份标识信息（例如，一个人的名字或一个 IP 地址）。由 CA 对这个证书进行数字签名。这些步骤显示在图 8-14 中。

我们现在来看怎样使用认证来对抗“比萨订购”中的恶作剧者（如 Trudy）和其他意外情况。Bob 下订单的同时，他也发送了其 CA 签署的证书。Alice 使用 CA 的公钥来核对 Bob 证书的合法性并提取 Bob 的公钥。

国际电信联盟（International Telecommunication Union，ITU）和 IETF 都研发了用于 CA 的系列标准。ITU X.509［ITU 2005a］规定了证书的鉴别服务以及特定语法。［RFC 1422］描述了安全

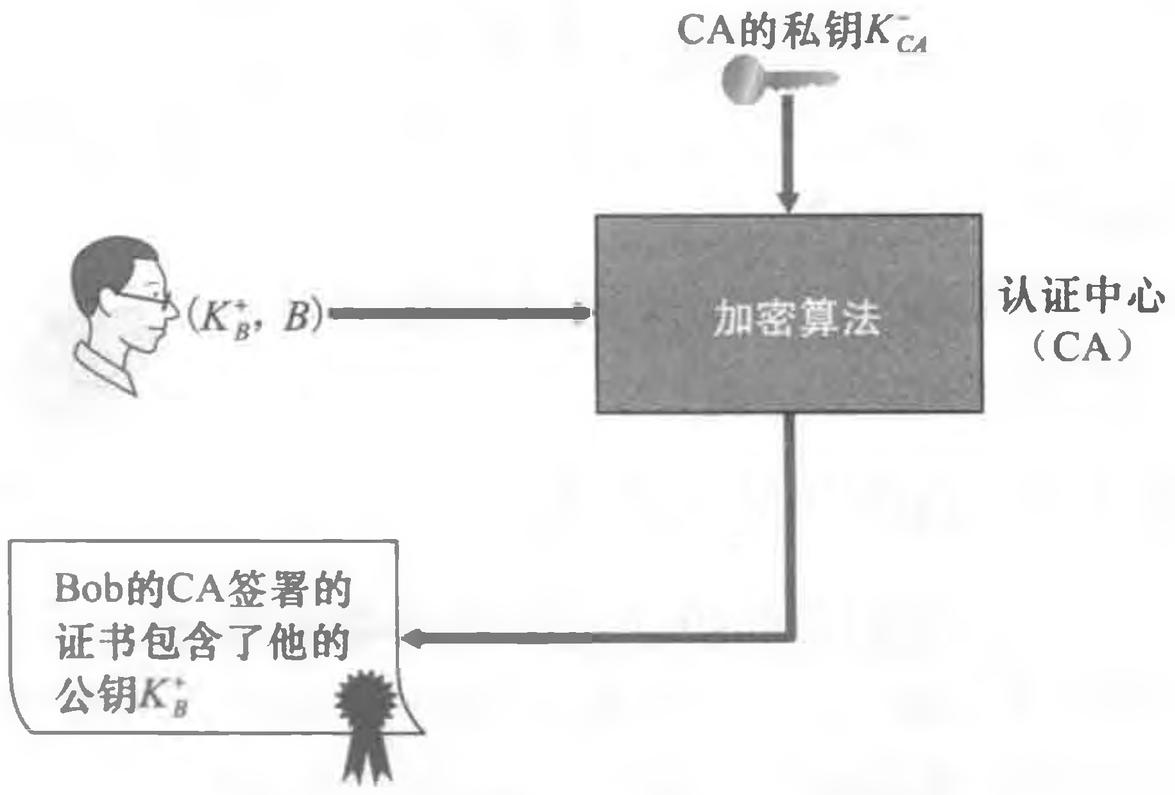

图 8-14　Bob 获得一份来自 CA 的证书

因特网电子邮件所用的基于 CA 的密钥管理。它和 X. 509 兼容，但比 X. 509 增加了密钥管理体系结构的创建过程和约定内容。表 8-4 显示了一份证书中的某些重要字段。

表 8-4　在 X. 509 和 RFC 1422 公钥证书中的部分字段

字段名	描述
版本（Version）	X. 509 规范的版本号
序列号（Serial number）	CA 发布的证书的独特标识符
签名（Signature）	规定了由 CA 所用的对该证书签名的算法
颁发者名称（Issuer name）	发行该证书的 CA 的标识符，用的是区别名（DN）格式［RFC 4514］
有效期（Validity period）	证书合法性开始和结束的时间范围
主题名（Subject name）	其公钥与该证书相联系的实体标识符，用 DN 格式
主题公钥（Subject public key）	该主题的公钥以及该公钥使用的公钥算法（及其参数）的指示

8.4　端点鉴别

端点鉴别（end-point authentication）就是一个实体经过计算机网络向另一个实体证明其身份的过程，例如一个人向某个电子邮件服务器证明其身份。作为人类，我们通过多种方式互相鉴别：见面时我们互相识别对方的面容，打电话时我们分辨对方的声音，海关的检查官员通过护照上的照片对我们进行鉴别。

在本节中，我们讨论经网络通信的双方如何能够鉴别彼此。此处我们重点关注当通信实际发生时鉴别“活动的”实体。一个具体的例子是一个用户向某电子邮件服务器鉴别他或她自己。这与证明在过去的某点接收到的报文确实来自声称的发送方稍有不同，如 8.3 节所述。

当经网络进行鉴别时，通信各方不能依靠生物信息比如外表、声波纹等进行身份鉴别。的确，我们会在后面的实例研究中看到，诸如路由器、客户/服务器进程等网络元素通常必须相互鉴别。此处，鉴别应当在报文和数据交换的基础上，作为某**鉴别协议**（authentication protocol）的一部分独立完成。鉴别协议通常在两个通信实体运行其他协议（例如，可靠数据传输协议、路由选择信息交换协议或电子邮件协议）之前运行。鉴别协议首先建立相互满意的各方的标识；仅当鉴别完成之后，各方才继续下面的工作。

同第 3 章中我们阐释可靠数据传输协议（rdt）的情况类似，我们发现阐释各种版本的鉴别协议——我们将称为 ap（authentication protocol）——是有启发的，并随着我们学习的深入指出各个版本的漏洞。（如果你喜欢这种逐步式的设计演化，你也许喜欢看［Bryant 1988］，这本书虚构了开放网络鉴别系统的设计者间的故事，以及他们对许多相关奇妙问题的发现。）

我们假设 Alice 要向 Bob 鉴别她自己的身份。

8.4.1　鉴别协议 ap1.0

也许我们能够想象出的最简单的鉴别协议就是：Alice 直接发送一个报文给 Bob，说她就是 Alice。这个协议如图 8-15所示。这个协议的缺陷是明显的，即 Bob 无法判断发送报文“我是

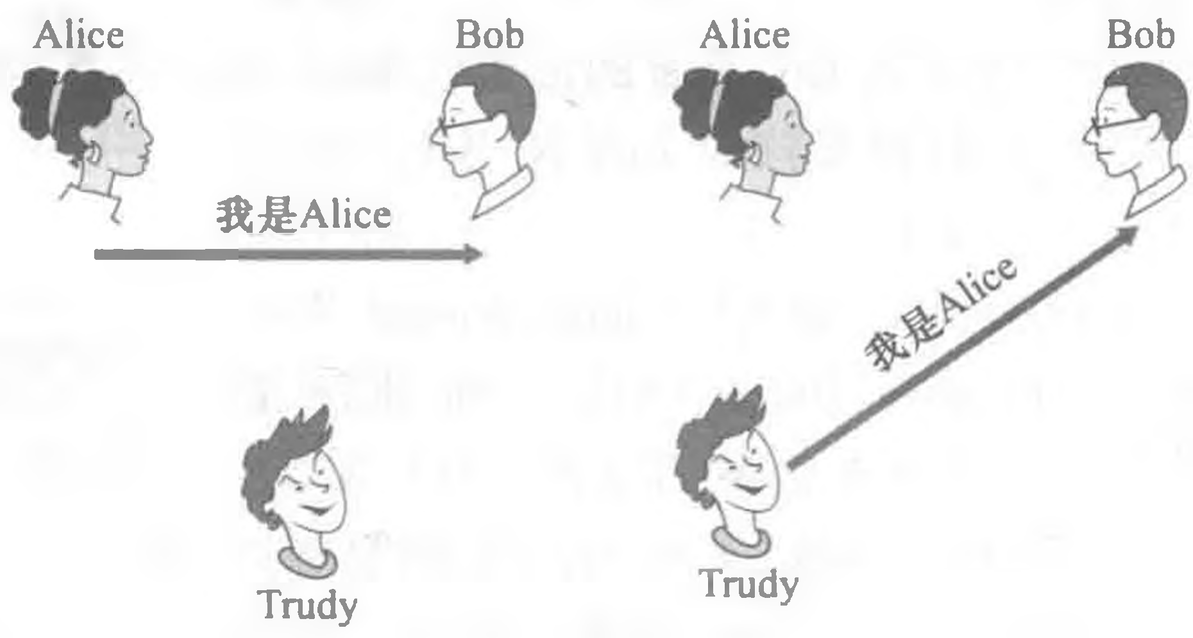

图 8-15　协议 ap1.0 和一种失败的情况

Alice”的人确实就是 Alice。例如，Trudy（入侵者）也可以发送这样的报文。

8.4.2　鉴别协议 ap2.0

如果 Alice 有一个总是用于通信的周知网络地址（如一个 IP 地址），则 Bob 能够试图通过验证携带鉴别报文的 IP 数据报的源地址是否与 Alice 的周知 IP 地址相匹配来进行鉴别。在这种情况下，Alice 就可被鉴别了。这可能阻止对网络一无所知的人假冒 Alice，但是它却不能阻止决定学习本书的学生或许多其他人！

