随着远程计算的重新出现，组织需要维护服务器。在过去，这意味着购买物理服务器，按需对其进行配置，将其与网络连接，并让它在某个地方的机柜中运行。组织可以物理访问计算机并完全控制其配置。但是，维护一台服务器（或一组服务器）可能是一项复杂且成本高昂的工作，这涉及购买和维护硬件、及时更新软件、处理安全问题和容量规划问题、管理网络配置等。一般来说，这项工作所需的技能和专业知识与组织目标并不一致。即便是以技术为主的公司也不一定想要做维护服务器的业务。这就是云计算的用武之地。

云计算通过互联网（云）提供远程计算功能。底层硬件由云服务公司（云提供商）维护，使需要这些功能的组织或用户（云用户）免于维护服务器。云计算允许按需购买计算服务。对于云用户而言，这就意味着释放对某些事物的控制权并相信第三方可提供可靠服务。云计算有多种形式，让我们看看其中的一些。

13.3.2 云计算的分类

云计算的各种类别通常由在云提供商和云用户之间划分责任的界限来定义。图13-4展示了四类云计算（IaaS、PaaS、FaaS和SaaS）及各自的责任划分。稍后，我们将逐一介绍它们。

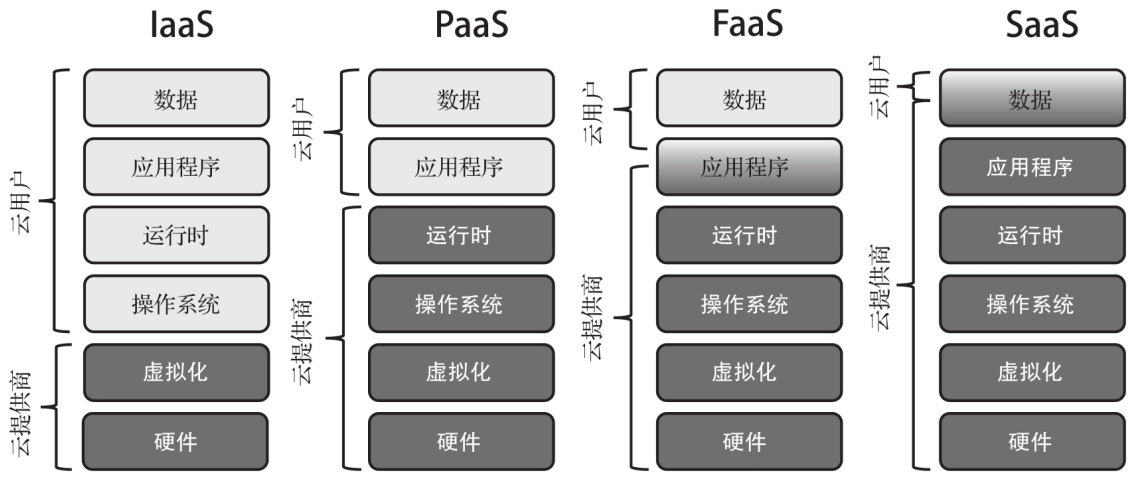

图13-4 各类云计算产品的责任划分

图13-4中的垂直栈表示应用程序运行所需的组件。不论使用哪种云计算产品，都需要全部的组件——不同类型之间的差异在于：每个组件是由云用户负责管理，还是由云提供商负责管理。每个栈中的各种组件应该看着很熟悉，因为我们之前已经介绍过这些内容了。但是，“运行时”还需要解释一下。运行时环境是应用程序执行的环境，它包括了所有需要的库、解释器、进程虚拟机等。现在，让我们从左到右介绍一下图13-4中的四类云计算。

基础设施即服务（Infrastructure as a Service，IaaS）是一种云计算场景，在这个场景中，云提供商只管理硬件和虚拟化，允许用户管理操作系统、运行时环境、应用程序代码和数据。IaaS的用户通常会得到连接到互联网的虚拟计算机，以按照他们认为合适的方式使用，一般是作为某种类型的服务器使用。这个虚拟计算机常常实现为基于管理程序的虚拟机或者Linux发行版的用户模式部分的容器。IaaS虚拟服务器的用户可以访问虚拟机的操作系统，还可以按照自己的想法对它进行设置。这给了用户最大的灵活性，但这也意味着维护系统软件（包括操作系统、第三方软件等）的责任完全落在了用户的肩上。IaaS提供了一台虚拟计算机，用户负责在该计算机上运行的所有内容。Amazon Elastic Compute Cloud（EC2）、Microsoft Azure Virtual Machines和Google Compute Engine都是IaaS的例子。

