第三部分　技术架构
Technical Architecture

第三部分 技术架构

Technical Architecture

总图 4 云计算的技术架构部分的章节逻辑结构

第15章　云计算的技术综述

15.1　云计算的技术发展回顾

云计算是伴随着 IT 不再重要的理念而提出的，最初概念是，企业和个人将自己数据、计算能力都放在云端，即网络云里，因为一般网络的图示是一朵云，使用者只要连接上网，就能获取自己数据及计算结果，这样企业可以将原来自己 IT 维护的数据和服务转移到云端，由专门的公司提供维护、支持与服务，企业自己无需 IT，IT 对企业来讲不再是重要部门，因此提出了 IT 不再重要的理念。但在云计算发展过程中，企业自己 IT 通过将在自己机房部署虚拟化系统，运行各种服务软件，提供员工从全世界各地接入，数据与计算在企业网络里，给了这种方式一个私有云的名称，但这显然同最初引入概念有所出入，因此在这里引入云服务，即通过云计算提供服务，是公有云概念延伸，这里强调的服务是云计算的最大特点，强调是通过互联网提供的不是产品，是服务，使用方式如同目前电信、电网或煤气网络一样的服务，按照使用付费，而不是按照购买产品所有权付费。

云计算实际上是一个商务和技术的结合，同时也是商务和技术共同推动的结果。实际上技术是基础，商务是应用。这一部分会从技术发展的角度来看云计算的历史。

15.1.1　云计算的技术概念的发展

（1）效应计算（Utility Computing）阶段 [1]

在 1960 年左右，由于计算机设备的价格非常昂贵，远非一般的企业、学校和机构所能承受，于是很多 IT 界的精英们就有了共享计算机资源的想法。在 1961 年，人工智能之父麦肯锡（John McCarthy）在一次会议上提出来“效应计算”这个概念，其核心就是借鉴了电厂模式，具体的目标是整合分散在各地的服务器，存储系统以及应用程序来共享给多个用户，让人们使用计算机资源就像使用电力资源一样方便，并且根据用户使用量来付费。可惜的是当时的 IT 界还处于发展的初期，很多强大的技术还没有诞生，比如互联网等等。虽然有想法，但是由于技术的原因还是停留在那里。

（2）网格计算（GRID Computing）阶段 [2]

在 20 世纪 90 年代中期，这个词用来形容可以让使用者随时（on-demand）获得计算能力（computing power）的技术。伊恩·福斯特（Ian Foster）和其他人假定，通过规范对计算能力使用的协议（protocol），可以推动和建立一个计算网格（computing GRID），类似于实际中的电网。

随后研究人员制定了许多令人兴奋的方式来进一步发展这些想法，例如大型联合系统（TeraGrid, Open Science Grid, caBIG, EGEE, Earth System Grid）。 这些系统不仅按需提供计算能力，而且也提供数据和软件。这些都是研究性或是在科研领域的使用。标准组织（例如，OGF，OASIS）制定了的相关标准。

网格计算是化大规模计算为许多小部分计算的方式，研究的是如何把一个需要非常巨大的计算能力才能解决的问题分成许多小部分，然后把这些小部分分配到许多低性能的计算机来处理，

最后把这些结果综合起来解决大问题。可惜的是，由于网格计算在商业模式、技术和安全性方面的不足，使得其并没有在工程界和商业界取得预期的成功。

（3）云计算（Cloud Computing）阶段 [3]

云计算的核心与效用计算和网格计算非常类似，也是希望 IT 技术能像使用电力那样方便，并且成本低廉。但与效用计算和网格计算最大的不同是，现在的商务需求已经成熟，特别是 SaaS 的发展，驱动了云计算服务的发展，同时在技术方面也已经基本成熟了。

15.1.2　云计算的相关技术的发展

在技术层面，云计算是在分布式计算、并行计算、虚拟化技术和海量存储技术基础之上发展而来，基础设施即服务（IaaS）其实就是虚拟化发展历程，而平台即服务（PaaS）更多的是分布式并行计算和海量存储发展历程，而软件即服务（SaaS）发展更早，其理念更像商业层面结果，对云计算技术层面似乎并不紧密相连，但随着发展，软件即服务（SaaS）往往会依赖于平台即服务（PaaS）和基础设施即服务（IaaS）技术，是综合使用云计算实现商业目标的范例。

1. 并行计算（Parallel Computing）

云计算是在并行计算之后产生的概念，是由并行计算发展而来，两者在很多方面有着共性。但二者并不等同，主要是二者解决问题和动机是不一样的，但解决问题的手段有很多相同之处。

并行计算或称平行计算是相对于串行计算来说的，所谓并行计算是指同时使用多种计算资源问题的过程，为执行并行运算，通常的计算资源包括一台或多台配有多 CPU 或计算处理单元的计算机和网络资源，并行计算的主要目的是快速解决大型且复杂的计算问题。

为利用并行计算，通常计算问题表现为以下特征 [4] ：

- 将工作分离成离散部分，有助于同时解决。
- 随时并及时地执行多个程序指令。
- 多计算资源下解决问题的耗时要少于单个计算资源下的耗时。

云计算与并行计算的联系与区别如下 [5] ：

（1）云计算萌芽于并行计算

云计算的萌芽应该从计算机的并行化开始，并行机的出现是人们不满足于 CPU 摩尔定率的增长速度，希望把多个计算机并联起来，从而获得更快的计算速度。这是一种很简单也很朴素的实现高速计算的方法，这种方法后来被证明是相当成功的。

（2）并行计算、网格计算只用于特定的科学领域、专业的用户

并行计算、网格计算的提出主要是为了满足科学和技术领域的专业需要，其应用领域也基本限于科学领域。传统并行计算机的使用是一个相当专业的工作，需要使用者有较高的专业素质，多数是命令行的操作，这是很多专业人士的噩梦，更不用说普通的业余级用户了。

（3）并行计算追求的高性能

在并行计算的时代，人们极力追求的是高速的计算、采用昂贵的服务器，各国不惜代价在计

算速度上超越他国，因此，并行计算时代的高性能机群是一个“ 快速消费品 ”，世界 TOP500 高性能计算机地排名不断地在刷新，一台大型机群如果在 3 年左右不能得到有效的利用就远远的落后了，巨额投资无法收回。

（4）云计算对于单节点的计算能力要求低

云计算时代并不去追求使用昂贵的服务器，也不用去考虑 TOP500 的排名，云中心的计算力和存储力可随着需要逐步增加，云计算的基础架构支持这一动态增加的方式，高性能计算将在云计算时代成为“ 耐用消费品 ”。

2. 分布式计算和网格计算 [6]

所谓分布式计算（Distributed Computing）是一门计算机科学，它研究如何把一个需要非常巨大的计算能力才能解决的问题分成许多小的部分，然后把这些部分分配给许多计算机进行处理，最后把这些计算结果综合起来得到最终的结果。 最近的分布式计算项目已经被用于使用世界各地成千上万位志愿者的计算机的闲置计算能力。

分布式要解决的项目都很庞大，需要惊人的计算量，仅仅由单个的电脑或是个人在一个能让人接受的时间内计算完成是决不可能的。在以前，这些问题都应该由超级计算机来解决。但是，超级计算机的造价和维护非常的昂贵，这不是一个普通的科研组织所能承受的。随着科学的发展，一种廉价的、高效的、维护方便的计算方法应运而生——分布式计算！

分布式计算是近年提出的一种新的计算方式。所谓分布式计算就是在两个或多个软件互相共享信息，这些软件既可以在同一台计算机上运行，也可以在通过网络连接起来的多台计算机上运行。分布式计算比起其它算法具有以下几个优点 ：

- 稀有资源可以共享。
- 通过分布式计算可以在多台计算机上平衡计算负载。
- 可以把程序放在最适合运行它的计算机上。

其中，共享稀有资源和平衡负载是计算机分布式计算的核心思想之一。

实际上，网格计算就是分布式计算的一种。如果我们说某项工作是分布式的，那么参与这项工作的一定不只是一台计算机，而是一个计算机网络，显然这种“ 蚂蚁搬山 ”的方式将具有很强的数据处理能力。网格计算的实质就是组合与共享资源并确保系统安全。

分布式计算使用的操作系统包括分布式操作系统、网络操作系统、基于中间件的操作系统。其中分布式操作系统又包括多处理器系统和多机系统，这个应该很好理解，多处理器系统肯定只有一个操作系统，多机系统的分布式也是只有一个操作系统分配机器资源，这样的分布式系统机器与机器之间具有非常高的透明性，而网络操作系统，基于中间件的操作系统，都是由多个计算机组成，每个计算机有独立的操作系统。

网格计算是（GRID Computing）分布式计算的一种。网格计算是伴随着互联网而迅速发展起来的，专门针对复杂科学计算的新型计算模式。这种计算模式是利用互联网把分散在不同地理位置的电脑组织成一个“ 虚拟的超级计算机 ”，其中每一台参与计算的计算机就是一个“ 节点 ”，而整个计算是由成千上万个“ 节点 ”组成的“ 一张网格 ”，所以这种计算方式叫网格计算。这样组

织起来的“虚拟的超级计算机”有两个优势，一个是数据处理能力超强；另一个是能充分利用网上的闲置处理能力。

网格计算和云计算有相似之处，特别是计算的并行与合作的特点；但他们的区别也是明显的。主要有以下几点[7]：

- 网格计算的思路是聚合分布资源，支持虚拟组织，提供高层次的服务，例如分布协同科学研究等。而云计算的资源相对集中，主要以数据中心的形式提供底层资源的使用，并不强调虚拟组织的概念。
- 在对待异构性方面，二者理念上有所不同。网格计算用中间件屏蔽异构系统，力图使用户面向同样的环境，把困难留在中间件，让中间件完成任务。而云计算实际上承认异构，用镜像执行，或者提供服务的机制来解决异构性的问题。当然不同的云计算系统还不太一样，像 Google 一般用比较专用的自己的内部的平台来支持。

总之，云计算是以相对集中的资源，运行分散的应用（大量分散的应用在若干大的中心执行）；而网格计算则是聚合分散的资源，支持大型集中式应用（一个大的应用分到多处执行）。但从根本上来说，从应对互联网的应用的特征特点来说，他们是一致的，为了完成在 Internet 情况下支持应用，解决异构性、资源共享等问题。

3 . 虚拟化技术和海量数据处理技术

在随后的章节中会专门介绍虚拟化技术和海量存储技术。这里，只对这些技术的概念和发展历史做些简要的介绍以给大家一些思路。

（1）虚拟化技术

在计算机技术中，虚拟化（virtualization）是将计算机物理资源如服务器、网络、内存及存储等予以抽象、转换后呈现出来，使用户可以比原本的组态更好的方式来应用这些资源。这些资源的新虚拟部份是不受现有资源的架设方式、地域或物理组态所限制[8]。

通常虚拟化的目标是管理任务的集中，同时要提高整体硬件资源可扩展性和利用率。随着虚拟化，多个操作系统可以在单一的 CPU 下并行运行。这种并行与多任务处理（multitasking）是不同的。多任务处理是相同的操作系统（OS）中运行几个程序。

使用虚拟化技术，企业可以更好地管理和更新操作系统和应用程序，而无需中断用户。传统的“一台服务器一个应用程序”，模式导致的最大问题就是资源的未充分利用[9]。

虽然虚拟化技术在最近几年才开始大面积推广和应用，但是如果从其诞生时间来看，可以说它的历史源远流长。1959 年，克里斯托弗（Christopher Strachey）发表了一篇学术报告，名为“大型高速计算机中的时间共享”（Time Sharing in Large Fast Computers），他在文中提出了虚拟化的基本概念，这篇文章也被认为是虚拟化技术的最早论述。可以说虚拟化作为一个概念被正式提出即是从此时开始。