根据我们学习的网络层和数据链路层的知识，我们就会知道做下列事情并不困难（例如，如果一个人能够访问操作系统代码并能构建自己的操作系统内核——比如 Linux 和许多其他免费可用的操作系统）：生成一个 IP 数据报，并在 IP 数据报中填入我们希望的任意源地址（比如 Alice 的周知 IP 地址），再通过链路层协议把生成的数据报发送到第一跳路由器。此后，具有不正确源地址的数据报就会忠实地向 Bob 转发。这种方法显示在图 8-16中，它是 IP 哄骗的一种形式。如果 Trudy 的第一跳路由器被设置为只转发包含 Trudy 的 IP 源地址的数据报，就可以避免 IP 哄骗［RFC 2827］。然而，这一措施并未得到广泛采用或强制实施。Bob 可能因为假定 Trudy 的网络管理员（这个管理员可能就是 Trudy 自己）已经配置 Trudy 的第一跳路由器，使之只能转发适当地址的数据报而被欺骗。

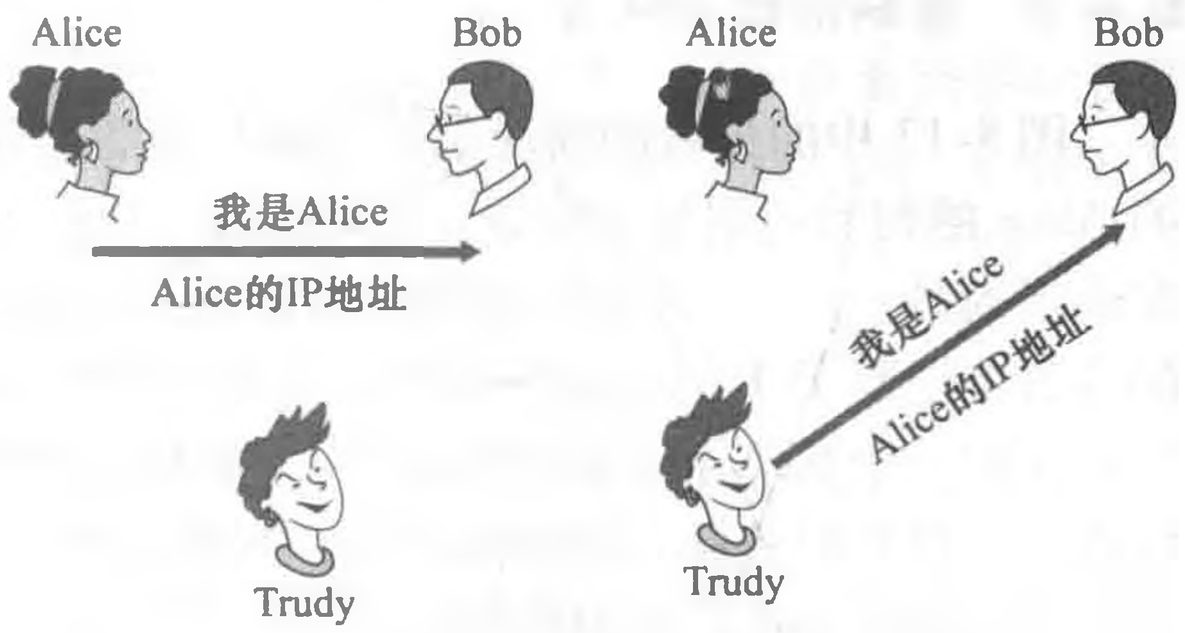

图 8-16　协议 ap2.0 和一种失败的情况

8.4.3　鉴别协议 ap3.0

进行鉴别的一种经典方法是使用秘密口令。口令是鉴别者和被鉴别者之间的一个共享秘密。Gmail、Telnet、FTP 和许多其他服务使用口令鉴别。在协议 ap3.0 中，Alice 因此向 Bob 发送其秘密口令，如图 8-17 所示。

由于口令的广泛使用，我们也许猜想协议 ap3.0 相当安全。如果这样想，我们就错了！这里的安全性缺陷相当明显：如果 Trudy 窃听了 Alice 的通信，则可得到 Alice 的口令。为了使你认识到这种可能性，考虑这样的事实，当你 Telnet 到另一个机器上并登录时，登录口令未加密就发送到了 Telnet 服务器。连接到 Telnet 客户或服务器 LAN 的某个人都可能嗅探（sniff）（读并存储）在局域网上传输的所有数据分组，并因此窃取到该注册口令。实际上，这是一种窃取口令的周知方法（例如，参见［Jimenez 1997］）。这样的威胁显然是真实存在的，所以协议 ap3.0 明显也不可行。

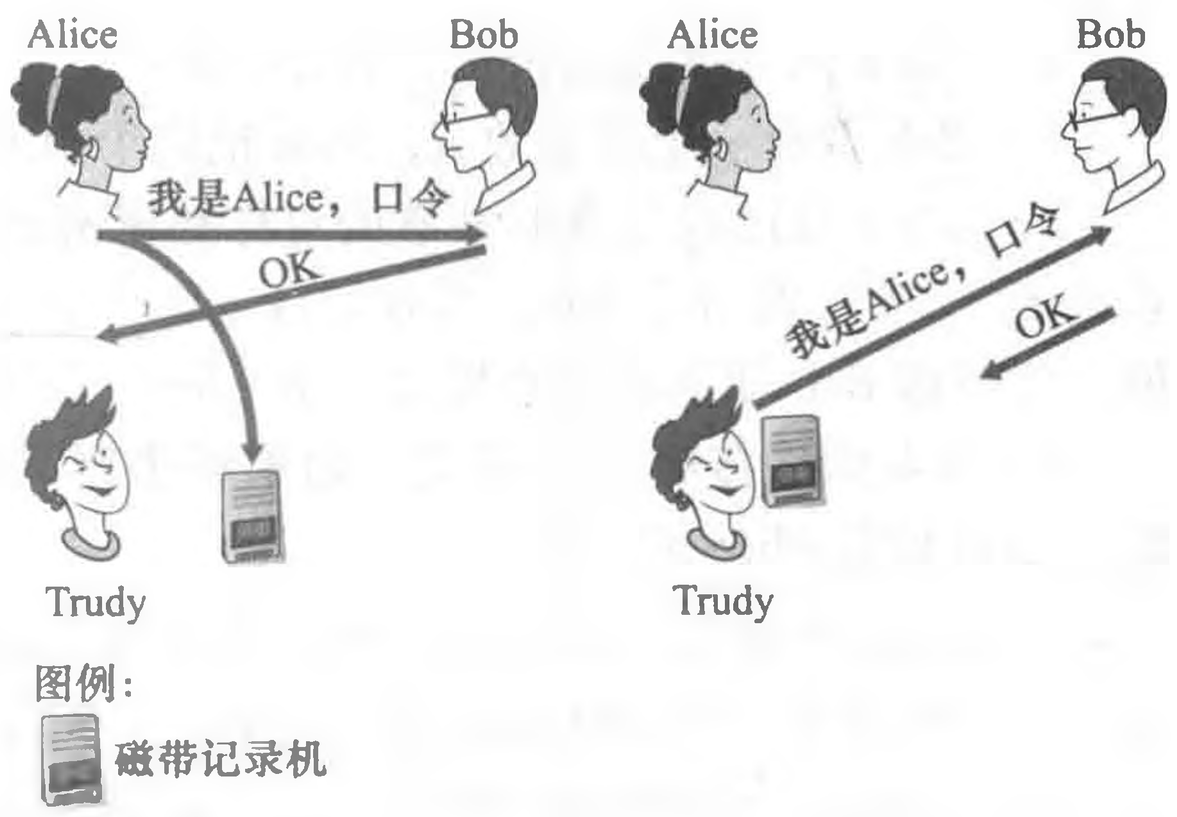

图 8-17　协议 ap3.0 和一种失败的情况

8.4.4　鉴别协议 ap3.1

我们完善协议 ap3.0 的下一个想法自然就是加密口令了。通过加密口令，我们能够防止 Trudy 得知 Alice 的口令。如果我们假定 Alice 和 Bob 共享一个对称秘密密钥 K_{A-B}，则 Alice 可以加密口令，并向 Bob 发送其识别报文“我是 Alice”和加密的口令。Bob 则解密口令，如果口令正确则鉴别了 Alice。因为 Alice 不仅知道口令，而且知道用于加密口令的共享秘密密钥值，Bob 才可以轻松地鉴别 Alice 的身份。我们称这个协议为 ap3.1。

尽管协议 ap3.1 确实防止了 Trudy 得知 Alice 的口令，此处使用密码术并不能解决鉴别问题。Bob 受制于**回放攻击**（playback attack）：Trudy 只需窃听 Alice 的通信，并记录下该口令的加密版本，并向 Bob 回放该口令的加密版本，以假装她就是 Alice。协议 ap3.1 中加密口令的使用，并未使它比图 8-17 中的协议 ap3.0 的局面有明显改观。

8.4.5　鉴别协议 ap4.0

图 8-17 中的失败的情况是因为 Bob 不能区分 Alice 的初始鉴别报文和后来入侵者回放的 Alice 的初始鉴别报文所致。也就是说，Bob 无法判断 Alice 是否还活跃（即当前是否还在连接的另一端），或他接收到的报文是否就是前面鉴别 Alice 时录制的回放。观察力极强的读者会记起 TCP 的三次握手协议需要处理相同的问题，如果接收的 SYN 报文段来自较早连接的一个 SYN 报文段的旧副本（重新传输）的话，TCP 连接的服务器一侧不会接受该连接。TCP 服务器一侧如何解决“判断客户是否真正还活跃”的问题呢？它选择一个很长时间内都不会再次使用的初始序号，然后把这个序号发给客户，然后等待客户以包含这个序号的 ACK 报文段来响应。此处我们能够为鉴别目的采用同样的思路。

