安全

我们讨论的分布式系统的最后一个原则是安全性。安全决不是最不重要的原则。然而，有人可能会说这是最困难的原则之一，因为安全性需要在整个系统中无处不在。与安全性有关的单个设计缺陷可能会使所有安全措施失效。在本章中，我们将重点介绍分布式系统中通常包含的各种支持安全性的机制。

我们首先介绍安全的基本问题。将各种安全机制构建到系统中实际上没有意义，除非知道如何使用这些机制，以及针对什么使用这些机制。这要求我们了解要执行的安全策略。首先讨论安全策略的概念，以及帮助执行此类策略的机制的一些一般设计问题。我们还简要介绍了必要的密码学。

分布式系统中的安全性大致可以分为两部分。其中一部分涉及用户或进程之间的通信，可能驻留在不同的机器上。确保安全通信的主要机制是安全通道。安全通道，更具体地说，身份验证、消息完整性和机密性，将在单独的部分中讨论。

另一部分涉及授权，它处理如何确保流程只获得对其有权访问的分布式系统中的资源的那些访问权限。授权将在处理访问控制的单独部分中讨论。除了传统的访问控制机制之外，当我们必须处理诸如代理之类的移动代码时，我们还将重点放在访问控制上。

我们还将回到命名，并注意相当棘手的问题，即确保用于检索对象的名称属于该对象，以及如何将安全命名与对人友好的名称结合起来。

安全通道和访问控制需要分发加密密钥的机制，还需要从系统中添加和删除用户的机制。

这些主题由所谓的安全管理涵盖。在另一节中，我们将讨论如何管理加密密钥、安全组管理和分发证明所有者有权访问指定资源的证书。

9.1安全简介

我们通过研究一些一般的安全问题来开始描述分布式系统中的安全性。首先，有必要定义什么是安全系统。我们将安全*策略*与安全*机制*区分开来。我们的第二个关注点是考虑一些安全系统的一般设计问题。最后，简要讨论了在安全协议设计中起关键作用的密码算法。

**安全威胁、策略和机制**

计算机系统中的安全性与可靠性的概念密切相关。非正式地说，可靠的计算机系统是我们有理由相信能够提供其服务的系统[Laprie, 1995]。可靠性包括可用性、可靠性、安全性和可维护性。然而，如果我们要信任计算机系统，那么也应该考虑机密性和完整性。**保密**是指计算机系统的属性，其信息仅向授权方披露。**完整性**是系统资产的变更只能以授权的方式进行的特性。换句话说，在一个安全的计算机系统中，不适当的更改应该是可检测和可恢复的。任何计算机系统的主要资产都是它的硬件、软件和数据。另一种看待计算机系统安全性的方法是，我们试图保护它提供的服务和数据免受**安全威胁**。有四种类型的安全威胁需要考虑[Pfleeger, 2003]：

1. 拦截

2. 中断

3. 修改

4. 制造

截取的概念是指未经授权的一方获得了对服务或数据的访问权。截取的一个典型例子是，双方之间的通信被其他人偷听了。截取同样啊发生在非法复制中，例如，在闯入私人文件系统中的私人目录中进行复制。

中断的一个例子是文件损坏或丢失。更一般地说，中断是指服务或数据变得不可用、被销毁等等的情况。从这个意义上讲，拒绝服务攻击(denial of service attack)是一种安全威胁，可以将其分类为中断。

修改涉及未经授权的数据更改或篡改服务，使其不再遵循其原始规范。修改的例子包括截取并随后更改传输的数据、篡改数据库条目，以及更改程序以便秘密地记录用户的活动。

伪造是指产生了通常不存在的额外数据或活动的情况。例如，入侵者可能试图将条目添加到密码文件或数据库中。同样，有时可以通过重播以前发送的消息来闯入系统。我们将在本章后面遇到这样的例子。

注意，中断、修改和伪造都可以看作是数据伪造的一种形式。

简单地说一个系统应该能够保护自己免受所有可能的安全威胁，并不是真正构建一个安全系统的方法。首先需要的是安全需求的描述，即安全策略。**安全策略**精确地描述了允许系统中的实体执行哪些操作，以及禁止执行哪些操作。实体包括用户、服务、数据、机器等等。一旦制定了安全策略，就可以将精力集中在可以执行策略的**安全机制**上。重要的安全机制包括：