最早在商业系统上实现虚拟化的是 IBM 公司在 1965 年发布的 IBM7044。它允许用户在一台主机上运行多个操作系统，让用户尽可能充分地利用昂贵的大型机资源。随后虚拟化技术一直只在大型机上应用，而在 PC 服务器的 x86 平台上仍然进展缓慢。这是因为当时 x86 平台的处理能

力非常单薄。

随着 x86 平台处理能力与日俱增，1999 年，VMware 在 X86 平台上推出了可以流畅运行的商业虚拟化软件。从此虚拟化技术终于走下大型机，来到 PC 服务器的世界之中。在随后的时间里，尤其是 CPU 进入多核时代之后，PC 具有了前所未有的强大处理能力，虚拟化技术在 x86 平台上得到了突飞猛进的发展 [10]。

（2）海量数据处理技术 [11]

海量数据（mass data）处理技术实际上涉及到存储和数据库两个方面。

① 对象存储技术

对象存储技术提供基于对象的访问接口，将 NAS 和 SAN 两种存储结构的优势进行了有效地整合，通过高层次的抽象，使之既具有 NAS 的跨平台共享数据和安全访问的优点，同时又具有 SAN 的高性能和可伸缩性的优点。

对象存储模式一般由 client、MDS（Metadata Server）和 OSD（Object Storage Device）三部分组成。

- Client 为客户端，用来发起数据访问 ；
- MDS 为服务器，用来管理对象存储系统中的元数据并保证访问的一致性 ；
- OSD 为存储对象数据的设备，它是一个智能设备，包括处理器、RAM 内存、网络接口、存储介质等以及运行在其中的控制软件。

对象存储设备（OSD）中，将对象（object）作为对象存储的基本单元，每个对象具有唯一的 ID 标识符。对象由对象 ID、对象数据的起始位置、数据的长度来进行访问。对象提供类似文件访问的方法，如 create、open、close、read、write、对象属性等 ；对象的数据包括自身的元数据和用户数据，其中，元数据用于描述对象特定的属性，如对象的逻辑大小、对象的元数据大小、总的字节大小 ；用户数据用来保存实际的二进制数据。对象分为根对象、组对象和用户对象。根对象定义了存储设备以及存储设备本身的不同属性 ；组对象为存储设备上对象提供了目录 ；用户对象存储实际应用数据。

对象存储模式的特性使其在处理海量数据存储请求时具有较大优势，主要体现在 ：

- 高性能数据存储 ：访问节点有独立的数据通路和元数据访问通路，可以对多个 OSD 进行并行访问，从而解决了当前存储系统的一个性能瓶颈问题。
- 跨平台数据共享 ：由于在对象存储系统上部署基于对象的分布式文件系统比较容易，所有能够实现不同平台下的设备和数据的共享。
- 方便安全的数据访问 ：I/O 通道的建立及数据的读写需要经过授权许可才能进行，从而保证了数据访问的安全性 ；另一方面，任何 client 都可以通过对象存储系统提供的标准文件接口访问 OSD 上的数据，统一的命名空间使 client 访问数据的一致性得到了保证。
- 可伸缩性 ：对象存储模式具有分布式结构的特性。由于 OSD 是独立的智能设备，可以通过增加 OSD 数量，使存储系统的聚合 I/O 带宽、存储容量和处理能力得到提高，这种平衡扩展模式使得存储系统能够具有良好的可伸缩性。

- 智能的存储设备 ：OSD 中集成了部分的存储管理功能，因此 OSD 具有一定智能的自主存储功能。

② 数据库策略

实现高性能的海量数据存储可采取的数据库策略有 ：

- 分区技术 ：为了更精细地对数据库对象如表、索引及索引编排表进行管理和访问。可以对这些数据库对象进行进一步的划分，这就是所谓的分区技术。
- 并行处理技术 ：为了提高系统性能，可以让多个处理器协同工作来执行单个 SQL 语句，这就是所谓的并行处理技术。

15.2　云服务的技术结构

云计算业务纷杂，按照目前流行标准，划分为基础设施即服务（IaaS）、平台即服务（PaaS）和软件即服务（SaaS）。基础设施即服务（IaaS）目标是在网上提供虚拟的硬件、网络等基础设施，用户可以使用该服务部署自己系统及软件，以实现自己的业务需求。而平台即服务（PaaS）是在网络中提供虚拟平台 API，使用者无限关系平台功能实现与部署，使用平台功能，实现自己业务应用。PaaS 可以认为是比 IaaS 高一个层次，提供公用业务服务，更容易实现用户业务。软件即服务（SaaS）历史更长，其实是实现了可配置业务服务，用户仅仅需要通过配置，以完成自己需要的业务功能，因此 SaaS 一般直接面向最终用户，实现云计算。[12]

在技术层面，云计算是在互联网服务技术、虚拟化技术、分布式并行计算和海量存储技术基础之上发展而来，基础设施即服务（IaaS）其实就是虚拟化发展历程，而平台即服务（PaaS）更多的是分布式并行计算和海量存储发展历程，而软件即服务（SaaS）发展更早，其理念更像商业层面结果，对云计算技术层面似乎并不紧密相连，但随着发展，软件即服务（SaaS）往往会依赖于平台即服务（PaaS）和基础设施即服务（IaaS）技术，是综合使用云计算实现商业目标的范例。

1. 软件即服务（SaaS）

   SaaS 的发展时间最长，是以二代互联网技术为依托，实现已经存在的企业软件 IT 服务，目前大多数在网络上提供专门服务的都是 SaaS 业务，如在线培训、在线商城，这些业务是直接面对最终用户提供服务的，发展时间最长，也最成熟，如 SOA 构架等一系列构架演进，都是依托 SaaS 发展而来。

2. 平台即服务（PaaS）

   PaaS 是在 SaaS 基础上发展而来，如 Salesforce，最初是 CRM 的 SaaS 服务，但随着业务发展，提供销售相关业务平台，客户的各种 SaaS 业务纷纷集成和部署在 Salesforce 平台上，造就了目前 Salesforce 的辉煌，这正是 PaaS 比 SaaS 更喜人的地方，再如淘宝，最初网上商城，到现在的支付、商城一体化平台，支持小企业网上销售、推广。但是 PaaS成熟是最近几年的事情，PaaS 构架探讨还远不如 SaaS 那么丰富。

3. 基础设施即服务（IaaS）

   IaaS 是在以前 ISV 基础上发展而来，典型如亚马逊云服务，在国内，成熟的 IaaS 服务商

并不多，比较亚马逊的云服务，大都在初级阶段，主要表现在其服务与支持手段还远没有完善。

15.2.1　云服务的技术构架层次

从云服务构架层次上来划分，IaaS 是基础，然后是 PaaS 和 SaaS。整体结构如图 15-1 所示 ：

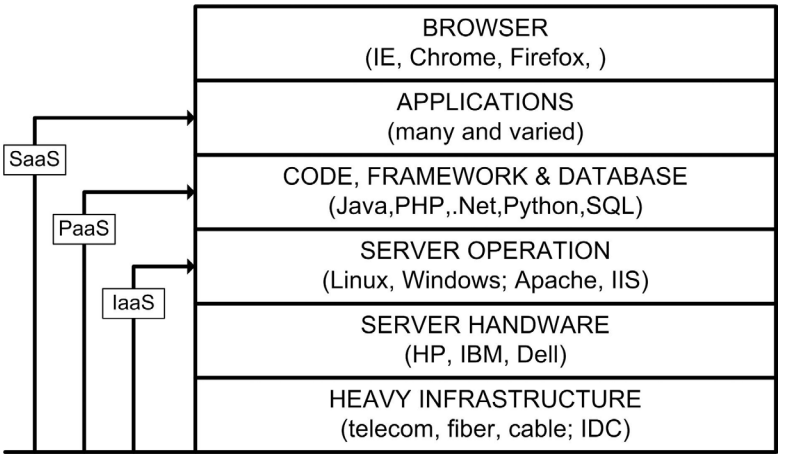


图15-1 云服务的技术构架层次

在 IaaS 层，服务于用户的是基础设施，如计算机，包括 CPU、内存、磁盘空间、网络连接等基础设备，此外还有操作系统等基础软件，其计费往往以 CPU、内存、存储空间和网络流量等使用收费。用户使用的一般都是虚拟机，因此 IaaS 服务是虚拟化技术发展的产物，如果希望构架 IaaS 服务，首先是对虚拟化技术了解。

PaaS 服务是在基础层之上提供中间件，让用户能够快速开发部署 SaaS 应用，这些应用开发是对原始 PaaS 应用扩展，使其能够快速开张业务，比如网络培训平台，当培训公司在其上部署自己应用，针对自己专业、客户提供服务，但一般的培训公司，更专注自己专业和流程，并不是实时通讯的专家，而培训平台能够提供这些功能，使得培训公司从自己不熟悉领域解放出来，更关注自己专业能力，更好更快提供服务给自己客户。这是 PaaS 平台特点。

IaaS 和 PaaS 有些界限并不是很明显，如亚马逊是一家 IaaS 服务公司，但也提供统一数据库服务，用户可以租用数据库，不用关心数据同步、备份等一系列问题，这些是 PaaS 功能，但被集成到 IaaS 服务中。

SaaS 服务是面向客户的应用，是基于 PaaS 开发，并可使用 IaaS 部署的服务，因此构建云服务时，要同时了解 IaaS、PaaS 和 SaaS 特点，有针对性设计构架。

15.3　云计算服务对技术团队带来的挑战

云计算服务对技术团队带来的挑战可以归纳成几点（见图 15-2）：

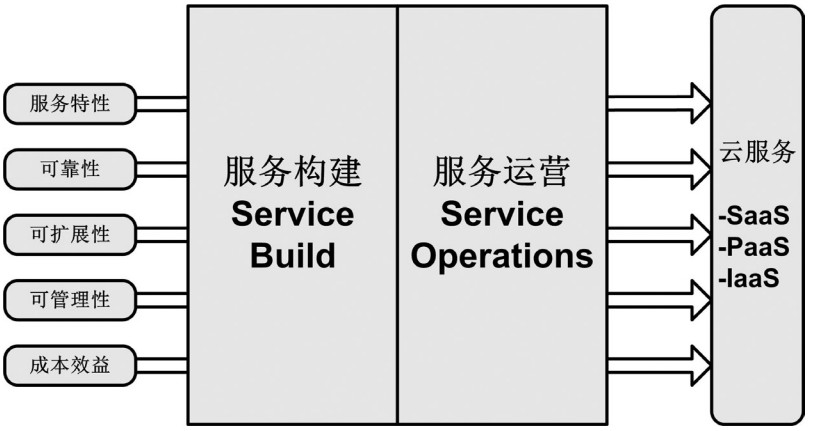


图15–2 云计算服务对技术团队带来的挑战

- 服务特性（Service Features）
  - 通用性和个性化对服务实现的挑战，云计算服务实现不针对特定的客户应用，在“云”的支撑下可以构造出千变万化的应用，同一个“云”可以同时支撑不同的应用运行。这要求产品设计和开发很好处理通用性和个性化的需求之间的平衡。
- 可靠性（Service Availability）
  - 云计算服务往往服务众多客户，遍布全球，需要 24 小时提供不间断服务，需要提供 99.99% 或更高的服务可用性。
- 可扩展性（Scalability）
  - “云”的规模可以动态伸缩，满足应用和用户规模增长的需要。这要求平台需要在构建是就可以满足随即可以扩展的要求。
- 可管理性（Manageability）
  - 运营商的大规模数据中心的管理，比如 Google 云计算已经拥有 100 多万台服务器，Amazon、IBM、微软、Yahoo 等的“云”均拥有几十万台服务器。企业私有云一般拥有数百上千台服务器。这些都需要平台的可管理性。
  - 用户对服务的自主管理，如自主进行计算能力的增加或减少。
- 成本效益（Cost Efficiency）
  - IaaS/PaaS/SaaS 的建设需求大量的投入。这需要云服务提供商能够在设计和建立阶段，都要以成本效益为重要的考量。