不重数（nonce）是在一个协议的生存期中只使用一次的数。也就是说，一旦某协议使用了一个不重数，就永远不会再使用那个数字了。协议 ap4.0 以如下方式使用一个不重数：

1）Alice 向 Bob 发送报文“我是 Alice”。

2）Bob 选择一个不重数 R，然后把这个值发送给 Alice。

3）Alice 使用她与 Bob 共享的对称秘密密钥 K_{A-B}来加密这个不重数，然后把加密的不重数 $K_{A-B}(R)$ 发回给 Bob。与在协议 ap3.1 中一样，由于 Alice 知道 K_{A-B}并用它加密一个值，就使得 Bob 知道收到的报文是由 Alice 产生的。这个不重数用于确定 Alice 是活跃的。

4）Bob 解密接收到的报文。如果解密得到的不重数等于他发送给 Alice 的那个不重数，则可鉴别 Alice 的身份。

协议 ap4.0 如图 8-18 所示。通过使用这个在生存期中只出现一次的值 R，然后核对返回的值 $K_{A-B}(R)$，Bob 能够确定两点：Alice 是她所声称的那个人（因为她知道加密 R 所需的秘密密钥），Alice 是活跃的（因为她已经加密了 Bob 刚刚产生的不重数 R）。

不重数和对称密钥密码体制的使用形成了 ap4.0 的基础。一个自然的问题是，我们是否能够使用不重数和公开密钥密码体制（而不是对称密钥密码体制）来解决鉴别问题？这个问题将在本章后面的习题中进行探讨。

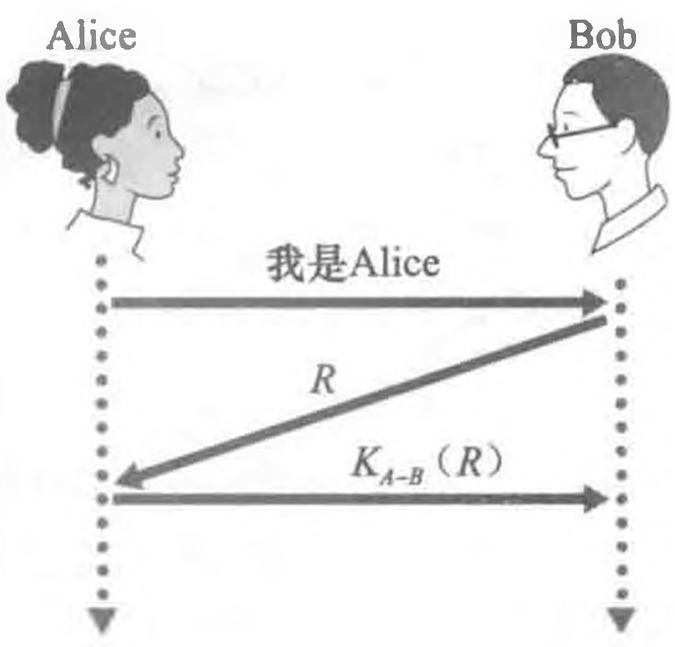

图 8-18　协议 ap4.0

8.5 安全电子邮件

在前面的各节中，我们分析了网络安全中的基本问题，包括对称密钥密码体制和公开密钥密码体制、端点鉴别、密钥分发、报文完整性和数字签名。我们现在着手研究如何使用这些工具在因特网中提供安全性。

有趣的是，为因特网协议栈上面4层的任一层提供安全性服务是可能的。当为某一特定的应用层协议提供安全性时，则使用这一协议的应用程序将能得到一种或多种安全服务，诸如机密性、鉴别或完整性。为某一运输层协议提供安全性时，则所有使用这一协议的应用程序都可以得到该运输层协议所提供安全性服务。在基于主机到主机的网络层提供安全性时，则所有运输层报文段（当然也包括所有应用层数据）都可以得到该网络层所提供的安全服务。当基于一条链路提供安全性时，则经过这个链路传输的所有帧中的数据都得到了该链路提供的安全性服务。

在8.5～8.8节中，我们考察了如何在应用层、运输层、网络层和数据链路层中使用这些安全性工具。为了与本书的整体框架保持一致，我们从协议栈的顶层开始，讨论在应用层的安全性。我们的方法是使用特定的应用程序如电子邮件，作为应用层安全性的一个学习案例。然后我们沿协议栈向下，分析SSL协议（它在运输层提供安全性）、IPsec协议（它在网络层提供安全性），以及IEEE 802.11无线局域网协议的安全性。

你可能会感到奇怪：为什么要在因特网的多个层次上提供安全性功能呢？仅在网络层提供安全性功能并加以实施还不足够吗？对这个问题有两个答案。首先，尽管可以通过加密数据报中的所有数据（即所有的运输层报文段），以及通过鉴别所有数据报的源IP地址，在网络层能够提供“地毯式覆盖”安全性，但是却并不能提供用户级的安全性。例如，一个商业站点不能依赖IP层安全性来鉴别一个在该站点购买商品的顾客。因此，此处除了较低层的地毯式覆盖安全性外，还需要更高层的安全性功能。第二，在协议栈的较高层上部署新的因特网服务（包括安全性服务）通常较为容易。而等待在网络层上广泛地部署安全性，可能还需要未来若干年才能解决，许多应用程序的开发者“着手做起来”，并在他们中意的应用程序中引入安全性功能。一个典型的例子就是PGP（Pretty Good Privacy），它提供了安全电子邮件（将在本节后面讨论）。由于只需要客户和服务器应用程序代码，PGP是第一个在因特网上得到广泛应用的安全性技术。

8.5.1 安全电子邮件

我们现在使用8.2节和8.3节的密码学原则来生成一个安全电子邮件系统。我们以递进的方式来产生这个高层设计，每一步引入一些新安全性服务。当设计安全电子邮件系统时，我们需要记住最初在8.1节中所介绍的那个有趣的例子，即Alice和Bob之间的风流韵事。设想一下Alice发送一个电子邮件报文给Bob，而Trudy试图入侵的情况。

在做出为Alice和Bob设计一个安全电子邮件系统的努力之前，我们应当首先考虑他们最为希望的安全特性是什么。重中之重是机密性。正如8.1节讨论的那样，Alice或Bob都不希望Trudy阅读到Alice所发送的电子邮件报文。Alice和Bob最希望在该电子邮件系统中看到的第二种特性是具备发送方鉴别。特别是，当Bob收到这样的报文“I don't love you anymore. I never want to see you again. Formerly yours, Alice（我不再爱你了。我再也不想看到你了。Alice）”时，Bob自然而然地要确定这个报文确实来自Alice，而非Trudy发

送的。另外，这两个情人欣赏的另一种特性是报文完整性，也就是说，确保 Alice 所发的报文在发送给 Bob 的过程中没有被改变。最后，电子邮件系统应当提供接收方鉴别；即 Alice 希望确定她的确正在向 Bob 发信，而不是向假冒 Bob 的其他人（如 Trudy）发信。

因此我们从处理最为关注的机密性开始。提供机密性的最直接方式是 Alice 使用对称密钥技术（如 DES 或 AES）加密所要传输的报文，而 Bob 则在接收时对报文解密。如 8.2 节讨论的那样，如果对称密钥足够长，且仅有 Alice 和 Bob 拥有该密钥，则其他人（包括 Trudy）要想读懂这条报文极为困难。尽管这种方法直截了当，但因为仅有 Alice 和 Bob 具有该密钥的副本，这使得分发对称密钥非常困难（我们在 8.2 节中讨论过）。因此我们自然就考虑用其他方法——公开密钥密码（例如使用 RSA）。在公开密钥方法中，Bob 使得他的公钥为公众所用（例如，从一台公钥服务器或其个人网页上得到），Alice 用 Bob 的公钥加密她的报文，然后向 Bob 的电子邮件地址发送该加密报文。当 Bob 接收到这个报文时，只需用他的私钥即可解密之。假定 Alice 确定得到的公钥是 Bob 的公钥，这种方法是提供所希望的机密性的极好方法。然而，存在的一个问题是公开密钥加密的效率相对低下，尤其对于长报文更是如此。

为了克服效率问题，我们利用了会话密钥（在 8.2.2 节中讨论过）。具体来说：①Alice选择一个随机对称会话密钥 K_S；②用这个对称密钥加密她的报文 m；③用 Bob 的公钥 K_B^+ 加密这个对称密钥；④级联该加密的报文和加密的对称密钥以形成一个“包”；⑤向 Bob 的电子邮件地址发送这个包。这些过程显示在图 8-19 中（在这张图和下一张图中，带圈的“+”表示级联，带圈的“-”表示级联的分解）。当 Bob 接收到这个包时：①他使用其私钥 K_B^- 得到对称密钥 K_S；②使用这个对称密钥 K_S 解密报文 m。