1. 加密

2. 身份验证

3. 授权

4. 审计

加密是计算机安全的基础。加密将数据转换成攻击者无法理解的东西。换句话说，加密提供了实现数据机密性的方法。此外，加密允许我们检查数据是否被修改。因此，它还提供了对完整性检查的支持。

身份验证用于验证用户、客户机、服务器、主机或其他实体的声明身份。对于客户机，基本的前提是，在服务开始代表客户机执行任何工作之前，服务必须了解客户机的标识(除非服务对所有人都可用)。通常，用户通过密码进行身份验证，但是还有许多其他方法可以对客户机进行身份验证。

客户机经过身份验证后，需要检查该客户机是否被授权执行请求的操作。访问医疗数据库中的记录就是一个典型的例子。根据访问数据库的人员，可以授予读取记录、修改记录中的某些字段或添加或删除记录的权限。

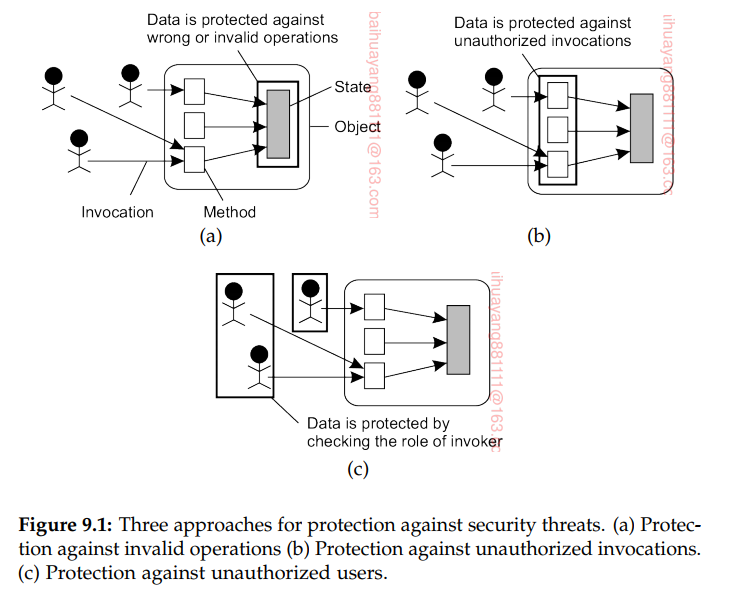
审计工具用于跟踪哪些客户机访问了哪些内容，以及以何种方式访问。虽然审计并不能真正提供任何针对安全威胁的保护，但是审计日志对于分析安全漏洞以及随后对入侵者采取措施非常有用。由于这个原因，攻击者通常不愿意留下任何可能最终导致暴露其身份的痕迹。 从这个意义上说，日志访问有时使攻击成为一项风险更大的业务。

**设计问题**

分布式系统或任何计算机系统都必须提供安全服务，通过这些服务可以实现各种各样的安全策略。在实现通用安全服务时，需要考虑许多重要的设计问题。在接下来的几页中，我们将讨论其中的三个问题:控制焦点、安全机制的分层和简单性(参见Gollmann [2006])。

**控制的重点**

在考虑保护(可能是分布式的)应用程序时，可以遵循三种不同的方法，如图9.1所示。第一种方法是直接集中精力保护与应用程序相关的数据。所谓直接，我们的意思是，不管可能对数据项执行什么操作，主要的关注点都是确保数据完整性。通常，这种类型的保护发生在数据库系统中，其中可以制定各种完整性约束，这些约束在每次修改数据项时自动检查。