15.3.1　对研发团队的挑战

提到“云”计算，大家首先会想到 Google File System、MapReduce、BigTable、Hadoop、NoSQL 等技术因素，会对研发团队带来挑战。但是，这些不是研发团队的唯一挑战。如何结合这些技术，实现自己的商务意义上的“云”服务，应对来自市场对“云”服务的根本要求，从传统软件开发，转向“云”服务软件开发，才是研发团队面临的最大挑战。

本质上讲，“云”服务是将传统企业 IT 服务互联网化，通过集中规模效应及引入新的服务模式，降低企业 IT 开销，“云”服务，无论是 SaaS、PaaS 还是 IaaS，对于研发，都会面对多租户、大容量、高可用、海量数据以及高安全性的挑战。

面对多租户环境，研发团队会面对标准化与个性化功能问题，固有的软件配置方式与多租户的个性化配置有着本质差异，同时，软件功能灵活性要求比传统 IT 软件要高很多。而多租户也带来了服务站点、个性化用户体验等一系列新的要求，对服务软件开发提出了挑战。

传统企业级 IT 软件，同时在线使用的是一个企业内部人员，根据企业的工作时间，中断服务维护是常见的方式，而对于“云”服务，服务于以百万计的企业，服务中断结果往往是灾难性的，这样对系统可靠性要求就显著提高了，如何实现高可靠性是对研发和技术运营的一个更大挑战。

多租户和大用户量是“云”服务的基本特征，“云”服务是以规模效应替代企业传统 IT 服务的，只有在大规模情形下，才能降低成本，替代企业传统 IT，而大用户量意味着高并发及海量数据，如何提升计算能力、储存能力，降低服务成本，稳定一致的支持大容量及海量数据是研发团队面对的另一个挑战。

对于云服务平台构架如何应对这些挑战，将在技术架构部分的后续的章节中详细讨论。

15.3.2　对技术运营团队的挑战

如前面所讲，“云”服务需要面对多租户及高可靠的需求，这些都对运营团队带来了不少挑战。

首先，传统 IT 运营，问题处理是面对一个企业，使用模式相当单一，问题容易定位处理，而“云”服务系统，使用企业的模式千差万别，产生问题是面对不同功能配置、不同流程的问题，问题处理就复杂很多。

其次是用量问题，系统在不停的增加不同行业、不同地域的企业，对系统、网络、存储的使用是极端不均衡的，如何应对在瞬间压力上升时，对网络、存储及计算能力产生的冲击，如何预测控制这些峰值出现，做好应对准备，都是技术运营的巨大挑战。

除了这些因素，多租户将一部分配置交给客户配置管理，根据用户站点配置是这样配置的一个特点，这些原来由运维团队完成的配置，由用户来完成，就不可避免的提升了系统维护的难度，如何应对客户配置对系统冲击，也是运营团队的挑战之一。

最后，高可用度、7x24 不间断服务、日常维护、升级等，都要求能够在服务不中断方式下完成，同时对于任何一个故障，都要能够有相应应对，保障服务不中断，这成为了运营团队的巨大挑战。

15.4　写作重点与章节结构

在这部分的技术讨论中，着眼点是怎样实现一个可靠的和可运营的云服务。在这样的角度上，逐步对上述的研发和运营面对挑战及应对手段做一一分析和讨论。

在章节组织上，从基础的云计算技术开始，然后进入应用层的服务架构讨论。

这部分有单独写作的一章来讨论技术监控体系。这是因为，没有有效的监控，云服务的问题就无法被及时发现和解决，这直接影响到云服务的可用度。监控体系的讨论实际上是服务平台必需具有可管理性的观点的体现。

另外一个单独的章节是质量工程。质量工程的重要性不言而喻。在技术领域中，有很多书籍

和文章在讨论。但是，针对云服务特点而进行的质量工程讨论，则非常的少。在这一章中，对云服务的质量工程，做了比较系统的讨论。

整个技术架构部分章节的逻辑结构如前面总图 4 所示。

在随后的章节中，会针对我们关注的重点做详细地讨论。希望能够帮助云计算服务企业构架自己的技术平台和团队。

参考文献：

[1] Garfinkel & Simson，Utility Computing，Hal. Architects of the Information Society，MIT Press，ISBN 978-0-262-07196-3，1999
[2] Grid computing，Wikipedia，http://www.e-sciencecity.org/EN/gridcafe/what-is-the-grid.html
[3] Cloud computing，Wikipedia，http://en.wikipedia.org/wiki/Cloud_computing
[4] 并行计算，百度百科，http://baike.baidu.com/view/1666.htm
[5] 并行计算就是云计算？云技术百科，http://www.cloudwhy.com/wuqu/2011/0317/123.html
[6] 分布式计算，百度百科，http://baike.baidu.com/view/30655.htm
[7] 网格计算和云计算区别何在，百度，http://zhidao.baidu.com/question/324910551.html
[8] 虚拟化，Wikipedia ，http://zh.wikipedia.org/ 虚拟化
[9] Virtualization ，Wikipedia，http://en.wikipedia.org/wiki/Virtualization）
[10] 张巍，《企业虚拟化实战——VMware 篇》，北京 ：机械工业出版社 , 2009 年 8 月
[11] 蒋然，海量数据存储关键技术浅析，电脑知识与技术，Vol.6，No.20，pp.5403-5405，July 2010
[12] Cloud computing architecture，Wikipedia，http://en.wikipedia.org/wiki/Cloud_computing_architecture

第16章　云技术基础（1）：虚拟化

16.1　虚拟化技术的发展历史

虚拟化的概念在 20 世纪 60 年代首次出现，利用它可以对属于稀有而昂贵资源的大型机硬件进行分区。随着时间的推移，微型计算机和 PC 可提供更有效、更经济的方法来分配处理能力，因此，到 20 世纪 80 年代，虚拟技术已不再广泛使用。但是到了 20 世纪 90 年代，计算机硬件技术的飞速发展，大大降低了硬件成本，也使得企业和组织大量的硬件采购。硬件的激增导致研究人员开始探索如何利用虚拟化解决与其相关的一些问题，例如，服务器硬件利用率不足、管理成本不断攀升、易受攻击和易管理性等。虚拟化技术重新开始被重视和研究。

现在，虚拟化技术日渐成熟，可以帮助企业升级和管理他们在世界各地的 IT 基础架构并确保其安全。虚拟化技术可以扩大硬件的容量，简化软件的重新配置过程。CPU 的虚拟化技术可以让多个系统及应用共享一个 CPU，允许一个硬件平台同时运行多个操作系统，并且应用程序都可以在相互独立的空间内运行而互不影响，从而显著提高计算机资源的利用率。实际上，虚拟化技术的初衷就是为了实现更高的设备利用率，使用户能够尽可能地利用系统资源。也就是说，如果能够在单个服务器上虚拟多个系统，就能够以少数几台计算机完成所有工作，这显然能够节省耗电、空间、冷却和管理开支。考虑到确定服务器利用状况的困难，虚拟化技术需要支持动态迁移（Live Migration）。动态迁移允许操作系统能够迁移到另一台全新的服务器上，从而减少当前主机的负载。

正是由于虚拟化技术能够降低 TCO（总拥有成本），提高资源的利用率和应用上的灵活性，虚拟化技术从原来的大型机、小型机、工作站到目前的 PC 服务器上都得到广泛使用，是云计算服务的基础技术之一，也是基于这一点，大家将企业使用虚拟化部署其 IT 服务软件成为私有云，虽然这和云计算目的与初衷明显有出入。

16.2　虚拟化技术分类

虚拟化技术主要有以下几类 ：

（1）硬件仿真

通过在物理机的操作系统上创建一个模拟硬件的程序来仿真所需要的硬件，并在此程序上运行虚拟机，而且虚拟机内部的客户操作系统无需修改。

- 代表产品 ：微软的 Virtual PC。
- 优点是客户操作系统无需修改，而缺点是速度非常慢，有时速度比物理主机甚至慢 100 倍以上。

（2）全虚拟化

主要是在客户操作系统和硬件之间捕捉和处理那些对虚拟化敏感的特权指令，使客户操作系统无需修改就能运行，速度会根据不同的实现而不同，但大致能满足用户的需求。这种方式是业

界现今最成熟和最常见的，而且属于 Hosted 模式和 Hypervisor 模式的都有。

- 代表产品 ：SUN 的 VirtualBox，KVM，VMware 的 Workstation 及 vSphere 产品。
- 优点是客户操作系统无需修改，速度和功能都非常不错，更重要的是使用简单。缺点是使用 Hosted 模式的产品性能不是特别优异，特别是在 I/O 方面。

（3）半虚拟化

它与完全虚拟化有一些类似，它也利用 Hypervisor 来实现对底层硬件的共享访问，但是由于在 Hypervisor 上面运行的 guest OS 已经集成与半虚拟化有关的代码，使得 guest OS 能够非常好地配合 Hypervisor 来实现虚拟化。通过这种方法将无需重新编译或捕获特权指令，使其性能非常接近物理机。

- 代表产品 ：Citrix 的 XEN，微软的 Hyper-V。
- 优点是与全虚拟化相比，架构更精简，在整体速度上有一些优势。缺点是需要对客户操作系统进行修改，用户体验比较差。

（4）硬件辅助虚拟化

Intel/AMD 等硬件厂商通过对部分全虚拟化和半虚拟化使用到的软件技术进行硬件化来提高性能。硬件辅助虚拟化技术常用于优化全虚拟化和半虚拟化产品，而不是独创一派。

- 代表产品 ：现在市面上的主流全虚拟化和半虚拟化产品都支持硬件辅助虚拟化，包括 VirtualBox、KVM、VMware ESX 和 XEN。
- 优点是通过引入硬件技术，将使虚拟机技术更接近物理机的速度。当然，现有的硬件实现还有很大的优化空间。

（5）操作系统虚拟化

这种技术通过对服务器操作系统进行简单地隔离来实现虚拟化，主要用于 VPS。

- 代表产品 ：有 Parallels Virtuozzo Containers，Unix-like 系统上的 chroot 和 Solaris 上的 Zone 等。
- 优点是因为这个是对操作系统的直接修改，所以实现成本低而且性能不错，缺点是在资源隔离方面表现不佳，而且对客户操作系统及版本都有限制。

16.3　系统虚拟化

虚拟化是资源的逻辑表示，这种表示不受物理限制的约束，它的主要目标是对包括基础设施、系统和软件等 IT 资源的表示、访问、配置和管理进行简化，并为这些资源提供标准的接口来接收输入和提供输出。

虚拟化技术包括两个层面，其一是硬件层面的虚拟化，其二是软件层面的虚拟化。实际上，我们通常所说的虚拟化是指系统虚拟化技术，除此之外，在应用层、表示层、桌面、存储和网络都可以做全方位的虚拟化。

系统虚拟化可以说是最为熟悉的，就是让多个操作系统和应用程序同时运行在不同的虚拟机上，而这些虚拟机建立在同一个物理服务器上。但是一个物理服务器上的虚拟服务器的数量取决