平台即服务（Platform as a Service，PaaS）赋予云提供商更多的责任。在PaaS场景中，云提供商不仅管理硬件和虚拟化，还要管理用户希望使用的操作系统和运行时环境。PaaS用户开发一个应用程序，这个应用程序的目的是在他们选择的云平台上运行，它会利用这个平台的各种功能。PaaS产品的云用户不需要关心如何维护底层OS或运行时环境，可以只关注应用程序代码。尽管云提供商确实抽象了底层系统的详细信息，但是用户还是需要管理提供商预配了哪些资源，以处理应用程序。预配置资源包括所需的存储容量和被分配的虚拟机类型。PaaS为运行中的代码提供了一个托管平台，用户负责在该平台上运行的应用程序。
Amazon Web Services Elastic Beanstalk、Microsoft Azure App Service和Google App Engine都是PaaS的例子。

函数即服务（Function as a Service，FaaS）采用了PaaS模型，但是比它更进一步。FaaS不需要用户部署完整的应用程序或提前给出平台实例，相反，用户只需要部署响应某些事件的运行代码（函数）。例如，开发人员可以编写一个函数，返回到最近杂货店的距离。这个函数可以响应网络浏览器，把它当前的GPS坐标发送到URL。这种事件驱动模型意味着云提供商负责按需调用用户代码。用户不再需要一直运行应用程序代码，等待请求。这可以简化用户的工作并降低成本，尽管在函数尚未运行时，这可能意味着在请求到来时响应时间会更长。

FaaS是一种无服务器计算，即一种用户无须处理管理服务器或虚拟机问题的云计算模型。当然，这个术语用法不当，服务器实际上是要运行代码的，只是用户不需要考虑这些问题罢了！FaaS为运行代码提供了一个事件驱动平台，用户负责为了响应事件而运行的代码。
Amazon Web Services Lambda、Microsoft Azure Functions和
Google Cloud Functions是FaaS的例子。

软件即服务（Software as a Service，SaaS）是一种完全不同的云服务。SaaS向用户交付完全在云中管理的应用程序。IaaS、PaaS和FaaS适合于希望在云中运行自己代码的软件工程团队，而SaaS则向最终用户或组织交付已编写好的、完整的云应用程序。现在有那么多软件都在云上运行，这看起来似乎并不显眼，但是它却与用户或组织在本地设备和网络上安装并维护软件形成了鲜明对比。SaaS提供了在云中管理的完整应用程序，用户只需负责存储在该应用程序中的数据。甚至连数据管理也有部分是提供商处理的，包括数据如何存储、备份等细节。Microsoft 365、
Google G Suite和Dropbox是SaaS的例子。

云提供商领域的一些主要参与者包括：Amazon Web Services、Microsoft Azure、Google Cloud Platform、IBM Cloud、Oracle Cloud和Alibaba Cloud。

13.4 深网和暗网

你可能已经看过关于发生在暗网（dark Web）或深网（deep Web）中的邪恶事件的新闻。不幸的是，这两个术语经常被搞混，但它们的含义是不同的。如图13-5所示，网络可以分成三大类：表层网、深网和暗网。

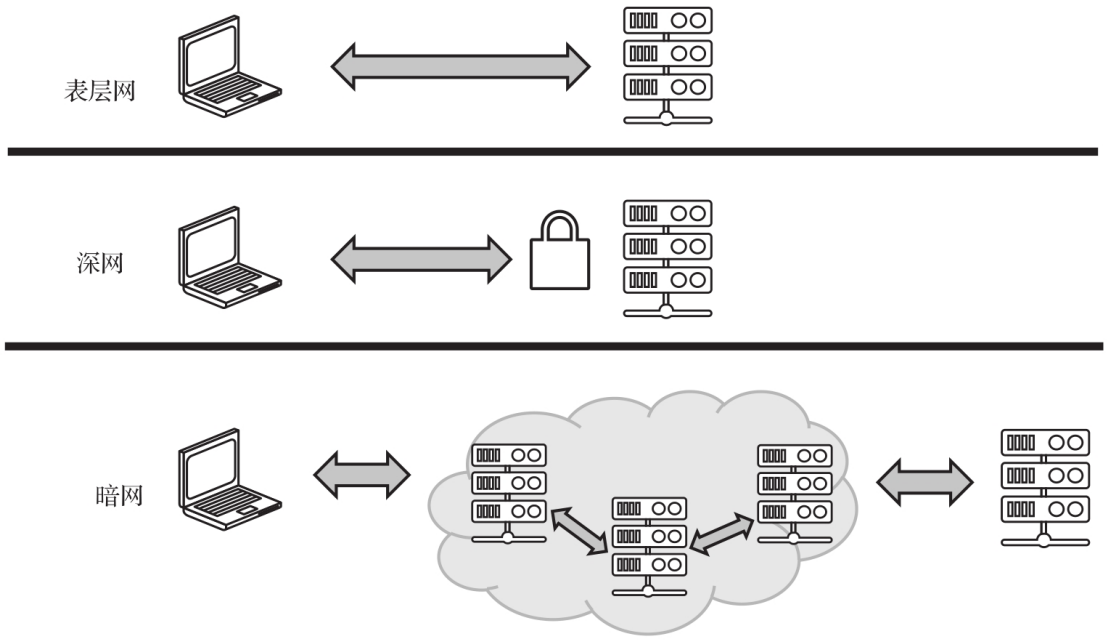

图13-5 表层网、深网和暗网

任何人都可以免费访问的内容是表层网的一部分。公共博客、新闻网站、公共推特帖子都是表层网内容的例子。表层网由搜索引擎索引，有时表层网也被定义为可以用搜索引擎找到的内容。