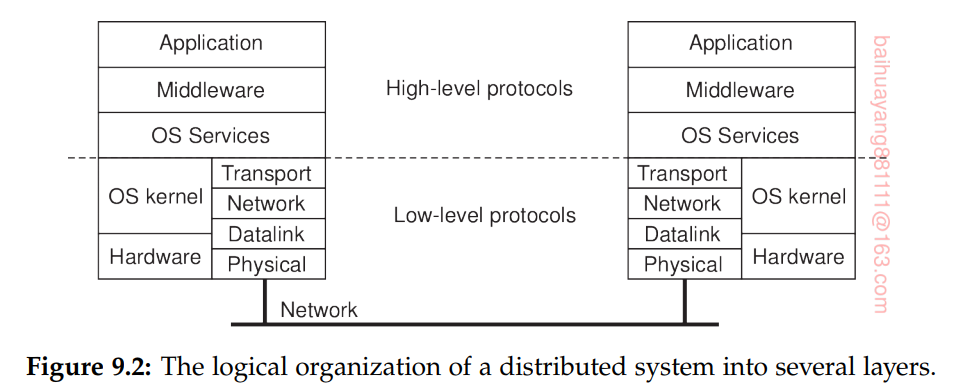


第二种方法是专注于保护，具体指定在访问某些数据或资源时可以调用哪些操作，以及由谁调用这些操作。在这种情况下，控制的重点与访问控制机制密切相关，我们将在本章后面详细讨论。例如，在基于对象的系统中，可以决定为客户机可用的每个方法指定允许哪些客户机调用该方法。或者，访问控制方法可以应用于对象提供的整个接口，或者应用于整个对象本身。因此，这种方法允许访问控制的各种粒度。

第三种方法是直接关注用户，采取措施，使只有特定的人能够访问应用程序，而不管他们希望执行什么操作。例如，银行中的数据库可以通过拒绝访问除银行高层管理人员和特定授权人员之外的任何人来保护。另一个例子是，在许多大学，某些数据和应用程序被限制只供教职员工使用，而不允许学生访问。实际上，控制的重点是定义用户所具有的**角色**，一旦验证了用户的角色，就会授予或拒绝访问资源。因此，作为设计安全系统的一部分，有必要定义人们可能具有的角色，并提供支持基于角色的访问控制的机制。本章稍后我们将回到角色。

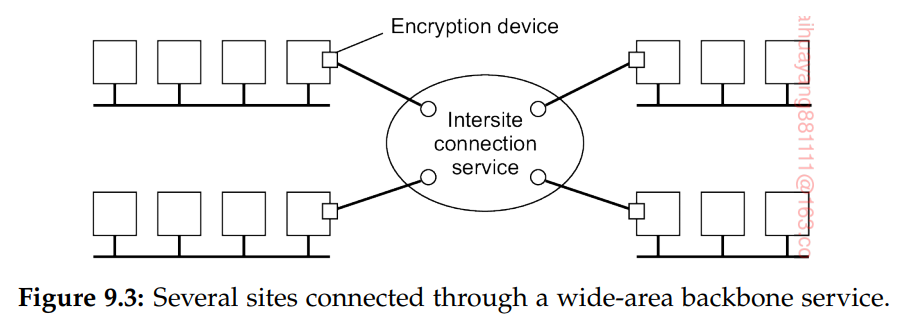
**安全机制的分层**

设计安全系统的一个重要问题是确定应该将安全机制放在哪个级别。在此上下文中，级别与系统逻辑组织成若干层有关。例如，正如我们在第4章中讨论的，计算机网络经常按照一些参考模型组织成层。在第1章中，我们介绍了由应用程序、中间件、操作系统服务和操作系统内核的独立层组成的分布式系统的组织。将这两个组织组合起来大致可以得到如图9.2所示的结果



实际上，图9.2将通用服务与通信服务分开。这种分离对于理解分布式系统中的安全分层非常重要，尤其是信任的概念。信任和安全之间的区别很重要。系统是安全的还是不安全的(考虑到各种概率度量)，但是客户端是否认为系统是安全的是一个信任的问题。安全是技术，信任是情感[毕晓普，2003]。在哪个层中放置安全机制取决于客户端对特定层中服务的安全性的信任程度。

例如，考虑一个位于不同站点的组织，该组织通过一个低层主干连接可能位于地理上分散的站点上的各种局域网，如图9.3所示。可以使用多协议标签交换(MPLS)或虚拟专用网(VPN)技术来配置这种连接。