于硬件的能力，所有虚拟服务器共享相同的硬件上，它们之间是相互独立运行，单独的虚拟服务器可以自行升级、启动，不会影响到其他虚拟服务器。

系统虚拟化解决了存在物理服务器环境下问题，通过虚拟化层可以隔离同一台机器上、不同操作系统中运行的程序，避免资源的冲突。另外，服务器虚拟化可以动态移动没有充分利用的硬件资源到最需要应用的程序中，从而能够充分提高底层硬件资源的利用率。

由于其诸多的优点，系统虚拟化技术已成功在桌面、数据中心服务器、云存储等各个方面得到广泛应用。

16.3.1　系统虚拟化的优势

系统虚拟化的优势及好处体现在以下几个方面 ：

（1）软件测试

通过使用虚拟化产品来配置测试环境，不仅比物理方式快捷很多，而且无需购买很多昂贵的硬件，更重要的是，通过它们自带的 snapshot/pause 功能可以非常方便地将错误发生的状态保存起来，这样将极有利于测试员和程序员之间的沟通。

（2）桌面应用

对于企业来说，传统的桌面计算机要消耗大量的硬件资源，并且要面临更多的风险，每一台笔记本和台式机都有遭遇黑客而丢失数据的机会。管理较为复杂，即使采用远程桌面管理，管理员在升级或者排错的时候都要占用员工工作的时间。但是如果企业采用虚拟桌面化应用，此时用户桌面仅仅是运行在服务器上的一个虚拟机，管理员就可以集中来管理用户环境，并保证安全。集中管理也让打补丁等安全措施，以及硬件和软件升级更加方便，开销更少。由于用户行为造成的错误和安全风险也能大大降低。

（3）服务器整合

通过 VMware ESX 和 XEN 等虚拟技术能够将分布到多台物理硬件上消耗不高的物理机上的工作量整合到一台物理机上。现有普遍的整合率在 1:5 左右，也就是使用这些软件能将原本需要五台物理机的工作量整合到一台物理机上。服务器整合不仅能减低硬件成本，场地等开支，还能极大地简化 IT 架构的复杂度，并且在能源电力的消耗上带来极大的成本缩减 。

（4）自动化管理

通过使用类似 DRS（Distributed Resource Scheduling，分布式资源调度）、vMotion、Live Migration（动态迁移）、DPM（Distributed Power Management，分布式电源管理）和 HA（High Availability，高可用性）等高级虚拟化管理技术，能极大地提高整个数据中心的自动化管理程度。

（5）加快应用部署

通过引入虚拟化应用发布格式 OVF（Open Virtualization Format），不仅能使第三方应用供应商更方便地发布应用，而且使系统管理员非常简单地部署这个应用，在系统的安装部署应用，原来安装部署一台服务器的时间在 30 分钟，现在通过集中客户端管理工具，一次同时可维护并部

署多台服务，这样在系统服务安装部署的是时间缩短到 10 分钟，工作效率提高 67%。

（6）集中化管理

通过集中化的客户端工具，可以批量的同时维护上百台甚至一千台的服务器，特别是针对远程维护的用户来说，仅通过集中化的客户端管理工具就可以清楚明白地观看到虚拟服务器的关机、重启等一切重要的操作过程，这对于出现问题的快速定位和解决带来了极大的方便。

16.3.2　系统虚拟化技术的难度

很明显，服务器的有效整合是虚拟化技术的一个优势。利用服务器虚拟化技术，使得一个物理服务器可以被分割成多个虚拟服务器，每个虚拟服务器运行自己的操作系统和存储空间。这恰恰就是提升服务器应用效率进而使得整个数据中心增加其灵活性的有效手段，但在实际应用中，虚拟化技术还存在一些问题阻碍其发展。

（1）不同厂家的虚拟化管理难以兼容

目前，虚拟化厂商做的管理还是基于自己的虚拟机系统，很难跨越它自己的虚拟机系统，而延伸到其他系统，这导致同一厂商软件无法管理其他厂商的虚拟机。

（2）应用软件上不支持

在服务器虚拟化技术如此盛行的今天，许多独立软件应用还是不支持在虚拟机上运行，或者会受到一定的限制。这种情况有多种原因，其中一种是软件开发商未关注虚拟机环境下软件应用。但无论什么原因，缺少厂商支持是限制虚拟化发展的一个主要因素。

但有人会提到，用户可以自己动手来虚拟化那些所谓“不能开展”的应用。但是一旦出现问题，软件商通常不会很轻易提供维护支持。特别是在企业运行的软件不支持虚拟环境条件下，如何让厂商提供支持，恐怕就需要更多方面的努力。

（3）如何平衡性能和优化

对性能和优化，一味地依托虚拟化技术，使得大多数企业盲目地把简单应用和非关键任务的应用全盘采用虚拟机。使得那些关键的 I/O 应用（如数据库和邮件服务器）却不能真正地应用，进而影响性能上的变化，因此企业需要更好的性能管理手段和优化策略。

另外，性能和优化都是客户切实关注的问题。随着虚拟化平台逐渐成熟，硬件平台在不断地改进下，如何很好适应虚拟化技术发展是一个挑战。正是这样原因，虚拟化越来越接近底层，这就要求服务器能够被快速整合和性能的大幅提高，让生产应用大规模在虚拟化技术下效率提升。

16.3.3　系统虚拟化的不足

虚拟化技术的优势给企业带来了极大的方便的同时，也给信息安全、通信网络和管理带来了不小的挑战。

目前可以预料到的包括 ：

（1）对终端安全的挑战：由于系统虚拟化在个人电脑上的广泛应用，传统安装在操作系统中的杀毒、终端监控难以对整个物理计算机提供终端安全防护。这些软件需要与 VMM 配合，甚至做到 VMM 内部。

（2）对数据中心网络接入带来的挑战：由于数据中心开始广泛使用虚拟化，越来越多的服务器被改造成支持虚拟化，数据中心内部的物理网口越来越少，一台物理服务器中的多个虚拟机共享一条上联网线。这带来两方面的问题：首先，单个操作系统和网络端口之间不再是一一对应的关系，从网管人员的角度来看，原来针对端口的策略都无法部署，增加了管理的复杂程度。

（3）对通信安全带来的挑战：既然交换机都需要将其强大的功能延伸进虚拟化的世界，那么保密机、防火墙等信息安全设备是否也需要呢？随着虚拟机的广泛应用，端到端的 IPSec 通道可能位于两个虚拟机之间，IPSec 加密卡能否支持这种功能？防火墙以往针对物理机的访问控制策略，现在是否要针对虚拟机？

16.4　网络虚拟化

网络虚拟化是使用基于软件的抽象从物理网络元素中分离网络流量的一种方式。网络虚拟化与其他形式的虚拟化有很多共同之处。例如，存储虚拟化允许组织将组织内部的所有存储资源整合到一个存储池中，然后再从存储池中分配存储容量。存储虚拟化与企业所使用的存储类型、存储的物理位置无关。

对网络虚拟化来说，抽象隔离了网络中的交换机、网络端口、路由器以及其他物理元素的网络流量。每个物理元素被网络元素的虚拟表示形式所取代。管理员能够对虚拟网络元素进行配置以满足其独特的需求。

网络虚拟化技术也随着数据中心业务要求有不同的形式。 多种应用承载在一张物理网络上，通过网络虚拟化分割（称为纵向分割）功能使得不同企业机构相互隔离，但可在同一网络上访问自身应用，从而实现了将物理网络进行逻辑纵向分割；在上层应用需要多个网络节点承载的情况下，传统的基于冗余的网络设计会很复杂性，网络虚拟化可以将多个网络节点进行整合（称为横向整合），虚拟化成一台逻辑设备，提升数据中心网络可用性、节点性能，同时将极大简化网络架构。

网络虚拟化在应用中又可以分为内部网络虚拟化和外部网络虚拟化。外部网络虚拟化应用于适当的网络中，影响了物理网络中的诸多元素，比如布线、网络适配器、交换机、路由器等等。外部网络虚拟化将多个物理网络整合为更大的逻辑网络，或者将单个物理网络划分为多个逻辑网络。

内部网络虚拟化通过在虚拟服务器内部定义逻辑交换机以及网络适配器，首先通过在物理网卡上创建了一个或多个逻辑交换机，然后在逻辑交换机上创建不同的虚拟端口组分配给不同的虚拟机使用。内部虚拟化网络能够连接运行在一台服务器上的两个或多个虚拟机之间，而且同一台物理机上的虚拟机之间的网络流量不会经过物理网络基础设施。内部网络虚拟化最小化了物理网络上的网络流量，是让服务器内部相关的工作负载进行网络通信的一种更快和更有效的方式。

16.4.1　网络虚拟化的优势

网络虚拟化技术允许管理员将多个物理网络整合进更大的逻辑网络中。反之，一个物理网络也可以被划分为多个逻辑网络。或者在虚拟机之间创建纯软件的网络。网络虚拟化为实现提高速度和自动化，加强网络管理，降低成本的目标提供了新的方法。

（1）网络虚拟化使网络多变一

多变一是指将多个网络设备变成一个，甚至将整个网络云通过虚拟化变成一个网络设备。

例如思科一个新的网络设备，可以将一个万兆交换机上的端口接入到一个简单，不需要管理，只起到分流作用的设备上去，将 1 个万兆端口变成十几个千兆端口，用户只需要管理那个中心的万兆交换机就可以了。

网络多变一的好处 ：

- 节省费用 ：如设备费用、电力费用、空间费用、维护费用、线缆费用等。
- 管理方便 ：更少的管理节点、更少的接口数量。

（2）网络虚拟化使网络一变多

将一个网络虚拟成几个单独的网络供不同的部门使用，网络设备可以做到重启单个虚拟交换机，而不会影响另外几个在同一个实体交换机上的虚拟网络。

在网络层面，网络虚拟化也可以将一个大的网络云虚拟成多个小的网络云，来服务于不同的云。

网络一变多的好处 ：

- 组网的灵活性，安全的灵活性和按需投入带来的成本节约。

16.4.2　网络虚拟化分类

（1）核心层虚拟化

核心层网络虚拟化，主要指的是数据中心核心网络设备的虚拟化。它要求核心层网络具备超大规模的数据交换能力，以及足够的万兆接入能力 ；提供虚拟机技术，简化设备管理，提高资源利用率，提高交换系统的灵活性和扩展性，为资源的灵活调度和动态伸缩提供支撑。

（2） 接入层虚拟化

接入层虚拟化，可以实现数据中心接入层的分级设计。根据数据中心的走线要求，接入层交换机要求能够支持各种灵活的部署方式和新的以太网技术。

（3） 虚拟机网络交换

虚拟机网络交换包括物理网卡虚拟化和虚拟网络交换机，在服务器内部虚拟出相应的交换机和网卡功能。虚拟交换机在主机内部提供了多个网卡的互联以及为不同的网卡流量设定不同的 VLAN 标签功能，使得主机内部如同存在一台交换机，可以方便的将不同的网卡连接到不同的

端口。虚拟网卡是在一个物理网卡上虚拟出多个逻辑独立的网卡，使得每个虚拟网卡具有独立的MAC 地址、IP 地址，同时还可以在虚拟网卡之间实现一定的流量调度策略。

16.4.3　网络虚拟化的不足

组织策略和虚拟网络交换机挑战通常使虚拟化网络管理变得复杂。虚拟化管理员经常管理虚拟交换机，这可能和网络管理员产生摩擦，因为网络管理员不再控制网络的某一部分（主机内的部分）。加上同一主机上的虚拟机之间的大量流量都在主机内部而不经过物理网络，这使得使用传统设备监控流量变得困难起来。