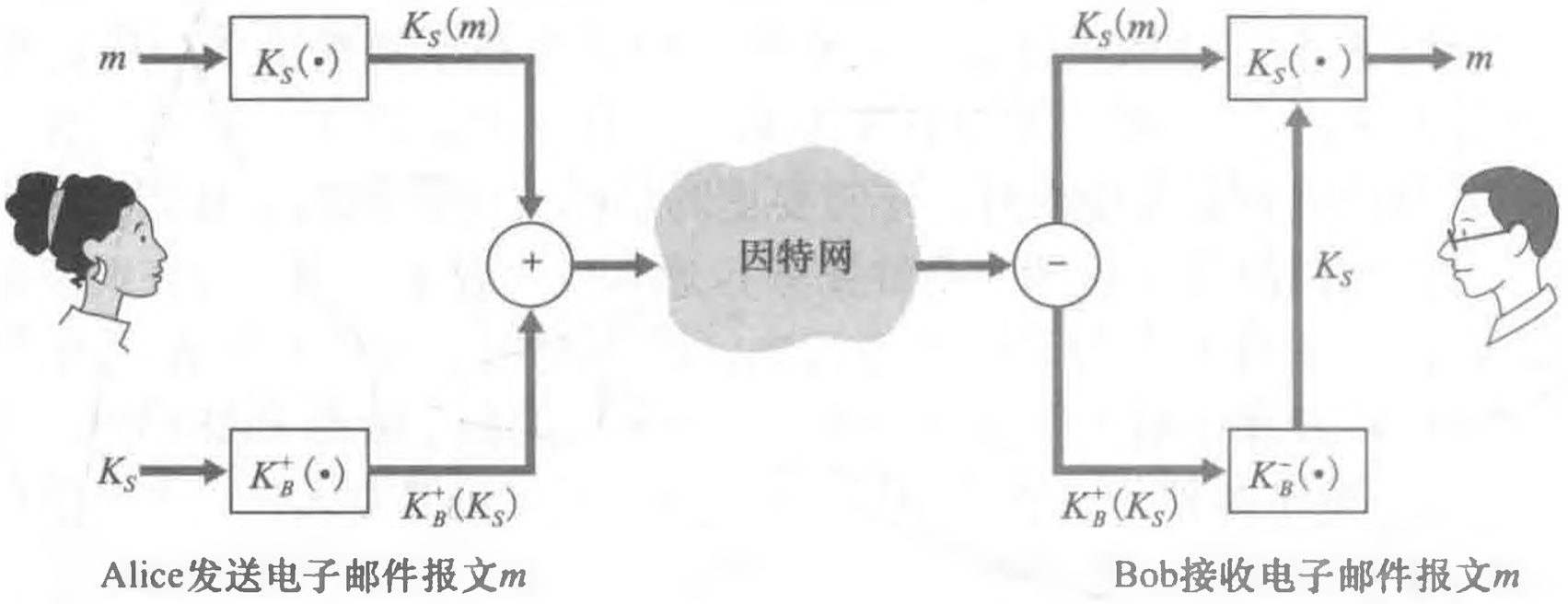

图 8-19　Alice 使用一个对称会话密钥 K_S 向 Bob 发送一个安全电子邮件

设计完提供机密性的安全电子邮件系统后，现在我们设计另一个可以提供发送方鉴别和报文完整性的系统。我们暂且假设 Alice 和 Bob 目前不关心机密性（他们要和其他人分享他们的爱情!），只关心发送方鉴别和报文完整性。为了完成这个任务，我们使用如 8.3 节所描述的数字签名和报文摘要。具体说来：①Alice 对她要发送的报文 m 应用一个散列函数 H（例如 MD5），从而得到一个报文摘要；②用她的私钥 K_A^- 对散列函数的结果进行签名，从而得到一个数字签名；③把初始报文（未加密）和该数字签名级联起来生成一个包；④向 Bob 的电子邮件地址发送这个包。当 Bob 接收到这个包时：①他将 Alice 的公钥 K_A^+ 应用到被签名的报文摘要上；②将该操作的结果与他自己对该报的散列 H 进行比较。在图 8-20 中阐述了这些步骤。如 8.3 节中所讨论，如果这两个结果相同，则 Bob 完全可以确信这个报文来自 Alice 且未被篡改。

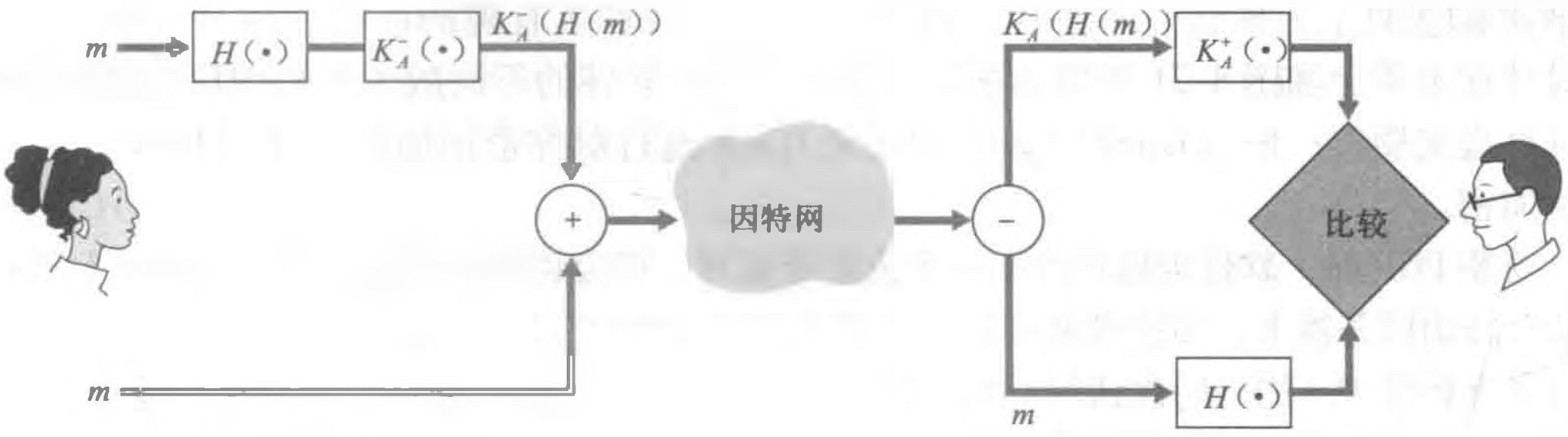

图 8-20　使用散列函数和数字签名来提供发送方鉴别和报文完整性

现在我们考虑设计一个提供机密性、发送方鉴别和报文完整性的电子邮件系统。这可以通过把图 8-19 和图 8-20 中的过程结合起来而实现。Alice 首先生成一个预备包，它与图 8-20 中的包完全相同，其中包含她的初始报文和该报文数字签名过的散列。然后 Alice 把这个预备包看作一个报文，再用图 8-19 中的发送方的步骤发送这个新报文，即生成一个新包发给 Bob。Alice 所做的这些步骤如图 8-21 所示。当 Bob 接收到这个包后，他首先应用图 8-19 中他这一侧的步骤，然后再应用图 8-20 中他这一侧的步骤。应当明确这一设计的目标是提供机密性、发送方鉴别和报文完整性。注意到在这一方案中，Alice 两次使用了公开密钥密码：一次用了她的私钥，另一次用了 Bob 的公钥。类似地，Bob 也两次使用了公开密钥密码：一次用了他的私钥，一次用了 Alice 的公钥。

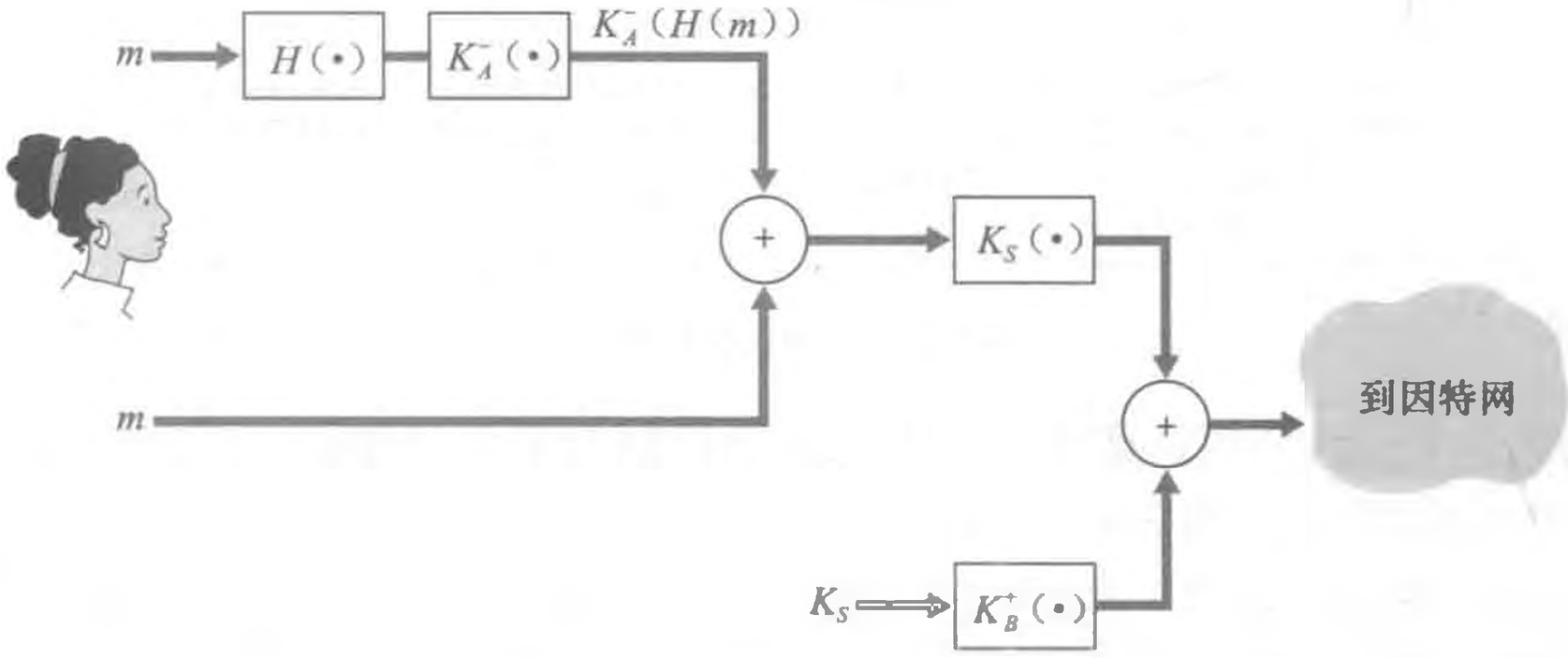

图 8-21　Alice 使用对称密钥密码、公开密钥密码、散列函数和数字签名来提供安全性、发送方鉴别和报文完整性

图 8-21 所示的安全电子邮件系统可能在大多数情况下都能为大多数电子邮件用户提供满意的安全性。但是仍有一个重要的问题没有解决。图 8-21 中的设计要求 Alice 获得 Bob 的公钥，也要求 Bob 获得 Alice 的公钥。但这些公钥的分发并不是一个小问题。例如，Trudy 可能假冒 Bob，发给 Alice 她自己的公钥，并告诉 Alice 这个公钥是 Bob 的公钥，使得 Trudy 就能接收到 Alice 发给 Bob 的报文。如我们在 8.3 节所学，安全地分发公钥的一种常用方法是通过 CA 验证该公钥。