深网是指不登录网站或网络服务就无法访问的网络内容。大多数互联网用户会定期访问深网内容。查看银行账户余额、通过像Gmail这样的网站阅读电子邮件、登录Facebook、查看Amazon上的个人购买记录——这些都是深网活动的例子。深网只是一些不公开的内容，通常需要某种密码才能访问。大多数用户不想公开自己的电子邮件或银行账户，所以有充分的理由说明为什么这种内容不公开且不能被搜索引擎索引。

暗网是需要用专用软件来访问的网络内容。只使用标准的网络浏览器是无法访问暗网的。最流行的暗网技术是To r（洋葱路由器）。通过加密和中继系统，To r允许匿名访问网络，防止用户的ISP监控访问了哪些网站，

也防止网站知道访问者的IP地址。此外，Tor还允许用户访问那些被称为洋葱服务的网站，如果没有To r就根本无法访问它们——这些网站就是暗网的一部分。To r隐藏了洋葱服务的I P地址，使它们变成匿名的。如你所料，暗网的匿名性有时会被用于犯罪目的。但是，暗网提供的隐私也有其合法用途，比如举报和政治讨论。建议在访问暗网内容时要谨慎。

13.5 比特币

加密货币是一种用于金融交易的数字资产，是传统货币的替代品。加密货币的用户持有该货币的余额，就像在传统的银行一样，也可以把这些货币用于购物和服务。有些用户主要把加密货币看作一种投资，而不是一种商业手段，对这些用户来说，它更类似于黄金。与传统货币不同的是，加密货币通常是分散的（decentralized），没有单一组织控制它们。

13.5.1 比特币基础

比特币于2009年推出，是第一种分布式加密货币，也是现在最著名的加密货币。从那时起，大量的替代加密货币（称为山寨币）如雨后春笋般涌现，但没有一个能挑战比特币的主导地位。比特币的主要货币单位也简称为比特币，缩写为BTC。

比特币和类似的加密货币都以区块链技术为基础。在区块链中，信息被分组成称为区块的数据结构，区块按照时间顺序链接在一起。也就是说，当创建一个新区块时，它被添加到区块链的末端。以比特币为例，区块保存着交易记录，跟踪比特币的移动轨迹。图13-6展示了比特币的区块链。

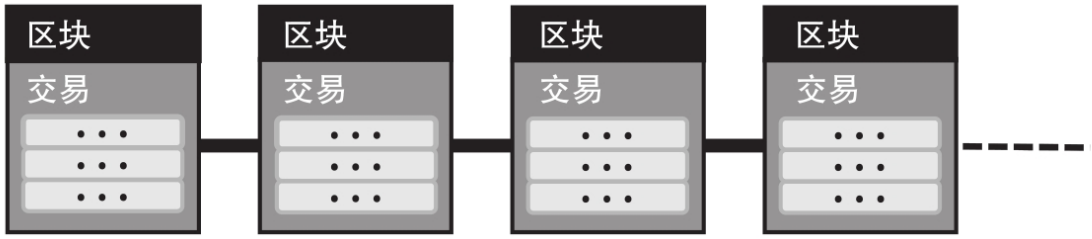

图13-6 比特币的区块链按时间顺序链接交易记录区块

区块链通过互联网等网络运行，由多台计算机一起来处理交易和更新区块链。一起工作以处理比特币交易的计算机被称为比特币网络。连接到比特币网络的计算机被称为节点，某些节点保存了区块链的副本，没有单一的底本。通过加密和解密来保证交易的完整性，并防止篡改区块链中的数据。一旦写入，区块链中的数据就是不可变的，即无法更改。比特币的区块链是一个公开的、分布式的、不可变的交易账本。这个账本用于记录比特币网络上发生的所有事件，比如比特币的转账。