可以通过在每个骨干交换机上放置加密设备来提供安全性，如图9.3所示。这些设备自动加密和解密站点之间发送的数据包，但不提供同一站点上主机之间的安全通信。如果站点A的Alice向站点B的Bob发送一条消息，并且她担心自己的消息被拦截，那么她至少必须相信站点间通信的加密能够正常工作。例如，这意味着她必须相信两个站点的系统管理员已经采取了适当的措施来防止篡改设备。

现在假设Alice不相信站点间通信的安全性。然后，她可能决定采取自己的措施，例如使用传输层安全性(Transport Layer Security, TLS)服务，该服务可用于跨TCP连接安全地发送消息。这里需要注意的重要一点是，TLS允许Alice设置到Bob的安全连接。所有传输级别的消息都将加密——在我们的示例中，链接级别也是加密的，但这与Alice无关。在这种情况下，Alice必须信任TLS。换句话说，她相信TLS是安全的。

在分布式系统中，安全机制通常位于中间件层。如果Alice不信任TLS，她可能希望使用本地安全RPC服务。同样，她将不得不信任这个RPC服务来做它承诺的事情，例如不泄漏信息或正确验证客户机和服务器。

只有在分布式系统中间件层中放置的安全服务所依赖的服务确实是安全的情况下，才可以信任它们。例如，如果安全RPC服务部分是通过TLS实现的，那么对RPC服务的信任取决于对TLS的信任程度。如果TLS不受信任，那么就不能信任RPC服务的安全性。

**安全机制的分布**

服务之间关于信任的依赖关系导致了可信**计算基础(TCB)**的概念。TCB是(分布式)计算机系统中执行安全策略所需的所有安全机制的集合，因此需要得到信任。TCB越小越好。如果分布式系统作为中间件构建在现有网络操作系统上，那么它的安全性可能取决于底层本地操作系统的安全性。换句话说，分布式系统中的TCB可能包括不同主机上的本地操作系统。

考虑分布式文件系统中的文件服务器。这样服务器可能需要依赖其本地操作系统提供的各种保护机制。这些机制不仅包括保护文件不被文件服务器以外的进程访问的机制，还包括保护文件服务器不被恶意关闭的机制。

因此，基于中间件的分布式系统需要信任它们所依赖的现有本地操作系统。如果不存在这种信任，那么可能需要将本地操作系统的部分功能合并到分布式系统中。考虑一个微内核操作系统，其中大多数操作系统服务作为普通用户进程运行。在这种情况下，例如，文件系统可以完全替换为适合分布式系统的特定需求的文件系统，包括它的各种安全措施。

与此方法相一致的是，根据所需的安全级别将服务分布在不同的机器上，从而将安全服务与其他类型的服务分离开来。例如，对于安全的分布式文件系统，通过将服务器放在具有可信操作系统的机器上(可能运行专用的安全文件系统)，可以将文件服务器与客户机隔离开来。客户机及其应用程序被放置在不受信任的机器上。