从交换机的角度来看，虚拟网络端口与物理网络端口存在一些差异。虚拟机的增加使网络流量猛增，无论是在网络核心还是边缘。10 台或者 20 台虚拟机共享相同的物理网络端口，每台虚拟机都运行很多应用程序，增加了数据量，造成潜在网络瓶颈问题和管理难题，同时增加的网络复杂性会影响性能。除此之外，网络虚拟化增加了交换结构的层级，增加了延迟性、功能损耗和管理复杂度。

16.5　其他虚拟化技术

虚拟化技术包括两个层面，其一是硬件层面的虚拟化，其二是软件层面的虚拟化。实际上，通常所说的虚拟化是指服务器虚拟化技术，除此之外，在应用层、表示层、桌面、存储和网络都可以做全方位的虚拟化。

（1）存储虚拟化

存储虚拟化就是为主机创建物理存储资源的过程。通过虚拟化技术，多个存储介质模块（如硬盘、RAID）通过一定的手段集中管理起来，所有的存储模块在一个存储池中得到统一管理。RAID（Redundant Array of Independent Disk）技术是虚拟化存储技术的雏形，目前使用的存储还有 NAS（Network Attached Storage）和 SAN（Storage Area Network）。

将存储资源虚拟成一个“ 存储池 ”（storage pool），这样做的好处是把许多零散的存储资源整合起来，从而提高整体利用率，同时降低系统管理成本。与存储虚拟化配套的资源分配功能具有资源分割和分配能力，可以依据服务水平协议（Service Level Agreement）的要求对整合起来的存储池进行划分，以最高的效率、最低的成本来满足各类不同应用在性能和容量等方面的需求。特别是虚拟磁带库，对于提升备份、恢复和归档等应用服务水平起到了非常显著的作用，极大地节省了企业的时间和成本。

（2）桌面虚拟化

桌面虚拟化技术是一种基于服务器的计算模型，并且借用了传统的瘦客户端（thin client）的模型，可以让管理员与用户能够同时获得两种方式的优点 ：将所有桌面虚拟机在数据中心进行托管并统一管理，同时用户能够获得完整 PC 的使用体验。桌面虚拟化最大的好处在于能够使用软件从集中位置来配置 PC 及其他客户端设备，这样方便了企业用户集中管理计算机，运维部门可以在数据中心加强对应用软件、系统补丁、杀毒软件的管理和控制。

（3）表示层虚拟化

在本地计算机显示和操作远程计算机桌面，在远程计算机执行存储信息和程序，一般通过终端服务来实现。

（4）应用虚拟化

在一台计算机上显示和操作计算机桌面，在另一台计算机上执行程序和存储信息。

16.6　市场主流虚拟化技术对比

众所周知，提及虚拟化，VMware 可以是当之无愧的领头羊，在虚拟化市场上占有 80% 的市场份额，使得其在虚拟化领域位置无人撼动。但随着各大厂商的进军虚拟化，开源虚拟化的不断成熟，加剧了这个领域内的竞争，让用户有了更多的选择权，最终成熟完善的产品才是用户所期待的。

在开源这条战线上，VMware 面临着 XEN 的挑战，但 XEN 并没有造成像微软那样的威胁，主要原因是 XEN 软件目前还不能很好地支持运行在 Windows 的虚拟机。然而，一旦 XEN 能够变得更加稳定，并且像对 Linux 那样对 Windows 提供无缝支持，恐怕 VMware 将在开放源代码领域面临一个强大的竞争对手。

提到 XEN 虚拟化，不能不提开源平台。XEN 技术是基于 Linux 平台开放源代码的虚拟化技术。但后来 Citrix 公司在 2007 年 8 月以 5 亿美元收购了 XENSource 公司，使得 Citrix 成为开源虚拟化的代表。 但 Citrix 并不是一味的拿来主义，而是在原有的平台上增加了一个完整的图形用户界面功能，同时 XENServer 还比 VMware 便宜，但绝不是免费的。

对大多数 VMware 的用户，产品是否成熟是最主要的考量指标。VMware 虚拟化产品提供集中管理功能，通过图形用户界面能够很好地执行任何管理操作，并有效地进行虚拟机集群管理。而对于开放源码软件来说，这种情况很少在应用中体现。

在谈到 VMware 和 XEN 的功能比较时，在很大程度上他们功能是相同的。实时迁移（Live Migration），这个在 VMware 企业级虚拟化技术广泛应用的技术，XEN 上也有着很强的实施，并且提供了多年的迁移支持。不同的是，VMware 提供存储池技术或存储虚拟化，这些是 XEN 所不能提供的，因为这不是 XEN 的工作。因此，可以客观地说，VMware 在产品成熟度上处于领先位置。

对于 XEN 应用来说，必须确定 IT 团队中有 Linux 专家。如果缺乏 Linux 的系统管理员，你可以使用在 XEN 虚拟化的几个基本功能，但集成其他开源工具和自动化整合将是不可能的。

其他的虚拟化开源组件正在逐步完善并建立应用。就如同 Red Hat 将虚拟化技术由原先的 XENSource 改至 KVM。KVM 最大的好处就在于它是与 Linux 内核集成的。因为 KVM 是与 Linux 内核集成的，所以可以说与 ESX 拥有相同的架构。但是 KVM 能够利用 Linux 驱动程序这一点与 ESX 有很大不同。能够利用庞大的 Linux 社区所提供的程序也是 KVM 的一大优势。而对 KVM 来说，另一个优势就是可靠性和多样化的工具。自从 2006 年 KVM 被集成到 Linux 内核之后，KVM 的可靠性和性能有了很大提高。

然而真正采用 KVM 的 Linux 比较少。很多 Linux 虚拟化的用户群都使用的是 XEN，很多企

业也不会马上迁移到这个平台上来。而且，KVM 的普及还是要取决于用户是否了解 KVM 的优点和用户的信赖程度。

由此可见，尽管开源虚拟化技术可能有更好的前途，但却不得不承认在未来一段时间内依旧无法和 VMware 相提并论，毕竟 VMware 丰富的虚拟化产品线是不可比拟的，相对完整的产品和解决方案也是其另一大特色，与之相比开源虚拟化大都还只是基于 XEN 的虚拟化管理软件而已。

对于其他任何虚拟化技术厂商来说，VMware 是一个强劲的对手。但挑战是，虽然 VMware 在竞争中已经领先了 18 到 24 个月的技术，并且有一个广泛和非常忠诚的用户基础。但是 VMware 的弱点在价格上，其高价让用户产生了很多抱怨。

16.7　由虚拟化引发的云计算

尽管虚拟化技术可以大大降低数据中心硬件成本、管理的复杂度，但企业仍然需要为了使用数据中心而进行的采购硬件、安装软件和日常的系统维护等环节。随着半导体、互联网和虚拟化技术的飞速发展，使全世界范围内的数据中心进行较大程度的集中成为了可能，在达到一定的规模性效应时，用户只是租用虚拟机而不需要大量的购置和维护就成为了可能。

基于虚拟化运行环境的云计算中心这样的构想由此产生，它采用创新的计算模式使用户通过互联网随时获得近乎无限的计算能力，使用户对计算和服务取用自由、按量收费。虚拟化技术的伸缩性和灵活性极大的帮助虚拟化运行环境提高资源利用率，同时简化了资源和服务的管理与维护难度。通过整合成千上万服务器的物理资源，构建资源池，最终以服务的形式按需提供给用户，提供 Linux 和 Windows 系列常用主流操作系统虚拟运行环境，最终让用户使用虚拟和运行环境中的虚拟机像使用本地物理机器一样。

云计算中心的基础架构主要包含计算（服务器）、网络和存储。对于网络，从云计算整个生态环境上来说，可以分为三个层面，数据中心网络、跨数据中心网络以及泛在的云接入网络，如图 16-1 所示。

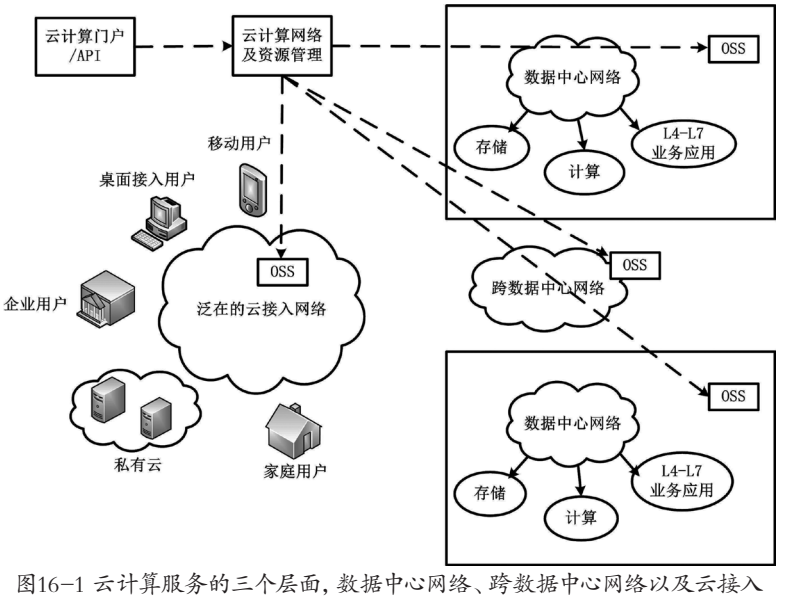


图16-1 云计算服务的三个层面，数据中心网络、跨数据中心网络以及云接入

数据中心网络虚拟化包括核心层虚拟化、接入层虚拟化和虚拟机网络交换。

- 核心层网络虚拟化是数据中心核心网络设备的虚拟化，可提高资源的利用率以及交换系统的灵活性和扩展性，为资源的动态伸缩和灵活调度提供支撑。
- 接入层虚拟化实现数据中心接入层的分级设计，支持新的以太网技术和各种灵活的部署方式。
- 虚拟机网络交换通过虚拟网络交换机和物理网卡虚拟化，在服务器内部形成相应的交换机和网卡功能。

数据中心之间可以通过跨数据中心网络进行计算或存储资源的迁移和调度，可以通过构建大范围的二层互联网络来进行大型的集群计算，也可以通过构建路由网络连接来集成多个虚拟数据中心以提供更大型的云计算服务。

虚拟化是支撑云计算的重要技术基石，云计算中所有应用的物理平台和部署环境都依赖虚拟平台的管理、扩展、迁移和备份，各操作都通过虚拟化层次完成。目前，大部分软件和硬件已经对虚拟化有一定支持，可以把各种 IT 资源、软件、硬件、操作系统和存储网络等要素都进行虚拟化，放在云计算平台中统一管理。

16.8　云计算推动数据中心变革

在传统数据网络系统中心中，数据中心各系统中的网络设备种类增多、并且零散分布。随着业务的迅速增长，这种集成度低的网络架构已经无法提供高度的稳定性和可靠性。由于系统种类的多样，网络设备的分散，也使得整体网络缺乏统一的建设和管理，而且缺乏有效的控制手段，同样给维护人员加大了工作量及工作难度。同时由于系统、网络设备的繁多，也对机房能耗、环境要求、配套要求、设备及线路安装等提出更高的要求。

在新一代数据中心中，所有的服务器、存储器，同时也包括网络等基础设施资源将全部通过虚拟化技术实现，形成三大共享基础设施资源池 ：处理池、存储池以及网络池。共享资源池中的资源可按照每一应用系统的需求被初始化分配与快速部署。

经过多年的技术演变和业务发展，当前数据中心的网络基础架构通常都是采用树状结构，分为接入层、聚合层和核心层。在大多数情况下，数据流从接入层到聚合层再到核心层，然后再返回，层次越多不仅使用的设备会越多，延迟也会增加。网络中每一跳的代价都很高，而且会增加复杂性。由于这些操作的重复和重叠，无法得到想要的性能，也导致了安全性难以保障。一直以来在传统网络中，是依靠不断添加机器来提升网络性能的，但是这种方法增加了数据中心的成本和复杂性。