13.5.2 比特币钱包

最终用户的比特币存储在所谓的比特币钱包中。不过，更准确地说是，比特币钱包持有加密密钥对的集合，如图13-7所示。

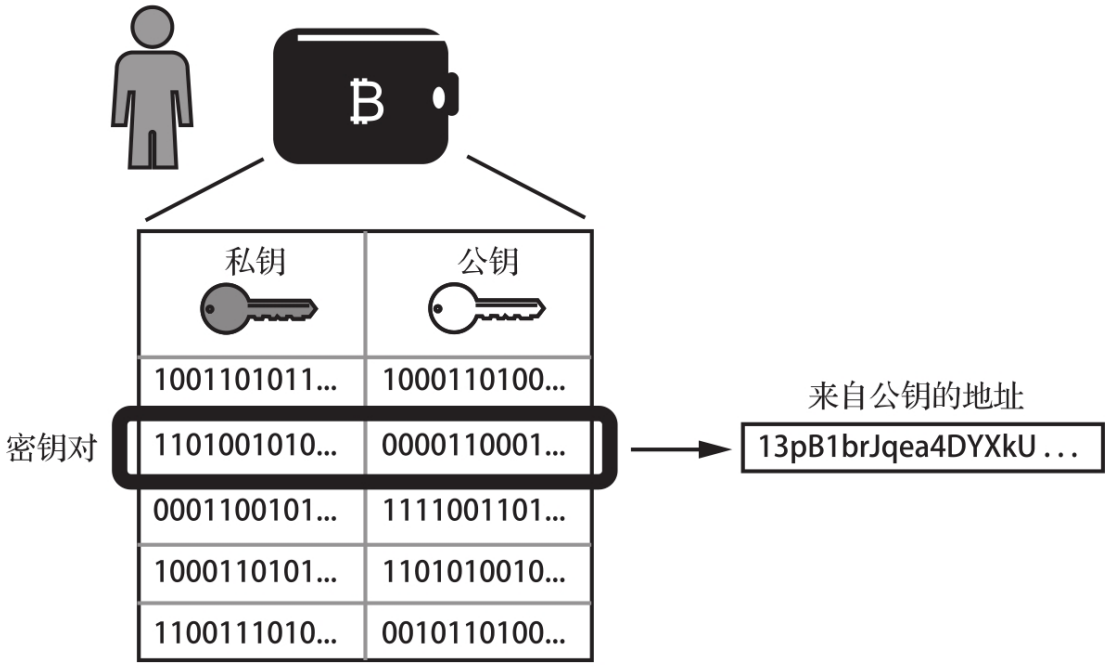

图13-7 比特币钱包包含密钥对，比特币地址派生于公钥

如图13-7所示，钱包中的每一对密钥都含有两个数字：一个私钥和一个公钥。私钥是一个随机生成的256位数字。这个数字必须保密，任何知道

私钥的人都可以使用与该密钥对关联的比特币。公钥用于接收比特币，它从私钥派生而来。在接收比特币时，公钥表示为比特币地址，一个由公钥生成的文本字符串，例如13pB1brJqea4DYXkUKv5n44HCgB kJHa2v1。

假设我有一个比特币要发给你。这个比特币和我控制的一个地址关联。也就是说，我有这个地址的私钥。如果你把由你控制的比特币地址的文本字符串给我，我就可以把我的比特币发送到这个地址。你不需要（也不应该）把你的私钥发给我。我可以把我的比特币发给你是因为我有自己地址的私钥，这允许我使用自己的比特币。反过来，我不能从你的地址转出任何比特币，因为我没有你的私钥。

13.5.3 比特币交易

让我们仔细看看这是如何工作的。比特币的转账被称为交易。为了发送比特币，钱包软件会构造一个交易，指定转账的细节，用私钥对其进行数字签名并向比特币网络广播该交易。比特币网络中的计算机会验证交易并把它添加到区块链的一个新区块中。图13-8展示了一个比特币交易例子。

如图13-8所示，交易包含输入和输出，表示比特币从哪里来到哪里去。图的左侧是前一个交易，只显示了输出，因为与前一个交易的输入无关。在前一个交易中，有0.5比特币被送到了地址A。