这种分离有效地将TCB减少到相对较少的机器和软件组件。通过随后保护这些机器免受来自外部的安全攻击，可以提高对分布式系统安全性的总体信任。

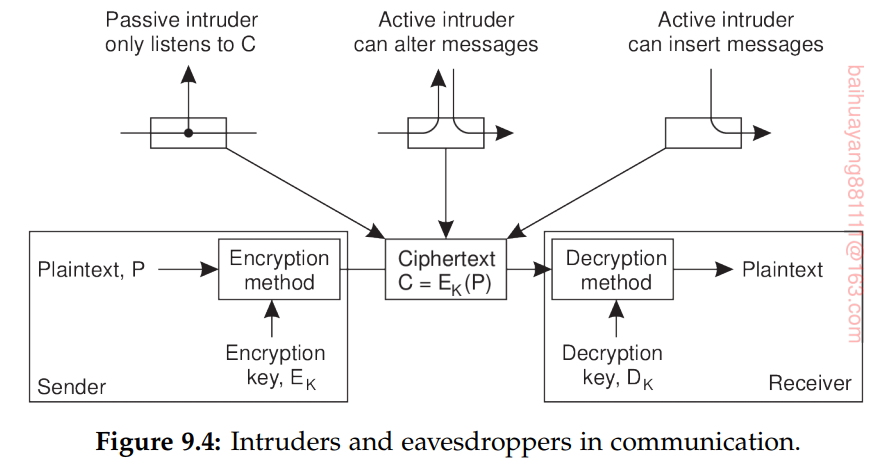
**简单**

与决定将安全机制放置在哪个层相关的另一个重要设计问题是简单性。设计一个安全的计算机系统通常被认为是一项困难的任务。因此，如果系统设计人员能够使用一些易于理解和信任的简单机制来工作，效果就会更好。不幸的是，简单的机制并不总是足以实现安全策略。再次考虑上面讨论过的Alice想要发送消息给Bob的情况。链路级加密是一种简单易懂的机制，可以防止截获站点间的消息流量。然而，如果Alice想确保只有Bob才能接收到她的消息，还需要做更多的工作。在这种情况下，需要用户级身份验证服务，Alice可能需要知道这些服务是如何工作的，以便信任这些服务。因此，用户级身份验证可能至少需要密码密钥的概念和证书等机制的意识，尽管许多安全服务是高度自动化的，并且对用户隐藏。

在其他情况下，应用程序本身就很复杂，引入安全性只会使情况变得更糟。数字支付系统是一个涉及复杂安全协议的应用领域。数字支付协议的复杂性通常是由多方需要通信才能进行支付这一事实造成的。在这些情况下，重要的是用于实现协议的底层机制是相对简单的并且易于理解。简单性将有助于最终用户对应用程序的信任，更重要的是，它将有助于使设计人员相信系统没有安全漏洞。

**密码学**

分布式系统安全的基础是使用密码技术。应用这些技术的基本思想很简单。假设发送方想要将消息m发送给接收方R。为了保护消息不受安全威胁，发送方首先将其加密为难以理解的消息m0，然后将m0发送给R。反过来，R必须将接收到的消息解密为其原始形式m。加密和解密是通过使用密钥参数化的加密方法来完成的，如图9.4所示。发送的消息的原始形式称为明文，如图9.4中P所示。加密的形式称为密文，如图C所示。



要描述用于为分布式系统构建安全服务的各种安全协议，有一个将明文、密文和密钥关联起来的符号是很有用的。按照常见的标记约定，我们将使用C = EK(P)表示密文C是通过使用密钥K加密明文P获得的。同样，P = DK(C)用于表示密钥K对密文C的解密，从而得到明文P。回到图9.4所示的示例中，在以密文C传输消息时，我们需要防范三种不同的攻击，而加密对此有所帮助。首先，**入侵者**可能在发送方或接收方都不知道正在发生窃听的情况下拦截消息。当然，如果传输的消息以一种没有正确密钥就无法轻松解密的方式进行加密，则截取是无用的:入侵者将只看到无法理解的数据。

需要处理的第二种攻击类型是修改消息。修改明文很容易;修改已正确加密的密文要困难得多，因为入侵者必须先解密消息，然后才能对其进行有意义的修改。此外，他还必须对其进行正确的加密，否则接收者可能会注意到消息已被篡改。

第三种攻击是入侵者将加密的消息插入通信系统，试图使R相信这些消息来自S。同样，加密可以帮助防范此类攻击。注意，如果入侵者可以修改消息，他也可以插入消息。

根据加密密钥和解密密钥是否相同，不同加密系统之间有一个基本的区别。在对称密码系统中，相同的密钥用于加密和解密消息：



**对称密码系统**也称为秘钥或共享密钥系统，因为发送方和接收方需要共享相同的密钥，为了确保保护有效，这个共享密钥必须保密;任何人都不许看这把钥匙。我们将使用符号KA,B来表示a和B共享的键。

在**非对称密码系统**中，用于加密和解密的密钥是不同的，但它们组合在一起形成一对惟一的密钥。换句话说，有一个用于加密的单独密钥KE和一个用于解密的密钥KD，诸如此类。