8.5.2　PGP

Philip Zimmermann 于 1991 年所写的 PGP（Pretty Good Privacy）是电子邮件加密方案的一个范例［PGPI 2016］。在公共领域中有各种版本的 PGP 可供使用；例如，你能够在国

际 PGP 的主页上为你喜爱的平台找到 PGP 软件以及许多有趣的读物［PGPI 2016］。PGP 的设计在本质上和图 8-21 中所示的设计相同。PGP 软件的不同版本使用 MD5 或使用 SHA 来计算报文摘要；使用 CAST、三重 DES 或 IDEA 进行对称密钥加密；使用 RSA 进行公开密钥加密。

安装 PGP 时，软件为用户产生一个公开密钥对。该公钥能被张贴到用户的网站上或放置在某台公钥服务器上。私钥则使用用户口令进行保护。用户每次访问私钥时都要输入这个口令。PGP 允许用户选择是否对报文进行数字签名、加密报文，或同时进行数字签名和加密。图 8-22 显示了一个 PGP 签名的报文。这个报文在 MIME 首部之后出现。报文中的加密数据为 $K_A^-(H(m))$，即数字签名的报文摘要。如我们上述讨论，Bob 为了验证报文的完整性，需要得到 Alice 的公钥。

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash:  SHA1
Bob:
Can I see you tonight?
Passionately yours, Alice
-----BEGIN PGP SIGNATURE-----
Version: PGP for Personal Privacy 5.0
Charset:  noconv
yhHJRHhGJGhgg/12EpJ+lo8gE4vB3mqJhFEvZP9t6n7G6m5Gw2
-----END PGP SIGNATURE-----
```

图 8-22　PGP 签名报文

图 8-23 显示了一个秘密 PGP 报文。这个报文也出现在 MIME 首部之后。当然，明文报文不包括在这个秘密电子邮件报文中。当一个发送方（例如 Alice）要确保机密性和完整性时，PGP 在如图 8-23 所示的报文中包含一个类似于图 8-22 中的报文。