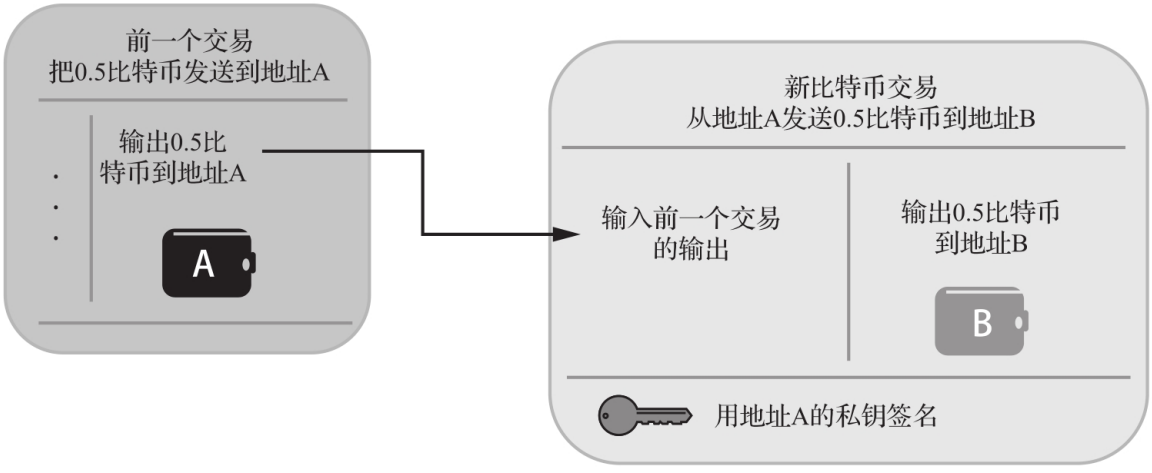

图13-8 比特币交易：把0.5比特币移动到地址B（忽略交易费用）

图13-8的右侧是新的交易，它把0.5比特币从地址A移动到地址B。为了简单起见，这个交易只有一个输入和一个输出。输入表示要转账的比特币的来源。你可能认为这是个比特币地址，但它不是的。相反，这个输入是前一个交易的输出。我们假设地址A是我的地址，我想把0.5比特币转到你的地址，即地址B。现在，我知道之前的0.5比特币已经转到了我的地址，所以我可以把前一个交易的输出当作新交易的输入，从而把0.5比特币转给你。交易的输出部分包含了发送比特币的地址。

尽管你可以认为地址具有比特币余额，但与地址关联的比特币数量并不存储在比特币钱包中，并且余额也不会直接存储在区块链中。相反，与这个地址相关的交易历史被存储在区块链中，而且根据这个历史可以计算出某个地址的余额。提醒一下，比特币钱包只包含支持比特币交易的密钥。

13.5.4 比特币挖掘

维护比特币区块链的过程称为比特币挖掘（Bitcoin mining）。来自全球的计算机把交易区块添加到区块链中，这些计算机被称为矿工（miner）。图13-9展示了比特币挖掘过程。

为了把交易区块添加到区块链，矿工必须验证区块中包含的交易（确保每个交易在语法上是正确的，输入的货币尚未被消费等），还必须完成一个困难问题的计算。要求矿工解决这个问题可以防止篡改区块链，因为改变一个区块就需求解决被改变区块以及该区块链上其后所有区块的问题。作为阻止不需要行为的手段，这种解决困难问题的系统被称为“工作证明”（proof of work）。

要找到计算问题的解决方案就需要大量的试错计算。解决方案难以产生，但易于验证。第一个解决问题的矿工将获得一笔比特币。这就是新比特币产生并被引入系统的方式。在这种情况下，比特币挖掘就类似于传统的挖矿——矿工进行工作并在可能的情况下“挖到金子”。除了获得新“铸造”的比特币之外，矿工还可以对该区块中包含的每笔交易收取一定费用，这

个费用要从交易的比特币总量中扣除。比特币被设计成总共只允许挖掘2100万比特币。一旦达到这个数字，比特币矿工将不再获得比特币，只能依靠交易费用来获得其运营资本。

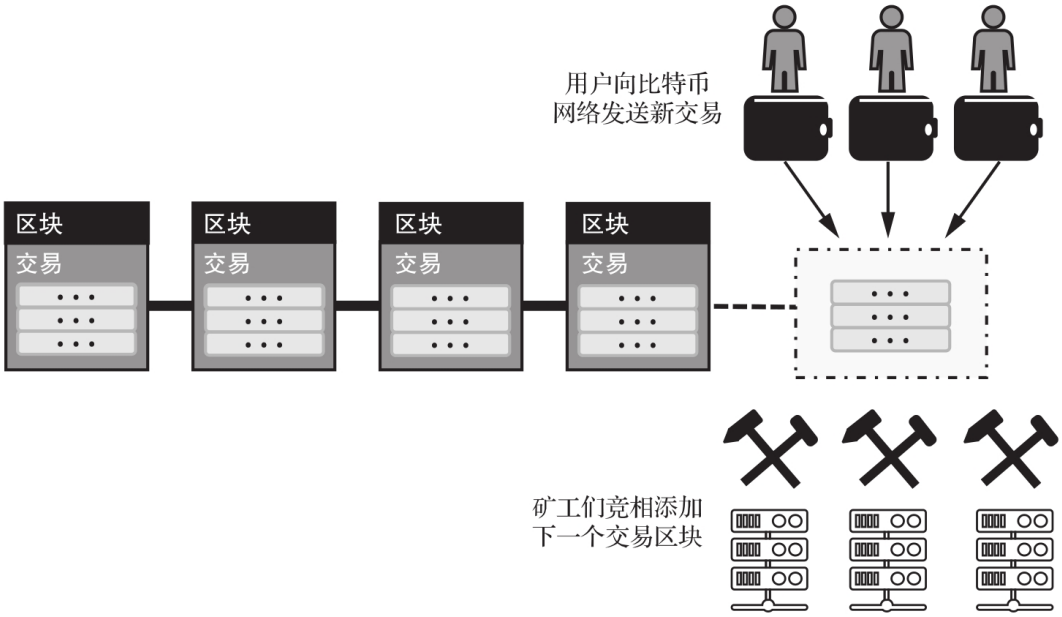

图13-9 比特币挖掘过程

比特币的开端

比特币区块链开始于2009年"挖掘"的第一个区块，这个区块被称为"创世区块"。这个区块由Satoshi Nakamoto挖掘，他被视为比特币的发明人。"Satoshi Nakamoto"被认为是一个化名，在撰写本书时，此人的身份仍存在争议。