非对称密码系统中的一个密钥是私有的;另一个是公开的。因此，非对称密码系统也称为**公钥系统**。在接下来的代码中，我们使用符号KA+表示属于a的公钥，而KA -表示它对应的私钥。

哪一个加密或解密密钥实际上是公开的，这取决于如何使用这些密钥。例如，如果Alice想向Bob发送一条机密消息，她应该使用Bob的公钥加密该消息。因为Bob是惟一持有关联和私有解密密钥的人，所以他也是惟一能够解密消息的人。

另一方面，假设Bob想确定他刚刚收到的消息实际上来自Alice。在这种情况下，Alice可以保持她的私有加密密钥，以便加密她发送的消息。如果Bob能够使用Alice的公钥成功解密消息（消息中的明文包含足够的信息，使其对Bob有意义），他知道消息一定来自Alice，因为解密密钥与加密密钥是唯一绑定的。

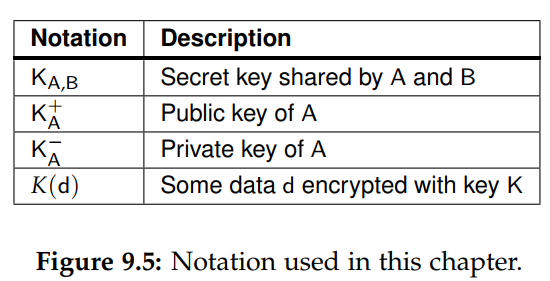
密码学在分布式系统中的最后一个应用是**哈希函数**的使用。哈希函数H接受任意长度的消息m作为输入，并生成一个长度固定的位串H作为输出：



哈希h在某种程度上类似于通信系统中为检测错误而附加到消息后面的额外位，比如循环冗余检查(CRC)。在密码系统中使用的哈希函数具有许多基本属性。首先，它们是**单向函数**，这意味着在计算上不可能找到对应于已知输出h的输入m。另一方面，从m计算h很简单。其次，它们具有较弱的抗碰撞性，这意味着给定一个输入m及其相关输出h = h (m)，在计算上不可能找到另一个不同的输入m' = m，比如h (m) = h (m')。最后，加密哈希函数也具有很强的抗碰撞性，这意味着，当只给定H时，在计算上不可能找到任意两个不同的输入值m和m'，比如H(m) = H(m')。

类似的属性必须应用于任何加密函数E和所使用的密钥。此外，对于任何加密函数EK，当给定明文P和相关密文C = EK(P)时，在计算上都不可能找到密钥K。同样，与碰撞电阻类似，同样，与碰撞抗力类似，当给定一个明文P和一个密钥K时，实际上不可能找到另一个密钥K'，使得EK(P) = EK'(P)。

为密码系统设计算法的艺术和科学有着悠久而迷人的历史[Kahn, 1967]，而构建安全系统通常是惊人地困难，甚至是不可能的[Schneier, 2000]。详细讨论这些算法超出了本书的范围。关于密码算法的信息可以在[Ferguson et al.， 2010]， [Menezes et al.， 1996]和[Schneier, 1996]中找到。图9.5总结了我们在本书的数学表达式中使用的符号和缩写。



9.2安全通道

客户机-服务器模型是组织分布式系统的一种方便方法。在这个模型中，服务器可能是分布式的和复制的，但是也可以作为相对于其他服务器的客户机。在考虑分布式系统中的安全性时，从客户机和服务器的角度考虑同样很有用。特别是，使分布式系统安全本质上归结为两个主要问题。第一个问题是如何确保客户机和服务器之间的通信安全。安全通信需要通信各方的身份验证。在许多情况下，还需要确保消息的完整性和保密性。作为这个问题的一部分，我们还需要考虑保护一组服务器中的通信。

第二个问题是授权问题:一旦服务器接受了来自客户机的请求，它如何确定客户机是否被授权执行该请求？授权与控制对资源的访问有关。

保护客户端和服务器之间的通信的问题，可以从在通信方之间建立一个**安全通道**的角度来考虑[Voydock和Kent, 1983]。安全通道保护发送方和接收方不受截取、修改和制造消息的影响。它也不一定能防止中断。保护消息不被截取是通过确保机密性来实现的:安全通道确保其消息不会被入侵者窃听。