```
-----BEGIN PGP MESSAGE-----
Version: PGP for Personal Privacy 5.0
u2R4d+/jKmn8Bc5+hgDsqAewsDfrGdszX68liKm5F6Gc4sDfcXyt
RfdS10juHgbcfDssWe7/K=lKhnMikLo0+1/BvcX4t==Ujk9PbcD4
Thdf2awQfgHbnmKlok8iy6gThlp
-----END PGP MESSAGE
```

图 8-23　一个秘密 PGP 报文

PGP 也提供了一种公钥认证机制，但是这种机制与更为传统的 CA 差异很大。PGP 公钥由一个可信 Web 验证。当 Alice 相信一个密钥/用户名对确实匹配时，她自己就可以验证这一密钥/用户名对。此外，PGP 允许 Alice 为她所信任的用户鉴别更多密钥提供担保。一些 PGP 用户通过保持密钥签署方（key-signing party）互相签署对方的密钥。用户实际走到一起，交换公钥，并用自己的私钥对对方的公钥签名来互相验证密钥。

8.6　使 TCP 连接安全：SSL

在前一节中，我们看到对一个特定的应用（即电子邮件），密码技术是怎样提供机密性、数据完整性和端点鉴别的。在这一节中，我们在协议栈中向下一层，考察密码技术如何用安全性服务加强 TCP，该安全性服务包括机密性、数据完整性和端点鉴别。TCP 的这种强化版本通常被称为**安全套接字层**（Secure Socket Layer，SSL）。SSL 版本 3 的一个稍加修改的版本被称为**运输层安全性**（Transport Layer Security，TLS），已经由 IETF 标准化［RFC 4346］。

SSL 最初由 Netscape 设计，而使 TCP 安全隐含的基本思想先于 Netscape 的工作（例如，参见［Woo 1994］）。由于 SSL 的崭露头角，它已经得到了广泛部署。SSL 得到了所有流行 Web 浏览器和 Web 服务器的支持，并基本上被用于所有因特网商业站点（包括 Ama-

zon、eBay、Yahoo!、MSN 等等）。每年经 SSL 花费了数百亿美元。事实上，如果你使用信用卡通过因特网购买任何东西的话，在你的浏览器和服务器之间的通信几乎一定使用了 SSL。（当你使用浏览器时，若 URL 以 https：开始而不是以 http 开始，就能认定正在使用 SSL。）

为了理解 SSL 的需求，我们浏览一下某典型的因特网商业的场景。Bob 在 Web 上冲浪，到达了 Alice 公司的站点，这个站点正在出售香水。Alice 公司站点显示了一个表格，假定 Bob 可以在该表格中输入香水的类型和所希望的数量、他的地址和他的支付卡号等信息。Bob 输入这些信息，点击“提交”，就期待收到（通过普通邮政邮件）所购买的香水；他也期待着在他的下一次支付卡报表中收到对所购物品的支付信息。所有这一切听起来不错，但是如果不采取安全措施，Bob 也许会有一些意外。

- 如果没有使用机密性（加密），一个入侵者可能截取 Bob 的订单并得到他的支付卡信息。这个入侵者则可以用 Bob 的费用来购买商品。
- 如果没有使用完整性，入侵者可能修改 Bob 的订单，让他购买比希望瓶数多 10 倍的香水。
- 最后，如果没有使用服务器鉴别，这个显示 Alice 公司著名徽标的服务器实际上是由 Trudy 维护的一个站点，Trudy 正在假冒 Alice 公司。当 Trudy 收到 Bob 的订单后，可能拿了 Bob 的钱一走了之。或者 Trudy 可能充当一名身份窃贼，收集 Bob 的名字、地址和信用卡号。

SSL 通过采用机密性、数据完整性、服务器鉴别和客户鉴别来强化 TCP，就可以解决这些问题。

SSL 经常用来为发生在 HTTP 之上的事务提供安全性。然而，因为 SSL 使 TCP 安全了，因此它能被应用于运行在 TCP 之上的任何应用程序。SSL 提供了一个简单的具有套接字的应用编程接口（API），该接口类似于 TCP 的 API。当一个应用程序要使用 SSL 时，它包括了 SSL 类/库。如在图 8-24 中所示，尽管 SSL 技术上位于应用层中，但从研发者的角度看，它是一个提供 TCP 服务的运输协议，而这里的 TCP 服务用安全性服务加强了。

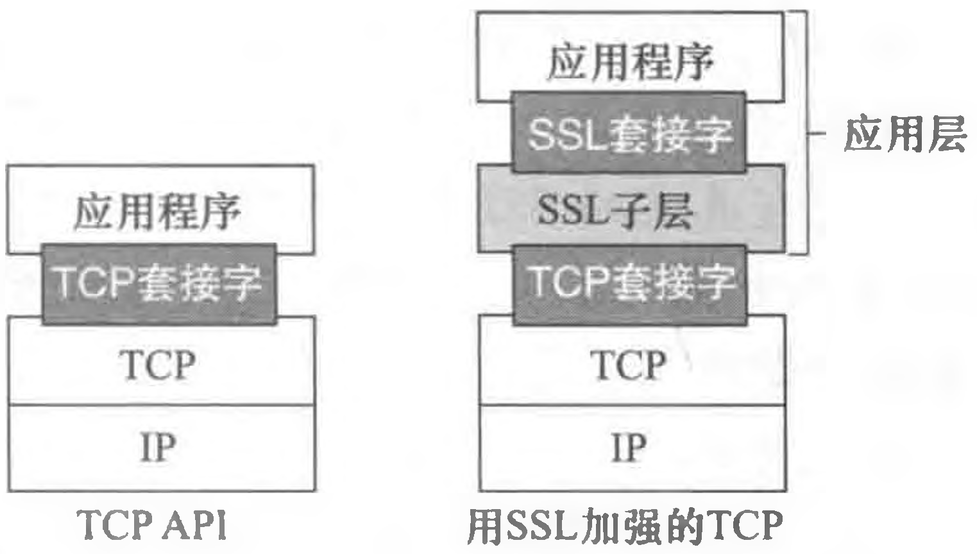

图 8-24　尽管从 SSL 技术上看位于应用层中，但从研发者的角度看它是一个运输层协议

8.6.1　宏观描述

我们从描述一个简化的 SSL 版本开始，这将使我们从宏观上理解 SSL 的工作原理和工作过程。我们将这个 SSL 的简化版本称之为“类 SSL”。描述过类 SSL 之后，在下一小节中我们将描述真实的 SSL，填充细节。类 SSL（和 SSL）具有三个阶段：握手、密钥导出和数据传输。我们现在描述针对一个客户（Bob）和一个服务器（Alice）之间的通信会话的这三个阶段，其中 Alice 具有私钥/公钥对和将她的身份与其公钥绑定的证书。

1. 握手

在握手阶段，Bob 需要：①与 Alice 创建一条 TCP 连接；②验证 Alice 是真实的 Alice；③发送给 Alice 一个主密钥，Bob 和 Alice 持用该主密钥生成 SSL 会话所需的所有对称密

钥。这三个步骤显示在图8-25中。注意到一旦创建了TCP连接，Bob就向Alice发送一个hello报文。Alice则用她的证书进行响应，证书中包含了她的公钥。如在8.3节所讨论，因为该证书已被某CA证实过，Bob明白无误地知道该公钥属于Alice。然后，Bob产生一个主密钥（MS）（该MS将仅用于这个SSL会话），用Alice的公钥加密该MS以生成加密的主密钥（EMS），并将该EMS发送给Alice。Alice用她的私钥解密该EMS从而得到该MS。在这个阶段后，Bob和Alice（而无别的人）均知道了用于这次SSL会话的主密钥。

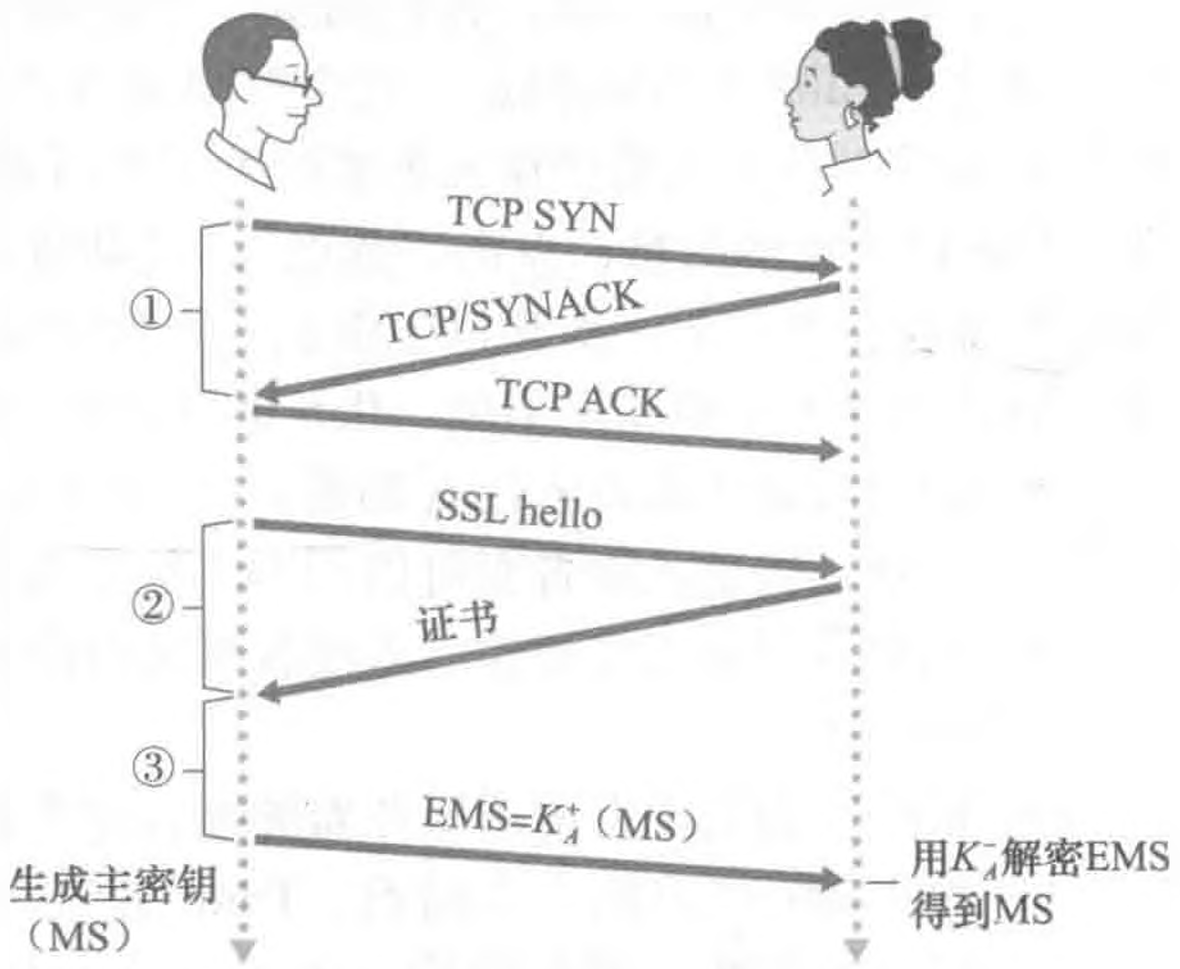

图8-25　类SSL握手，首先建立一个TCP连接

2. 密钥导出

从原则上讲，MS此时已由Bob和Alice共享，它能够用作所有后继加密和数据完整性检查的对称会话密钥。然而，对于Alice和Bob每人而言，使用不同的密码密钥，并且对于加密和完整性检查也使用不同的密钥，通常认为更为安全。因此，Alice和Bob都使用MS生成4个密钥：

- E_B，用于从Bob发送到Alice的数据的会话加密密钥
- M_B，用于从Bob发送到Alice的数据的会话MAC密钥
- E_A，用于从Alice发送到Bob的数据的会话加密密钥
- M_A，用于从Alice发送到Bob的数据的会话MAC密钥

Alice和Bob每人都从MS生成4个密钥。这能够通过直接将该MS分为4个密钥来实现。（但在真实的SSL中更为复杂一些，我们后面将会看到。）在密钥导出阶段结束时，Alice和Bob都有了4个密钥。其中的两个加密密钥将用于加密数据；两个MAC密钥将用于验证数据的完整性。

3. 数据传输

既然Alice和Bob共享相同的4个会话密钥（E_B，M_B，E_A和M_A），他们就能够经TCP连接开始发送安全的数据。因为TCP是一种字节流协议，一种自然的方法是用SSL在传输中加密应用数据，然后将加密的数据在传输中传给TCP。但是如果我们真的这样做，我们将用于完整性检查的MAC置于何处呢？我们无疑不希望等到TCP会话结束时才验证所有Bob数据的完整性，Bob数据的发送要经历整个会话！为了解决这个问题，SSL将数据流分割成记录，对每个记录附加一个MAC用于完整性检查，然后加密该“记录+MAC”。为了产生这个MAC，Bob将数据连同密钥M_B放入一个散列函数中，如在8.3节所讨论。为了加密“记录+MAC”这个包，Bob使用他的会话加密密钥E_B。然后这个加密的包将传递给TCP经因特网传输。