要通过比特币挖掘获利，运行挖掘硬件的成本不能超过获得的比特币的价值。比特币挖掘硬件常常很耗电，所以比特币挖掘者的电费可能很高。最初，比特币是在普通计算机上挖掘的，但现在使用专门的昂贵硬件进行尽可能快的挖掘（请记住，奖励授予第一个解决问题的计算机）。这些成本，加上比特币高度波动的价格，意味着比特币挖掘不是一个有保障的盈利途径！

比特币区块链是公开的——任何人都可以查看全部交易。但是，区块链不包含比特币转账的个人身份记录。所以，虽然余额和交易历史记录是公开的，但并没有便捷的方法能把地址和个人联系起来。出于这个原因，比特币吸引了那些想要保持匿名的人，比如那些在暗网上经营商业网站的人。

区块链技术与加密货币密切相关，它被用作金融账本，还可以用于其他目的。任何需要防篡改历史记录的系统都可以使用区块链。时间会证明比特币或其他加密货币是否能取得长期成功，但无论如何，我们可能都会看到区块链以其他新颖的方式发挥作用。

13.6 虚拟现实和增强现实

虚拟现实（Virtual Reality，VR）和增强现实（Augmented Reality，AR）这两项技术有可能从根本上改变我们与计算机交互的方式。虚拟现实是一种计算形式，它让用户沉浸在三维虚拟空间中，这个空间一般通过头戴式设备来呈现。VR允许用户通过各种输入方式与虚拟对象交互，输入方式包括用户的注视、语音命令和专用手持控制器。相比之下，增强现实是把虚拟元素叠加到现实世界中，要么通过头戴式设备实现，要么由用户"通过"手持便携设备（比如智能手机或平板电脑）实现。VR让用户沉浸在另一个世界，AR改变现实世界。

尽管几十年以来已经对VR进行了各种尝试，但直到21世纪10年代，VR才变成主流。Google在2014年通过Google Cardboard帮助推广了VR，Google Cardboard的命名出自VR头戴式设备可以由纸版、镜片和智能手机构成的想法。通过把左眼内容呈现在智能手机一半屏幕上，右眼内容呈现在屏幕的另一半上，专门设计的Cardboard应用程序把VR内容呈现给用户，如图13-10所示。