在大规模采用服务器虚拟化技术的新一代数据中心里，数据流量将主要集中在本地服务器之间的通信，通过路由器和万兆以太网交换机帮助扁平化和简化现有数据中心网络，既可动态及同时支持更多的用户、服务以及带宽，提高性能，也可以帮助用户节省运营时间。虚拟化技术能够通过资源共享与合并资源来提高效率并降低成本，减少数据中心网络资本性支出。

16.8.1　虚拟化数据中心优点

基于虚拟化的数据中心网络架构与传统网络设计相比，有以下几大特点 ：

- 运营管理简化 ：数据中心全局网络虚拟化能够提高运营效率，虚拟化的每一层交换机组被逻辑化为单一管理点，包括配置文件和单一网关 IP 地址。
- 整体无环设计 ：跨设备的链路聚合创建了简单的无环路拓扑结构，不再依靠生成树协议（STP）。虚拟交换组内部经由多个万兆互联，在总体设计方面提供了灵活部署的能力。
- 进一步提高了可靠性 ：虚拟化能够优化不间断通信，在一个虚拟交换机成员发生故障时，不再需要进行 L2/L3 层的重收敛，能决速实现确定性虚拟交换机的恢复。
- 安全整合 ：安全虚拟化在于将多个高性能安全节点虚拟化为一个逻辑安全通道，安全节点之间实时同步状态化信息，从而在一个物理安全节点发生故障时另一个节点能够接管任务。

采用虚拟化技术构建的数据中心具有现有体制数据中心不具备的优势，包括 ：

- 快速业务开通的能力

  由于对业务开通所需要的各类资源实现了虚拟化，实现了系统运行所依赖的虚拟资源与实际的物理资源的采购与管理相互隔离，只要现有资源池中的资源能够满足业务系统的运行要求，就可以通过虚拟化手段为业务系统提供相对独立的虚拟化运行平台，缩短了业务开通时间，降低了业务开通对物理设备的依赖性，避免了设备采购周期对业务开通的不利影响。

- 更低的 TCO（Total Cost of Ownership）

  资源的虚拟化通常伴随着系统的整合，从而提高系统的利用率，如 CPU 利用率可以从 15%至 20%之间提高到 60%以上，同样的硬件资源可以承载更多的应用系统，降低了在设备采购、管理、电力供应、制冷等多方面的成本，从而降低数据中心的整体拥有成本 TCO。

- 更精细的服务质量控制

  在传统的数据中心中，一旦专用应用系统的硬件平台搭建完成，其服务质量就基本确定，对服务质量的调整（提升与降低）相对困难。但是在数据中心中，可以随时调整为应用系统分配的资源以提升或降低其性能、安全性、可用性等指标，从而实现对服务质量的精细管理。

- 更细粒度的管理体系

  资源的虚拟化必定带来对系统监控与管理管理要求的提升，系统的监管不再只局限于硬件系统层，而是拓展到了虚拟化层和平台构建层。由此，可以更准确地根据应用系统的要求分配资源。而且资源的分配也不是一成不变的，在资源分配的过程中可以考虑更多的因素，如时间段和应用量等。

16.8.2　虚拟化数据中心存在的风险

虚拟化数据中心的建设在为企业带来利益的同时，也对数据安全和基础架构提出了新的要求。目前虚拟化数据中心有几个风险点：

（1）高资源利用率带来的风险集中

通过虚拟化技术，提高了服务器的利用率和灵活性，也导致服务器负载过重，运行性能下降。虚拟化后多个应用集中在一台服务器上，当物理服务器出现重大硬件故障是更严重的风险集中问题。虚拟化的本质是应用只与虚拟层交互，而与真正的硬件隔离，这将导致安全管理人员看不到设备背后的安全风险，服务器变得更加不固定和不稳定。

（2）网络架构改变带来的风险

虚拟化技术改变了网络结构，引发新的安全风险。在部署虚拟化技术之前，可在防火墙上建立多个隔离区，对不同的物理服务器采用不同的访问控制规则，可有效保证攻击限制在一个隔离区内，在部署虚拟化技术后，一台虚拟机失效，可能通过网络将安全问题扩散到其他虚拟机。

（3）虚拟机脱离物理安全监管的风险

一台物理机上可以创建多个虚拟机，且可以随时创建，也可被下载到桌面系统上，常驻内存，可以脱离物理安全监管的范畴。很多安全标准是依赖于物理环境发挥作用的，外部的防火墙和异常行为监测等都需要物理服务器的网络流量，有时虚拟化会绕过安全措施。存在异构存储平台的无法统一安全监控和无法有效资源隔离的风险。

（4）虚拟环境的安全风险

- 黑客攻击

  控制了管理层的黑客会控制物理服务器上的所有虚拟机，而管理程序上运行的任何操作系统都很难侦测到流氓软件等的威胁。

- 虚拟机溢出

  虚拟机溢出的漏洞会导致黑客威胁到特定的虚拟机，将黑客攻击从虚拟服务器升级到控制底层的管理程序。

- 虚拟机跳跃

  虚拟机跳跃会允许攻击从一个虚拟机跳转到同一个物理硬件上运行的其他虚拟服务器。

- 补丁安全风险

  物理服务器上安装多个虚拟机后，每个虚拟服务器都需要定期进行补丁更新、维护，大量的打补丁工作会导致不能及时补漏而产生安全威胁。安全研究人员在虚拟化软件发现了严重的安全漏洞，即可通过虚拟机在主机上执行恶意代码。黑客还可以利用虚拟化技术隐藏病毒和恶意软件的踪迹。

16.8.3　虚拟化数据中心风险应对

正是由于虚拟化数据中心存在一定的安全风险，所以数据中心的安全建设尤为重要，那应该如何增强这方面的安全呢？

（1）数据中心网络架构高可用性设计

在新一代数据中心虚拟化网络架构中，通过 IRF（Intelligent Resilient Framework）技术将多台网络设备虚拟化成一台设备统一管理和使用，整体无环设计并提高可用性。在此架构下，基本原则就是服务器双网卡接在不同交换机上，汇聚交换机堆叠后，将两层交换机用多条链路进行捆绑连接，实现基于物理端口的负载均衡和冗余备份。

数据中心架构规划设计时，需要按照模块化、层次化原则进行。从可靠性角度看，三层架构和二层架构均可以实现数据中心网络的高可用，而二层扁平化网络架构更适合大规模服务器虚拟化集群和虚拟机的迁移。在内部网中根据应用系统的重要性、流量特征和用户特征的不同，可大致划分几个区域，以数据中心核心区为中心，其它功能区与核心区相连，成为数据中心网络的边缘区域。

（2）网络安全的部署设计

虚拟化数据中心关注的重点是实现整体资源的灵活调配，因此在考虑访问控制时，要优先考虑对计算资源灵活性调配的程度。网络安全的控制点尽量上移，服务器网关尽量不设在防火墙，避免灵活性的降低。

（3）安全策略的动态迁移

数据中心需针对不同类型的应用系统制定不同级别的防护策略。虚拟化环境下，应用系统和服务器是自由匹配和随需迁移的，每一次虚拟机迁移都对应安全策略的改变和调整，因此发生虚拟机创建或迁移时，需要利用虚拟机软件保证虚拟机在服务器上的快速迁移，同时要保证网络配置的实时迁移，以确保虚拟机业务的连续性。目前业界最优的解决方法，即在服务器邻接的物理交换机采用 VPORT 的概念。一个虚拟机绑定一个或几个 VPORT，虚 拟机迁移时，只需在邻接的物理交换机上将虚拟机对应的网络配置（profile）绑定到 VPORT 上，而不会对其他虚拟机的 VPORT 产生影响。

（4）存储虚拟化的安全设计

在应用存储虚拟化后，虚拟化管理软件应能全面管理 IP SAN、FC SAN 、NAS 等不同虚拟对象，通过上层应用封装对用户提供一致的管理界面，屏蔽底层对象的差异性。通过基于主机的授权、基于用户认证和授权来实现存储资源隔离和访问控制。采用基于虚拟机技术的行为监控技术，获得上层操作系统真实的硬件访问行为，避免恶意代码通过修改操作系统造成信息隐瞒。通过部署与主机独立的、基于存储的入侵检测系统，对存储设备所有读写操作进行抓取和分析，以检测存储设备中文件 / 属性的改变、检测文件模式的非正常修改、监控文件结构的完整性、扫描检测可疑文件等。

（5）制定相关管理策略

对整个数据中心来说需要制定相关制度，明确责任管理 ；制定专门的虚拟机审核、追踪流程，防止虚拟机蔓延而导致的管理失控 ；利用虚拟化监控工具，检测出未授权的拷贝和“ 克隆 ”虚拟机的行为，确保敏感信息在正确的管控中。

数据中心的发展正在经历从整合和虚拟化到自动化的演变，基于云计算的数据中心则是未来的目标。在现阶段和将来的数据中心，虚拟化技术都扮演着一个非常重要的角色。

16.9　案例研究：微软中国的IT虚拟化方案

16.9.1　背景介绍

服务器与开发工具事业部是微软中国研发集团的核心研发部门之一，部门的服务器数量和管理成本与应用数量呈线性递增关系，导致 IT 管理和成本压力巨大。如何更有效的管理数据中心、促进数据集中、降低 IT 资产及管理成本、如何利用有限的资源投资获得更好的业务成果、提高业务效率和可用性，成为 IT 工程师们面临的迫切问题。微软 IT 工程师们通过部署微软虚拟化技术解决了这些问题并实现了业务的持续增长后对服务与管理的高效运营。[1]

16.9.2　解决方案

通过部署使用 MS Virtual Server，管理员拥有完全脚本化的移动式管理、随时连接的虚拟机、简易的部署自动化变更和配置等功能，并且通过使用随着 Windows Server 2008 操作系统一起发布的 Hyper-V 技术，公司能够使用 Virtual Server 来开始虚拟化的部署，从而深入了解虚拟化的应用和系统管理。通过使用通用虚拟磁盘（Virtual Hard Disk，VHD）文件格式，公司将能够轻松地从 Virtual Server 迁移到 Hyper-V 技术平台。 整体方案的架构如图 16-2 所示。

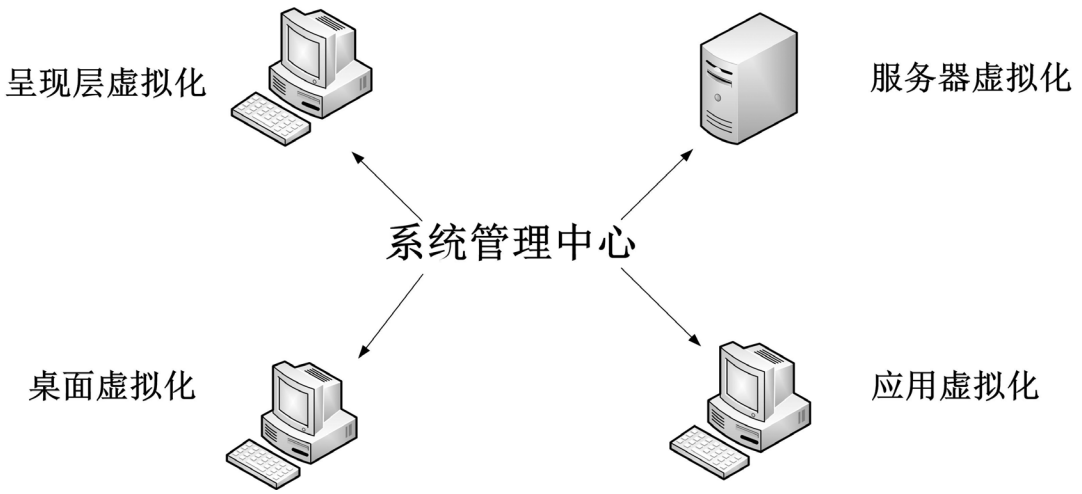


图16-2　微软虚拟化解决方案架构

在部署系统中心虚拟机管理器的选择上，选择基于 Intel 虚拟化技术架构（VT-x）的 Intel 虚

拟化技术架构四核服务器，为 32 位和 64 位虚拟化环境提供支持。VT-x 避免了需要使用复杂的代码提供虚拟化的处理器特权的需求，可为虚拟机中的托管操作系统直接提供所需特权，不仅可以让托管操作系统正常运行，还可以让操作系统相比没有 VT-x 时更有效率。Intel 架构下的虚拟化解决方案促进了 IT 架构日益增强的整合应用程序能力，并对降低成本，优化业务敏捷度上提供了强大的硬件支持。