**身份验证**

在详细讨论各种身份验证协议之前，值得注意的是，身份验证和消息完整性不能相互独立。 例如，考虑一个分布式系统，它支持两个通信方的身份验证，但不提供确保消息完整性的机制。在这样的系统中，Bob可以确定Alice是消息m的发送者。但是，如果不能保证Bob在传输期间没有修改m，那么知道Alice发送了(原始版本)m对他有什么用呢。

同样，假设只支持消息完整性，但不支持身份验证机制。当Bob收到一条消息说他刚刚中了100万美元的彩票时，如果他不能证实这条消息是由彩票的组织者发出的，他会高兴吗？

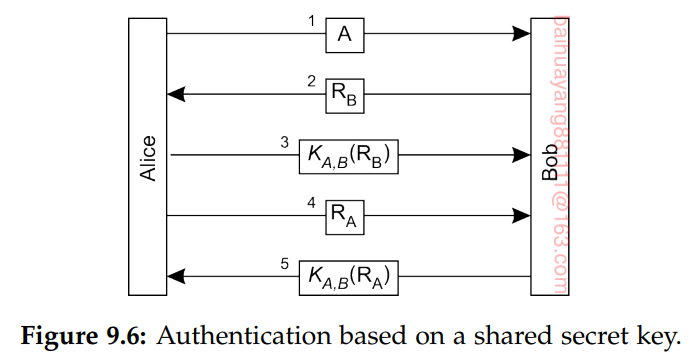
因此，身份验证和消息完整性应该同时进行。在许多协议中，这种组合的工作原理大致如下。同样，假设Alice和Bob想要通信，并且Alice在设置通道时采取主动。Alice首先向Bob发送一条消息，或者向将帮助设置通道的可信第三方发送一条消息。一旦通道设置好，Alice确定她在和Bob说话，Bob也确定他在和Alice说话，他们可以交换消息。

为了确保在进行身份验证之后交换的数据消息的完整性，通常使用会话密钥来使用密钥加密。会话密钥是一种共享(秘密)密钥，用于加密消息，以确保消息的完整性和保密性。这种键通常只在通道存在时使用。当通道关闭时，其关联的会话密钥将被销毁。

**基于共享密钥的身份验证**

让我们从查看基于Alice和Bob之间已经共享的密钥的身份验证协议开始。目前，这两者如何以安全的方式获得共享密钥并不重要。在协议描述中，Alice和Bob分别缩写为A和B，共享密钥记为KA,B。协议采用一种共同的方法，即一方向另一方发出挑战，要求对方做出只有在另一方知道共享密钥的情况下才能正确的响应。这种解决方案也称为**质询-响应协议**。

在基于共享密钥的身份验证的情况下，协议继续执行，如图9.6所示。首先，Alice将她的身份发送给Bob(消息1)，表示她想在两者之间建立一个通信通道。Bob随后向Alice发送一个**质询RB**，如消息2所示。这样的**质询**可以以随机数的形式出现。Alice需要使用她与Bob共享的密钥KA B加密挑战，并将加密的挑战返回给Bob。这个响应如图9.6中的消息3所示，其中包含KA,B(RB)。



当Bob收到对他的质询RB的响应KA,B(RB)时，他可以再次使用共享密钥解密消息，看看它是否包含RB。如果是这样，那么他就知道Alice在另一边，因为还有谁可以在一开始就用KA B加密RB。换句话说，Bob现在已经证实他确实在和Alice说话。但是，请注意，Alice还没有验证它确实是通道另一端的Bob。因此，她发送一个challenge RA (message 4)， Bob通过returnin KA,B(RA)对其进行响应，如message 5所示。当Alice用KA,B解密它并看到她的RA时，她知道她在和Bob说话。（这么做的前提是，A和B已经共享了 KAB）。

多年来，密码协议的开发人员逐渐学会了许多原则。一个重要的教训是，设计能够完成它们应该完成的任务的安全协议通常比看起来要困难得多。此外，调整现有协议以提高其性能，很容易影响其正确性。关于协议设计原则的更多信息可以在[Abadi和Needham, 1996]中找到。