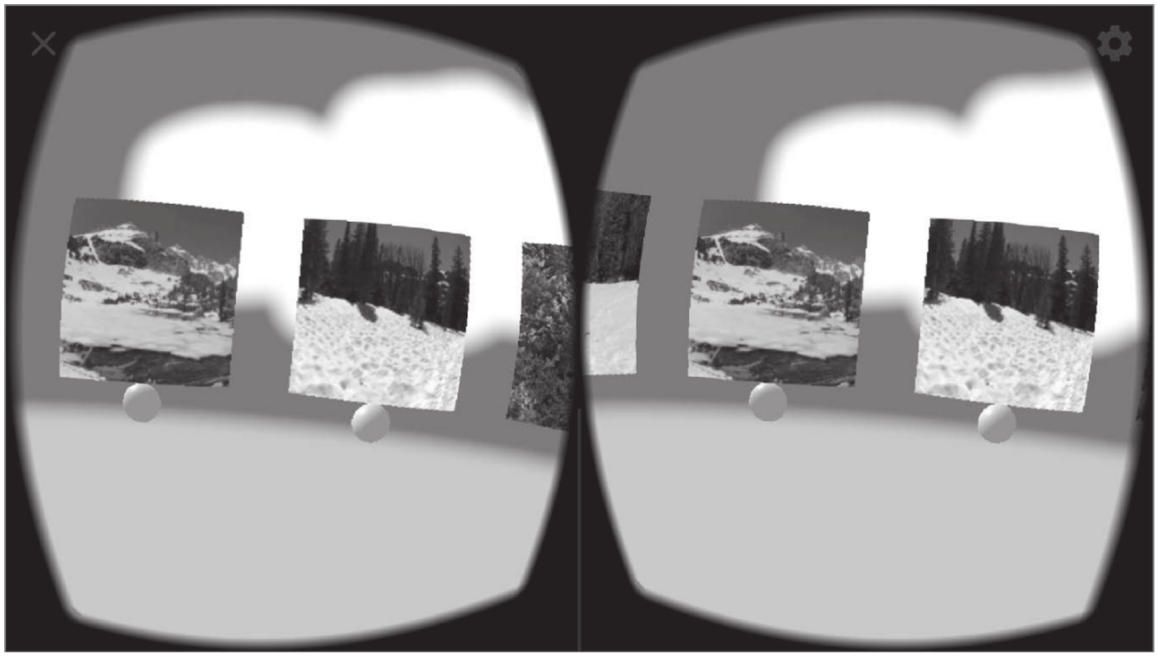

图13-10 专门为Google Cardboard设计的应用程序的VR模式

为Cardboard设计的应用程序依赖于智能手机检测陀螺仪运动的能力，允许显示内容随着用户头部的移动而更新。这样的头戴式设备据说有3个自由度（3DoF），头戴式设备可以追踪有限的头部运动，但无法追踪空间中的其他运动。这使得用户可以使用头戴式设备环顾，但不能移动。Cardboard还支持基本的一键输入。Cardboard简单，但有效。它向许多用户介绍了VR，否则，他们可能不会尝试VR。

更具有沉浸感的体验需要6个自由度（6DoF）：用户可以通过在现实空间中物理移动身体以实现在VR中的四处移动。有些VR头戴式设备支持6DoF，VR控制器可以支持3DoF或6DoF。握在用户手中的6DoF控制器可以追踪控制器在VR空间中的位置，从而允许与VR环境进行更自然的互动。

自Google Cardboard推出以来，消费市场已经发布了许多VR解决方案。有些依赖于智能手机（Samsung Gear VR和Google Daydream）。另外一些通过连接VR头戴式设备和控制器（Oculus Rift、HTC Vive和Windows Mixed Reality），使用个人计算机进行处理。还有一些是独立设备，不需要智能手机或PC（Oculus Go、Oculus Quest和

Lenovo Mirage Solo）。一般来说，连接PC的解决方案提供了最高的图形保真度，但也是最贵的，特别是考虑到所需计算机的成本时。

如前所述，增强现实（AR）是一种相似但独特的技术。VR试图让用户完全沉浸在虚拟世界中，而AR则把虚拟元素叠加到现实世界中。这可以通过移动设备来完成：使用后置摄像头观察现实世界，将模拟元素叠加到摄像头所看到的内容上。先进的AR技术允许软件理解房间内的物理元素，这样叠加的虚拟元素就可以与环境进行无缝交互。移动应用程序中的AR是以基本形式实现的，但在专用设备中它的实现更为充分，比如在Google Glass、Magic Leap的头戴式设备和Microsoft HoloLens。将这种AR设备戴在头上可以把计算机生成的图形叠加到用户视野中。用户能以各种方式（比如语音命令或手部跟踪）与虚拟元素进行交互。

各种VR和AR技术（统称为XR）为软件开发人员提供了多种目标平台。许多VR开发人员依赖于现有的游戏引擎，这些游戏引擎一般用于制作3D游戏，比如Unity游戏引擎或Unreal游戏引擎。游戏开发人员对这些游戏引擎已经非常熟悉了，使用它们能让开发人员相对轻松地面向多个VR平台制作自己的软件。网络开发人员可以使用被称为WebVR和WebXR的JavaScript API开发VR和AR内容。WebVR先出现，它专注于VR。WebXR紧随其后出现，它同时支持AR和VR。

13.7 物联网

传统上，我们认为服务器在互联网上提供服务，用户通过联网的个人计算机设备（比如PC、笔记本计算机和智能手机）与这些服务器进行交互。近些年，我们已经看到与互联网连接的新型设备越来越多——如扬声器、电视机、恒温器、门铃、汽车、灯泡，应有尽有！这种把各种设备连接到互联网的概念称为物联网（Internet of Things，IoT）。

电子元件的成本和物理尺寸正在下降，Wi-Fi和蜂窝互联网接入正在普及，消费者希望他们的设备更加“智能”。所有这些都促成了把一切都连接到互联网的趋势。如果没有某种网络服务的支持，IoT设备一般不会运行，所以云计算的兴起也促进了物联网的传播。对消费者而言，IoT设备在“智能

家居”中很突出，而“智能家居”中所有类型的家用电器都能被监视和控制。在商业领域，在制造业、医疗保健、交通运输等领域中都能找到IoT设备。

虽然这些类型的连接设备带来了明显的好处，但这些设备也带来了风险。它们的安全性是一个特别值得关注的领域。不是所有的IoT设备都能很好地抵御恶意攻击。即使攻击者对设备上的数据没兴趣，攻击者也可能将这个设备充当攻击其他防御良好的网络的一个据点，或者当成对不同目标进行远程攻击的发射点。特别是对消费者来说，IoT设备似乎已经足够无害了，并且在把这样的设备连接到家庭网络时，安全问题通常不是首要考虑的问题。

隐私是IoT设备带来的另一个风险。许多这样的设备本质上就是收集数据的。这些数据常常被发送给云服务进行处理。最终用户应该在多大程度上信任使用他们的个人数据运行这些服务的组织？即使是善意的组织也有可能成为数据泄露的受害者，用户数据可能以意想不到的方式被暴露。像智能音箱这样的设备必须一直处于听的状态，以等待口头命令。这就产生了意外记录私人对话的风险。看到今天的消费者心甘情愿用隐私换取便利，George Orwell的小说《1984》的现代读者可能会发现某种讽刺意味。