除了部署系统中心虚拟机管理器以外，微软中国研发集团还部署了其他微软系统解决方案，包括系统中心数据保护管理器（System Center Data Protection Manager）、系统中心运行管理器（System Center Operations Manager）、以及系统中心配置管理器（System Center Configuration Manager）。通过一起使用这些产品，集团的 IT 团队在简化了整个基础架构的管理同时，物理机器和虚拟机器资产的管理与维护也更加便利。

16.9.3　收益分析

以往每年花费在服务器部署上的成本相当昂贵，大多数成本用以服务器的更新，每年花在更新服务器与维护服务器的成本成为 IT 部门的主要开销。通过使用虚拟服务器和系统中心虚拟机管理器，只需通过升级目前的服务器与更新部分服务器硬件，就可以节约出不少于 1/3 的费用。通过最大化服务器的利用率来实现最小化服务器的数量，通过使用系统中心虚拟机管理器来量化系统性能以判断哪些系统可以承担额外的工作任务，让 IT 工程师及时的了解到虚拟化的综合信息，从而满足服务器过去的性能表现和现有的业务需求。在执行虚拟化升级前 IT 工程师发现目前大部分物理服务器的实际使用率只有 50%，通过性能的进一步提升将作为虚拟化的主机它能够在虚拟服务器和系统中心虚拟机管理器的帮助下达到超过 70% 至 85% 的利用率（见图 16-3）。

研发团队		虚拟化前		虚拟化后	
	所需要的服务器数量	单台服务器耗电（瓦/小时）	耗电总量（瓦/小时）	部署的服务器数量	耗电总量（瓦/小时）
WLC	300	550	165,000	20	15,000
STBC	1,310	550	720,500	82	63,880
CRD Group	345	550	189,750	47	32,080
总计	1,955		1,075,250	149	110,960

参考数据
每一度电（千瓦/小时）可以节省： · US $ 0.16, 和 0.778公斤二氧化碳(参考来自美国国家环境保护局)
每个服务器机架大约0.48平方米
2006年上海每户家庭年均耗电量大约为2832度电

研发团队	节能减排结果			
	节约电费总量（US$/年）	节约空间（平方米）	减少排放二氧化碳（公斤/年）	节省的电力每年可上海的家庭(户)
WLC	210,240	550	1,022,292	465
STBC	920,319	550	4,475,049	2,031
CRD Group	220,990	550	1,074,565	487
总计	1,351,549	-	6,571,906	2,981

图16–3　微软中国研发集团部署虚拟化前后能源消耗对照

16.9.4　经验总结

（1）简化虚拟环境管理

通过使用系统中心虚拟机管理器，从时间的节省和方便的管理当中获得了很大的收益，只需要一个工具来操作和管理整个虚拟数据中心，从而使 IT 工程师从繁忙的工作中解放出来。

（2）简单的部署

系统中心虚拟机管理器的部署过程非常简单。例如，通过安装向导将会非常清楚地了解到虚拟机管理器是如何工作的，该使用哪个端口用作哪个功能，这与旧的安装方式完全不同，过去常常需要进入注册表以进行修改。

（3）灵活的数据存储

虚拟化的存储对日常的数据备份、恢复带来了更加安全的解决方案。通过对多用户或应用程序提供一种访问存储内容的方法，将无需担心其地理位置或管理方式。它使环境中的物理存储能够跨多个应用服务器共享，并且能够查看和管理虚拟化层背后的物理设备。通过虚拟化存储网络支持技术，能够向服务器掩饰或隐藏未授权访问的卷，从而提供更高的安全级别。并能够动态更改和增加卷以满足各个服务器的需要。

（4）虚拟化使得服务连续性和服务恢复变得简单

虚拟化技术可以让所有资产的维持可持续性和维持灾难恢复战略的过程变得简单。通过划分负载，可以防止 一个应用程序影响其他程序的性能，或导致系统崩溃。 就算不怎么稳定的遗留应用程序也可以运行在安全、被隔离的环境中。

全面的虚拟化策略 还可以维护随时可用的容错规划，在发生意外时保证业务连续性。通过将操作系统和应用程序实例转换为数据文件，可以帮助实现自动化和流线化的备份、复制及供应更稳健的业务连续性，并加快故障或自然灾难后的恢复速度。

参考文献：

[1] 微软，微软中国研发集团虚拟化解决方案，百度文库，http://wenku.baidu.com/view/5e899d563c1ec5da50e27081.html

第17章 云技术基础（2）：大数据的存储与数据库

目前，随着云服务的使用，企业 IT 服务模式也正在发生着巨变，我们已经逐渐进入了大数据（Big Data）时代，而传统数据库在处理每日上亿条数据，已经是力不从心了。而且，就数据使用方式来讲，也正在发生着革命性的变化，以前在企业内部，数据是专业化、部门化使用的。如销售使用 CRM，数据单独存储，为 CRM 使用部门专有。在企业内部，跨部门使用数据，也是不可能的，就不要谈跨企业使用数据了，而大数据时代，行为预测是建立在对尽可能完整的数据分析基础上的，因此，数据进入大融合时代，当企业员工在使用系统时，也不再是面对一个一个孤立系统，而是建立在完整数据分析之上的建议与决策，这样能够极大提升企业效率，但对数据的存储、交换与分析，提出了更高需求。分布式存储是大数据时代存储的典型方式，目前提供与服务的巨头如 Google、Amazon、Facebook 和 Yahoo 都投入大量人力物力，在分布式存储及 NoSQL 数据库方面进行开发。企业级的分布式存储方案也有很多提供商，但总体来讲 Google 的 GFS 以及 Hadoop 的 HDFS 是这个领域的领头，特别是开源的 Hadoop，已经形成了一个完整的开源方案，并且有 Amazon、Yahoo 和 Facebook 等公司的持续支持和实践，已经趋于成熟。

17.1 分布式存储

17.1.1 云服务对存储与数据的挑战

云存储架构是云存储组建的一个重要环节，尤其是对以后的服务十分重要。一个好的云存储架构能为以后的服务水平起到一个很好的提升作用。而云存储架构成功的关键是创建一个能灵活按需分配资源给每一个应用的基础架构。2009 年 4 月 SNW 成立了 SNIA 云存储架构工作组，在 2010 年 4 月 12 日公布第一个云存储标准 CDMI 1.0，目前最新版本是 1.0.1。为了理解云存储架构的挑战，这里再重温当年 SNIA 评估新的或现有的存储解决方案合适性的时候，应该考虑的问题，对于理解云存储架构面临挑战十分有意义 ：

（1）能够灵活地扩展吗？类似于你对虚拟服务器做的事情，你需要能够迅速且以最少的开销分配、增加、减少并重新分配存储。

（2）能够自动化存储管理流程吗？你进行配置、备份和复制等常规操作的自动化程度越高，你的环境的可扩展性就越强。

（3）能够测量和汇报使用情况吗？为了实施云存储架构服务，你必须将资源理解为你的服务需求的一个用户，能够汇报实际的使用情况，并且现在或将来能够按照资源的使用情况进行收费。

（4）能够自由移动数据吗？如果你的数据束缚在缺乏灵活性的存储里，有效性和可用性将变差。

（5）能够在确保资源足够安全的时候建立多租户吗？允许多个业务单元或独立实体分享同一存储硬件是有效云存储架构的一个必要条件。

（6）能够提升云存储架构效率吗？第一步是提高利用率，除此之外，削减开销、自动精简配

置和消除冗余都有助于提高效率。

（7）能够有效保护你的数据吗？成功实现云存储架构的一个关键是集成你所有的流程，让它们变得简单、可重复和有效。具有合适的策略级别、覆盖你提供的每项服务的一致数据保护和灾难恢复流程是基本的。

（8）能够在单一网络结构上做所有的事情吗？ FCoE 的到来让在单一以太网结构上整合 SAN 和 LAN 以获得更低的成本和更高的灵活性成为可能。

（9）能够支持服务器虚拟化吗？假设服务器虚拟化将是你云存储架构的一个关键组成部分，你将需要存储与你现在正在使用和将来很可能采用的任何虚拟化解决方案紧密集成在一起。

云服务面向多租户大用户量的特点决定了云服务系统数据量是海量的，而数据是积累增长的，因此如何构建自己存储架构，应对数据急剧增加，特别是在数据增加时访问效率如何保证，是存储构建过程中最核心的问题。而数据是一切应用的基础，数据效率很大程度决定了系统效率。灵活高效 、安全可靠是对云服务存储的最大挑战。

分布式存储是针对这些问题而提出的方案，经过 Google、Amazon 及 Facebook 等企业的应用于实践，能够满足日益增长的数据存储与分析需求。

17.1.2　云服务存储构架

目前，在云计算中广泛使用的数据存储系统是分布式文件系统，如 Google 的 GFS 和 Hadoop HDF 系统，运行在廉价的普通硬件上，提供多备份支持的容错功能，能够有效利用服务器上配置硬盘，形成海量存储空间，目前 GFS 支持服务器节点以十万计，而 Hadoop HDF 也有过万节点系统在运行。无论是 GFS，还是 HDF，虽然在具体处理技术上各自有各自的实现，但就总体构架来看，基本上都是下面结构（见图 17-1）：

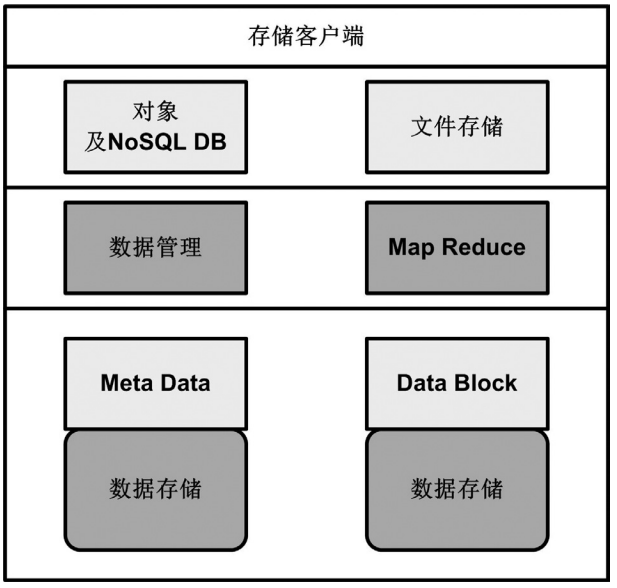


图17-1　云计算数据存储系统

在存储层面，将数据分成一个一个块（block）。Meta-data 是块的相关信息，如同文件系统的目录节点文件中对文件块描述，可以理解为具体数据块的定位信息，当然包含了备份数据块信息。这里 meta-data 是数据存储的目录结构，以数据块为对象。数据块是实际数据存储，被分配到系统数据节点上的块设备上，一般是硬盘。这一层可以理解成像 OS 中的块设备驱动，不同的是分布式的。