虽然这种方法几经周折，但它为整个报文流提供数据完整性时仍未达到无懈可击。特别是，假定Trudy是一名“中间人”，并且有在Alice和Bob之间发送的TCP报文段流中插入、删除和代替报文段的能力。例如，Trudy能够俘获由Bob发送的两个报文段，颠倒这两个报文段的次序，调整TCP报文段的序号（这些未被加密），然后将这两个次序翻转的

报文段发送给 Alice。假定每个 TCP 报文段正好封装了一个记录，我们现在看看 Alice 将如何处理这些报文段。

1）在 Alice 端运行的 TCP 将认为一切正常，将这两个记录传递给 SSL 子层。

2）在 Alice 端的 SSL 将解密这两个记录。

3）在 Alice 端的 SSL 将使用在每个记录中的 MAC 来验证这两个记录的数据完整性。

4）然后 SSL 将解密的两条记录的字节流传递给应用层；但是 Alice 收到的完整字节流由于记录的颠倒而次序不正确！

鼓励读者观察类似的场景，如当 Trudy 删除报文段或当 Trudy 重放报文段时。

对该问题的解决方案如你可能猜想的那样，那就是使用序号。SSL 采用如下的方式。Bob 维护一个序号计数器，计数器开始为 0，Bob 每发送的一个 SSL 记录它都增加 1。Bob 并不实际在记录中包括一个序号，但当他计算 MAC 时，他把该序号包括在 MAC 的计算中。所以，该 MAC 现在是数据加 MAC 密钥 M_B 加当前序号的散列。Alice 跟踪 Bob 的序号，通过在 MAC 的计算中包括适当的序号，使她验证一条记录的数据完整性。SSL 序号的使用阻止了 Trudy 执行诸如重排序或重放报文段等中间人攻击。（为什么?）

4. SSL 记录

SSL 记录（以及类 SSL 记录）显示在图 8-26 中。该记录由类型字段、版本字段、长度字段、数据字段和 MAC 字段组成。注意到前三个字段是不加密的。类型字段指出了该字段是握手报文还是包含应用数据的报文。它也用于关闭 SSL 连接，如下面所讨论。在接收端的 SSL 使用长度字段以从到达的 TCP 字节流中提取 SSL 记录。版本字段是自解释的。

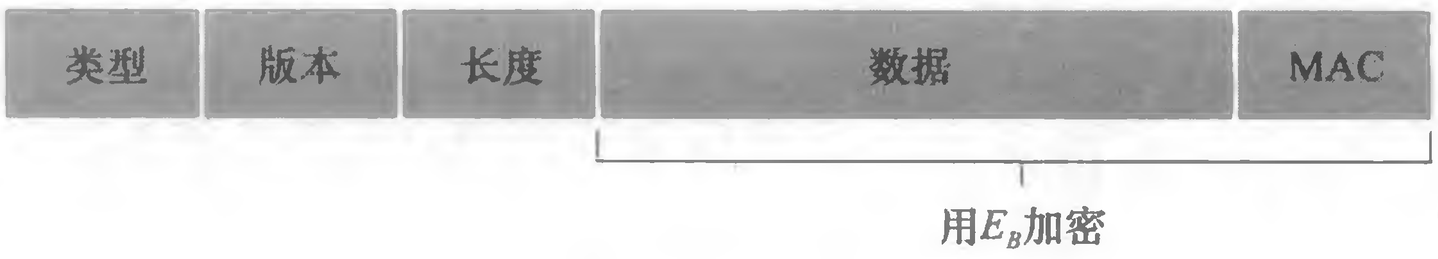

图 8-26　SSL 记录格式

8.6.2　更完整的描述

前一小节涉及了类 SSL 协议；其目的是让我们对 SSL 的工作原理和工作过程有一个基本理解。既然我们已经对 SSL 有了基本了解，就能够更深入地研究实际 SSL 协议的要点了。为了配合阅读对 SSL 协议的描述，鼓励读者完成 Wireshark SSL 实验，它在本书配套的 Web 网站上可供使用。

1. SSL 握手

SSL 并不强制 Alice 和 Bob 使用一种特定的对称密钥算法、一种特定的公钥算法或一种特定的 MAC。相反，SSL 允许 Alice 和 Bob 在握手阶段在 SSL 会话开始时就密码算法取得一致。此外，在握手阶段，Alice 和 Bob 彼此发送不重数，该数被用于会话密钥（E_B，M_B，E_A 和 M_A）的生成中。真正的 SSL 握手的步骤如下：

1）客户发送它支持的密码算法的列表，连同一个客户的不重数。

2）从该列表中，服务器选择一种对称算法（例如 AES）、一种公钥算法（例如具有特定密钥长度的 RSA）和一种 MAC 算法。它把它的选择以及证书和一个服务器不重数返回给客户。

3）客户验证该证书，提取服务器的公钥，生成一个**前主密钥**（Pre- Master Secret，PMS），用服务器的公钥加密该 PMS，并将加密的 PMS 发送给服务器。

4）使用相同的密钥导出函数（就像 SSL 标准定义的那样），客户和服务器独立地从 PMS 和不重数中计算出**主密钥**（Master Secret，MS）。然后该 MS 被切片以生成两个密码和两个 MAC 密钥。此外，当选择的对称密码应用于 CBC（例如 3DES 或 AES），则两个初始化向量（Initialization Vector，IV）也从该 MS 获得，这两个 IV 分别用于该连接的两端。自此以后，客户和服务器之间发送的所有报文均被加密和鉴别（使用 MAC）。

5）客户发送所有握手报文的一个 MAC。

6）服务器发送所有握手报文的一个 MAC。

最后两个步骤使握手免受篡改危害。为了理解这一点，观察在第一步中，客户通常提供一个算法列表，其中有些算法强，有些算法弱。因为这些加密算法和密钥还没有被协商好，所以算法的这张列表以明文形式发送。Trudy 作为中间人，能够从列表中删除较强的算法，迫使客户选择一种较弱的算法。为了防止这种篡改攻击，在步骤 5 中客户发送一个级联它已发送和接收的所有握手报文的 MAC。服务器能够比较这个 MAC 与它已接收和发送的握手报文的 MAC。如果有不一致，服务器能够终止该连接。类似地，服务器发送一个它已经看到的握手报文的 MAC，允许客户检查不一致性。

你可能想知道在步骤 1 和步骤 2 中存在不重数的原因。序号不足以防止报文段重放攻击吗？答案是肯定的，但它们并不只是防止“连接重放攻击”。考虑下列连接重放攻击。假设 Trudy 嗅探了 Alice 和 Bob 之间的所有报文。第二天，Trudy 冒充 Bob 并向 Alice 发送正好是前一天 Bob 向 Alice 发送的相同的报文序列。如果 Alice 没有使用不重数，她将以前一天发送的完全相同的序列报文进行响应。Alice 将不怀疑任何不规矩的事，因为她接收到的每个报文将通过完整性检查。如果 Alice 是一个电子商务服务器，她将认为 Bob 正在进行第二次订购（正好订购相同的东西）。在另一方面，在协议中包括了一个不重数，Alice 将对每个 TCP 会话发送不同的不重数，使得这两天的加密密钥不同。因此，当 Alice 接收到来自 Trudy 重放的 SSL 记录时，该记录将无法通过完整性检查，并且假冒的电子商务事务将不会成功。总而言之，在 SSL 中，不重数用于防御“连接重放”，而序号用于防御在一个进行中的会话中重放个别分组。

2. 连接关闭

在某个时刻，Bob 或者 Alice 将要终止 SSL 会话。一个方法是让 Bob 通过直接终止底层的 TCP 连接来结束该 SSL 会话，这就是说，通过让 Bob 向 Alice 发送一个 TCP FIN 报文段。但是这种幼稚设计为*截断攻击*（truncation attack）创造了条件，Trudy 再一次介入一个进行中的 SSL 会话中，并用 TCP FIN 过早地结束了该会话。如果 Trudy 这样做的话，Alice 将会认为她收到了 Bob 的所有数据，而实际上她仅收到了其中的一部分。对这个问题的解决方法是，在类型字段中指出该记录是否是用于终止该 SSL 会话的。（尽管 SSL 类型是以明文形式发送的，但在接收方使用了记录的 MAC 对它进行了鉴别。）通过包括这样一个字段，如果 Alice 在收到一个关闭 SSL 记录之前突然收到了一个 TCP FIN，她可能知道正在进行着某些耍花招的事情。

到此为止完成了对 SSL 的介绍。我们已经看到它使用了在 8.2 节和 8.3 节讨论的许多密码学原则。希望更深入地探讨 SSL 的读者可以阅读 Rescorla 的有关 SSL 的可读性很强的书籍［Rescorla 2001］。

8.7 网络层安全性：IPsec 和虚拟专用网

IP 安全（IP Security）协议更常被称为 IPsec，它为网络层提供了安全性。IPsec 为任意两个网络层实体（包括主机和路由器）之间的 IP 数据报提供了安全。如我们很快要描述的那样，许多机构（公司、政府部门、非营利组织等等）使用 IPsec 创建了运行在公共因特网之上的**虚拟专用网**（Virtual Private Network，VPN）。

在学习 IPsec 细节之前，我们后退一步来考虑为网络层提供机密性所包含的意义。在网络实体对之间（例如，两台路由器之间，两台主机之间，或者路由器和主机之间）具有网络层机密性，发送实体加密它发送给接收实体的所有数据报的载荷。这种载荷可以是一个 TCP 报文段、一个 UDP 报文段、一个 ICMP 报文等等。如果这样的网络层服务适当的话，从一个实体向其他实体发送的所有数据报将隐形于任何可能嗅探该网络的第三方，发送的数据报包括电子邮件、Web 网页、TCP 握手报文和管理报文（例如 ICMP 和 SNMP）。正因为如此，网络层安全性被认为提供了“地毯覆盖”。