IoT设备的所有功能常常依赖于云服务，这也是它们带来的一个风险。如果一个设备与互联网的连接中断，这个设备可能会暂时变得不那么有用。更令人担忧的是，设备制造商可能会在某天永久关闭支持该设备的服务。那时，智能设备就会还原成非智能设备！

注意

请参阅设计41使用学到的硬件、软件和网络知识来构建一个联网的“自动贩卖机”IoT设备。

13.8 总结

本章讨论了与现代计算机相关的各种主题。你学习了app，包括本机app和网络app。你了解了虚拟化和仿真是如何让计算机在虚拟硬件上运行

软件的。你还了解了云计算是如何为运行软件提供新平台的。你学习了表层网、深网和暗网的区别，以及比特币等加密货币如何实现分布式支付系统。我们谈到了虚拟现实和增强现实，以及它们如何为计算机提供独特的用户界面。你还了解了IoT，并有机会搭建一个联网的"自动贩卖机"。

在这里，让我们总结性地回顾一些主要的计算机概念，看看它们是如何结合在一起的。计算机是二进制数字设备，其中的所有信息都表示为0和1。二进制逻辑也被称为布尔逻辑，为计算机运算打下了基础。计算机用数字电路实现，其电平表示为二进制状态——低电平为0，高电平为1。数字逻辑门是基于晶体管的电路，它能执行布尔运算，比如AND和OR。我们可以把这样的逻辑门排布成更加复杂的电路，比如计数器、存储设备和加法电路。这些类型的电路为计算机硬件提供了概念基础，这些硬件包括：执行指令的中央处理器（CPU），上电时存储指令和数据的随机存取存储器（RAM）以及与外界交互的输入/输出（I/O）设备。

计算机是可编程的，它们可以在不改变硬件的情况下执行新任务。告诉计算机要做什么的指令被称为软件或代码。CPU执行机器码，而软件开发人员则通常用高级编程语言编写源代码。计算机程序一般在操作系统上运行，操作系统即与计算机硬件通信并为程序执行提供环境的软件。计算机用互联网通信，互联网是一组全球连接的计算机网络，这些网络都使用TCP/IP协议套件。互联网的一个流行的应用是万维网，它是一组分布式的、可寻址的、链接的资源，通过HTTP在互联网上传递。所有这些技术都为现代计算机的创新提供了一个蓬勃发展的环境。

我希望这本书能让你全面了解计算机是如何工作的。本书涵盖了大量的基础知识，但我们仍然只涉及大多数主题的表面。如果某个特定的领域引起了你的注意，那么我鼓励你继续探索这个主题！还有大量关于计算机的知识有待发现。

设计41：用Python控制自动贩卖机电路

前提条件：设计7和设计8，在这两个设计中，你搭建了一个自动贩卖机电路。一台运行Raspberry Pi操作系统的Raspberry Pi。

在本设计中，你将使用所学的硬件、软件和网络知识构建一个联网的“自动贩卖机”IoT设备。在第6章，你用按钮、LED和数字逻辑门搭建了一个自动贩卖机电路。在这个设计中，你将更新这个设备。保留按钮和LED，用在Raspberry Pi上运行的Python代码代替逻辑门。这将让你轻松地在软件中添加功能，比如通过网络连接到这个设备的功能。

本设计需要如下组件：

❑面包板；

❑LED；

❑与LED一起使用的限流电阻，大约220Ω；

❑两个适合面包板的开关或按钮；

❑跨接线，包括4根公对母导线；

❑Raspberry Pi。

GPIO

除了体积小、成本低之外，Raspberry Pi还有另一个区别于典型计算机的特点——GPIO引脚。每个通用输入/输出（General-Purpose Input/Output，GPIO）引脚都可以指定为电气输入或输出。当一个引脚充当输入时，在Raspberry Pi上运行的代码可以读取该引脚：高电平3.3V或低电平0V。Raspberry Pi甚至还有内部的上拉和下拉电阻，它们通过软件启动，所以你不再需要对输入按钮添加这种电阻。当一个引脚充当输出时，它可以被设置为高电平（3.3V）或者低电平（0V），两者都通过软件控制。有些引脚总是设置为接地、5V或3.3V。这些引脚在软件中用数字编号进行引用。图13-11给出了GPIO引脚名称。

如图13-11所示，GPIO编号与引脚编号不是对应的。引脚用灰色矩形框表示，它简单地从1开始编号到40，从左上角开始到右下角结束。例如，左边第2个引脚是GPIO 2，其引脚编号是3。当在代码中引用这些GPIO引脚时，需要使用GPIO编号，而不是引脚编号。