除了机密性，网络层安全协议潜在地能够提供其他安全性服务。例如，它能提供源鉴别，使得接收实体能够验证安全数据报的源。网络层安全协议能够提供数据完整性，使得接收实体能够核对在数据报传输过程中可能出现的任何篡改。网络层安全服务也能提供防止重放攻击功能，这意味着 Bob 能够检测任何攻击者可能插入的任何冗余数据报。我们将很快看到 IPsec 的确提供了用于这些安全服务的机制，即机密性、源鉴别、数据完整性和重放攻击防护。

8.7.1 IPsec 和虚拟专用网

跨越在多个地理区域上的某机构常常希望有自己的 IP 网络，使它的主机和服务器能够以一种安全和机密的方式彼此发送数据。为了达到这个目标，该机构能够实际部署一个单独的物理网络，该网络包括路由器、链路和 DNS 基础设施且与公共因特网完全分离。这样一种为特定的机构专用的分立网络被称为**专用网络**（private network）。毫不奇怪，专用网络可能耗资巨大，因为该机构需要购买、安装和维护它自己的物理网络基础设施。

不同于部署和维护一个专用网络，如今许多机构在现有的公共因特网上创建 VPN。使用 VPN，机构办公室之间的流量经公共因特网而不是经物理上独立的网络发送。而为了提供机密性，办公室之间的流量在进入公共因特网之前进行加密。图 8-27 中显示了 VPN 的一个简单例子。这里的机构由一个总部、一个分支机构和旅行中的销售员组成，销售员通常从他们的旅馆房间接入因特网。（在该图中仅显示了一名销售员。）在这个 VPN 中，无论何时，位于总部的两台主机相互发送 IP 数据报或位于分支机构的两台主机要通信，它们都使用经典的 IPv4（即无 IPsec 服务）。然而，当两台机构的主机经过跨越公共因特网的路径时，这些流量在进入因特网之前进行加密。

为了感受 VPN 的工作过程，我们浏览图 8-27 场景中的一个简单例子。当总部中的一台主机向某旅馆中的某销售员发送一个 IP 数据报时，总部中的网关路由器将经典的 IPv4 转换成为 IPsec 数据报，然后将该 IPsec 数据报转发进因特网。该 IPsec 数据报实际上具有传统的 IPv4 首部，因此在公共因特网中的路由器处理该数据报，仿佛它对路由器而言是一个普通的 IPv4 数据报。但是如图 8-27 所示，IPsec 数据报的载荷包括了一个 IPsec 首部，该首部被用于 IPsec 处理；此外，IPsec 数据报的载荷是被加密的。当该 IPsec 数据报到达

销售员的便携机时，便携机的操作系统解密载荷（并提供其他安全服务，如验证数据完整性），并将解密的载荷传递给上层协议（例如，给 TCP 或 UDP）。

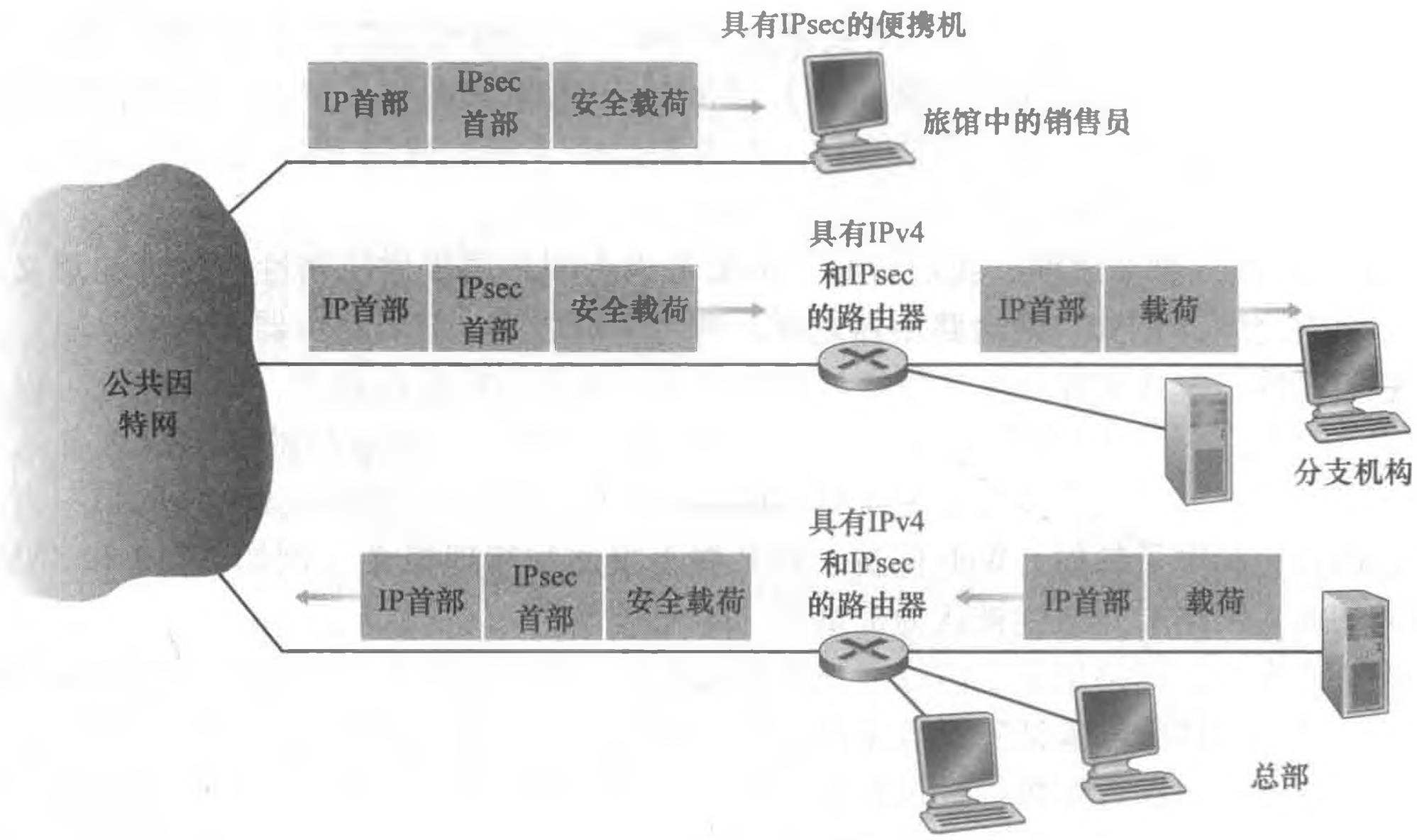

图 8-27　虚拟专用网

我们刚刚给出了某机构能够应用 IPsec 生成一个 VPN 的高层面的展望。为了通过局部看全局，我们已经去掉了许多重要的细节。现在我们来进行更深入的学习。

8.7.2　AH 协议和 ESP 协议

IPsec 是一个相当复杂的整体，即它被定义为 10 多个 RFC 文档。两个重要的文档是 RFC 4301 和 RFC 6071，前者描述了总体 IP 安全体系结构，后者提供了一个 IPsec 协议集的概述。在本教科书中我们的目标与往常一样，并不只是一味重复枯燥和晦涩难解的 RFC 文档，而是采用一种更具可操作性和易于教学的方法来描述协议。

在 IPsec 协议族中，有两个主要协议：**鉴别首部**（Authentication Header，AH）协议和**封装安全性载荷**（Encapsulation Security Payload，ESP）协议。当某源 IPsec 实体（通常是一台主机或路由器）向一个目的实体（通常也是一台主机或路由器）发送安全数据报时，它可以使用 AH 协议或 ESP 协议来做到。AH 协议提供源鉴别和数据完整性服务，但不提供机密性服务。ESP 协议提供了源鉴别、数据完整性和机密性服务。因为机密性通常对 VPN 和其他 IPsec 应用是至关重要的，所以 ESP 协议的使用比 AH 协议要广泛得多。为了讲清 IPsec 并且避免许多难题，我们将此后专门关注 ESP 协议。鼓励还想学习 AH 协议的读者研讨相关的 RFC 和其他在线资源。

8.7.3　安全关联

IPsec 数据报在网络实体对之间发送，例如两台主机之间、两台路由器之间或一台主机和一台路由器之间。在从源实体向目的实体发送 IPsec 数据报之前，源和目的实体创建了一个网络层的逻辑连接。这个逻辑连接称为**安全关联**（Security Association，SA）。一个 SA 是一个单工逻辑连接；也就是说，它是从源到目的地单向的。如果两个实体要互相发

送安全数据报，则需创建两个 SA，每个方向一个。

例如，再次考虑图 8-27 中那个机构的 VPN。该机构由一个总部、一个分支机构和 n 个旅行销售员组成。为了举例的缘故，我们假设在总部和分支机构之间有双向 IPsec 流量，并且总部和销售员之间也有双向 IPsec 流量。在这个 VPN 中，有多少个 SA 呢？为了回答这个问题，注意到在总部网关路由器和分支机构网关路由器之间有两个 SA（一个方向一个）；对每个销售员的便携机而言，在总部网关和便携机之间有两个 SA（仍是一个方向一个）。因此，总计为（$2+2n$）个 SA。然而记住，并非从网关路由器或便携机发送进因特网的所有流量都将是 IPsec 安全的。例如，总部中的一台主机可能要访问公共因特网中的某 Web 服务器（例如 Amazon 或谷歌）。因此，该网关路由器（或该便携机）将发送经典的 IPv4 数据报和安全的 IPsec 数据报进入因特网。

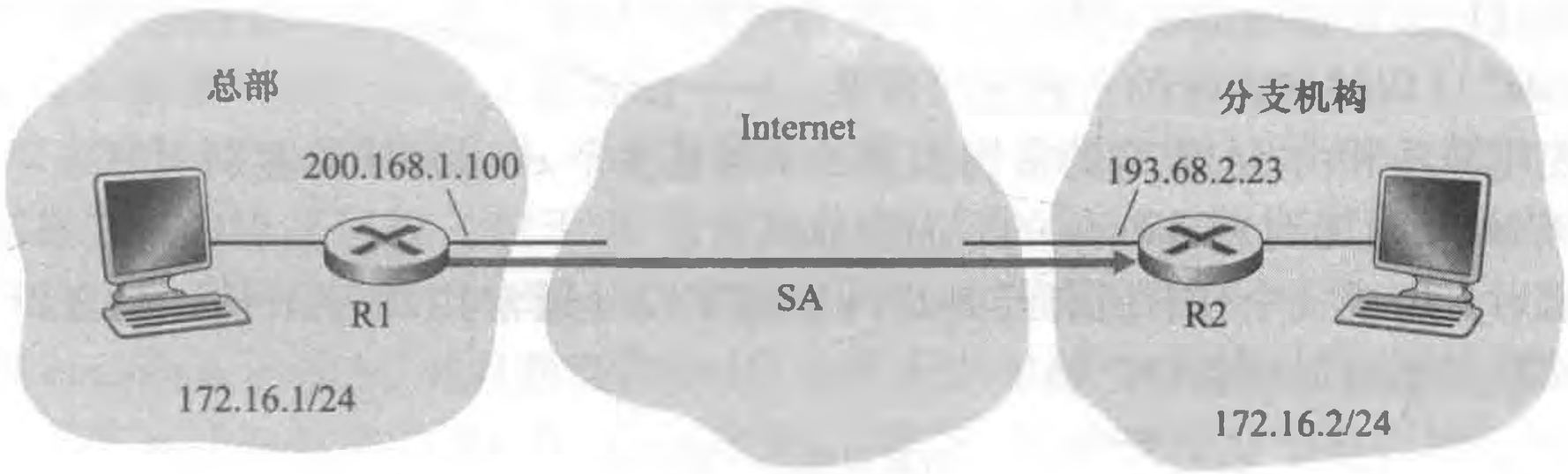

图 8-28　从 R1 到 R2 的安全关联

我们现在观察 SA 的“内部”。为了使讨论明确和具体，我们在图 8-28 中的一个从路由器 R1 到路由器 R2 的 SA 场景下进行观察。（你能够认为路由器 R1 是图 8-27 中的总部网关路由器，而路由器 R2 是图 8-27 中的分支机构网关路由器。）路由器 R1 将维护有关该 SA 的状态信息，这将包括：

- SA 的 32 比特的标识符，称为**安全参数索引**（Security Parameter Index，SPI）。
- SA 的初始接口（在此例中为 200.168.1.100）和 SA 的目的接口（在此例中为 193.68.2.23）。
- 将使用的加密类型（例如，具有 CBC 的 3DES）。
- 加密密钥。
- 完整性检查的类型（例如，具有 MD5 的 HMAC）。
- 鉴别密钥。

无论何时路由器 R1 需要构建一个 IPsec 数据报经过这个 SA 转发，它访问该状态信息以决定它应当如何鉴别和加密该数据报。类似地，路由器 R2 将维护对此 SA 的相同的状态信息，并将使用该信息鉴别和加密任何从该 SA 到达的 IPsec 数据报。

一个 IPsec 实体（路由器或主机）经常维护许多 SA 的状态信息。例如，在图 8-27 中具有 n 个销售员的 VPN 例子中，总部网关路由器维护（$2+2n$）个 SA 的状态信息。一个 IPsec 实体在它的**安全关联数据库**（Security Association Database，SAD）中存储其所有 SA 的状态信息，SAD 是实体操作系统内核中的一个数据结构。

8.7.4　IPsec 数据报

在描述了 SA 后，我们现在能够描述实际的 IPsec 数据报了。IPsec 有两种不同的分组形式，一种用于所谓**隧道模式**（tunnel mode），另一种用于所谓**运输模式**（transport