在存储层面之上是数据处理层，主要目标是将块组合成用户需要的数据，如对象，或者文件，这时对于大量对象与大文件访问时，为了提升性能，往往需要通过 MapReduce，实际是分布式数据处理，提升数据访问效率。

数据存储一般对上暴露有三种不同接口，首先是对象存储，这是一个较低层接口，一般用于程序对象直接存储 ；其次是 NoSQL DB，提供数据库接口，一般在存储上实现数据库都是 NoSQL 的，如 Google 的 Bigtable 和 Hadoop 的 Hbase ；还有文件存储接口，对应用来讲，就是以文件方式访问数据，对于以网页为中心系统，文件接口应用更多。

分布式系统通常通过读写分离，和在写入时同步来保障数据最终一致性的，图 17-2 是 Ceph 支持的三种读写的顺序图，用于理解分布式存储的读写方式 ：

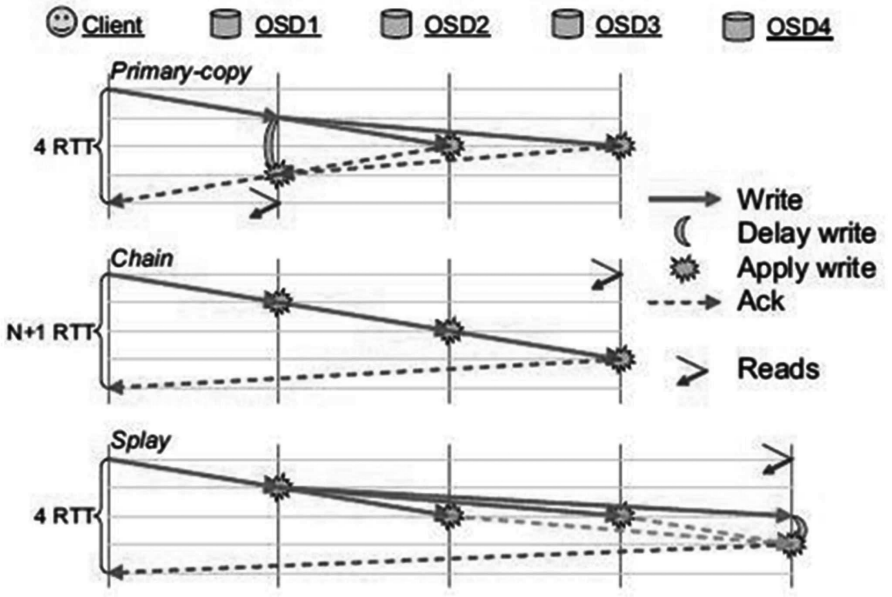


图17-2　分布式存储的读写：Ceph支持的三种读写的顺序图

MapReduce 是通过把对数据集的大规模操作分发给网络上的每个节点实现可靠性 ；每个节点会周期性地把完成的工作和状态的更新报告回来。如果一个节点保持沉默超过一个预设的时间间隔，主节点记录下这个节点状态为死亡，并把分配给这个节点的数据发到别的节点。每个操作使用命名文件的不可分割操作以确保不会发生并行线程间的冲突 ；当文件被改名的时候，系统可能会把他们复制到任务名以外的另一个名字上去。化简操作工作方式很类似，但是由于化简操作在并行能力较差，主节点会尽量把化简操作调度在一个节点上，或者离需要操作的数据尽可能近的节点上了。

17.1.3　数据存储设计

相较与传统数据库存储及文件存储，分布式存储引入了对象存储和 NoSQL 数据库，但都还远远没有成熟。相较与传统数据库的分割方式来支持大数据并发，分布式存储优势明显，但设计模式基本都在实践中，已经有不少的工程实现，但理论还远没成熟。这需要技术人员不断尝试、借鉴，建立适合自己需求的存储与数据系统。

应用针对存储设计，首先是数据形式，那些数据需要以文件方式保存、那些数据要以对象形式保持、那些数据要用数据库。一般来讲，音视频、文档及网页等用数据适合以文件形式保存在存储中，而应用中格式化数据适合以对象或数据库形式保存。对于对象实例较多，而且需要查询的，一般使用数据库方式存储。此外，分布式存储都提供数据域，区分数据域是用于将数据分割存储到不同物理区域。虽然数据可以动态分布到所有节点，但这样效率并不高，因此需要尽量将处理与数据统一到一个区域，提升数据访问效率。

NoSQL 设计实际是对象模型设计。同传统数据库 schema 设计不同，在使用 NoSQL 时应用程序需要设计自己对象模型树，针对对象实现流化，即以数据块方式存储数据对象。NoSQL 与传统数据库不同，每个对象的属性可以动态改变，而一个对象就是传统数据库的一个行，对象属性可以认为是数据库的列，而传统数据库的列是预先定义和不可改变的。NoSQL 数据库支持每一行数据自行定义自己的属性，这样对象设计自由度更大，可以完全按照程序逻辑设计自己数据对象和确定数据图。NoSQL 数据库都支持数据图关系，这样在获取对象时就能够获取到相关对象，从而减少查找，以提升应用效率。

由于分布式存储设计方法还不成熟，只能借鉴实践经验，下面是一些实践者的经验总结 ：

（1）eBay 架构经验

- Partition Everything 切分万物
- Asynchrony Everywhere 处处异步
- Automate Everything 全部自动
- Remember Everything Fails 记录失败
- Embrace Inconsistency 兼容不同是谓大同
- Expect Revolution 期待变革
- Dependencies Matter 重视依赖
- Be Authoritative 独断专行
- Never Enough Data 数据多多益善
- Custom Infrastructure 自定义基础设施

（2）淘宝架构经验

- 适当放弃一致性
- 备份和隔离解决稳定性问题
- 分割和异步解决性能问题（类似 eBay 的 Asynchrony Everywhere）
- 自动化降低人力成本（类似 eBay 的 Automate Everything）
- 产品化管理

（3）Flickr 架构经验

- 教会机器自动构建（Teach machines to build themselves）
- 教会机器自我监控（Teach machines to watch themselves）
- 教会机器自行修复（Teach machines to fix themselves）
- 通过流程减少 MTTR（Reduce MTTR by streamlining）

17.1.4　NoSQL的问题

目前，分布式存储，特别是在其上的 NoSQL 数据库，不支持数据关联性处理，数据关系要通过类似全文搜索来处理，因此对于结构数据，特别关系密切型数据处理，有很大限制。对于熟悉关系数据库处理的程序人员来讲，这是很大缺陷。

另一个方面，数据库拆分，目前也是用来解决大数据问题的常用用法，但对于线性增长的数据，数据库会一点一点变慢，如果分库数量过多，管理和重组数据开销一样会增多，系统性能还是逐步降低，对于每日上亿条数据入库，数据库分库也很难满足要求，而系统延展性会变差，而数据库管理会随着库数量上升而日益困难，应用开发难度增加。

为了克服这些困难，人们又提出了在分布式存储方案上实现分布式关系数据库的方式，比较典型的如基于 Hadoop 的 HadoopDB 方案，该方案通过使用 Hadoop 数据分布及 MapReduce 并行计算方案，将关系数据库数据分布存储到不同数据库上。在查询时，使用 MapReduce 并行计算，同时在多个并行数据库上执行，并将执行接口汇总到 MapReduce 任务，合成成一个完整的结果，图 17-3 是该方案的设计图 ：

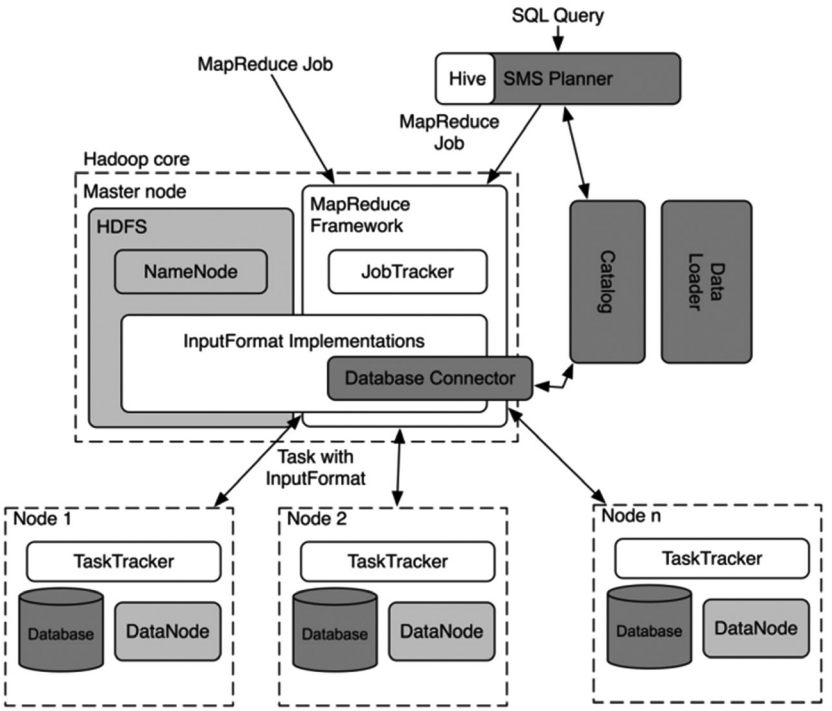


图17-3　分布式存储方案上实现分布式关系数据库的方式

该方式可以支持关系型数据库的数据库表关联查询，相较于 NoSQL 的类似全文查询，该方式要友好很多，性能也提升很多。对使用者来讲，由于该方式可以认为是分布式的关系型数据库，可以极大降低应用开发的难度与成本。

17.1.5　案例分析：存储设计

本节将以一个企业微博服务为例，围绕技术架构关注的几个方面进行详细分析。

（1）业务介绍

主要业务场景如下 ：

以在线服务的方式，为全国近 1,000 万家企业提供企业微博服务，包括企业内部沟通与信息发布、项目组沟通与信息发布、替换目前企业 IM 沟通等。企业平均规模在 50 人以上，支持项目讨论及基于文档讨论，文档需要一并存储。

- 包含在企业门户上提供入口，企业客户匿名登录同销售及售后服务人员的微博沟通。
- 含有企业内部人员结构、好友及常用通信录，支持项目组群，以方便组织内部沟通。
- 用户可以设定是否公开企业那些联系人或组，公开的联系人可以跨企业查询，其他的企业用户对企业内部可见，对外保密。
- 支持用户在组群中发布文档，并支持以附件方式发布或传递文档。

（2）业务分析

该企业微博服务要求支持 1,000 万家企业，包括企业内部用户及企业外部客户，而且是企业日常沟通使用，替代目前 IM 沟通，按照平均规模计算，要支持 50,000 万即 5 亿注册用户，按照平均每十人一天发布 1 条消息计算，每天有 5,000 万条消息，按照每 10,000 人一天发布一个文档计算，有 5 万个文档发布，用户信息和文档增长总量每天 50G 左右，按照企业需要，项目一般需要在线查询 1 年数据，离线查询 3 年数据，该服务存储总量在 18PB 左右，而结构化数据占 8PB，非结构化数据占 10PB。数据量巨大，使用 SQL 数据库需要按照企业划分，需要划分成太多块，非常难以管理，该规模数据适用于使用存储方式，用 Hadoop HDFS 及 Hbase 来实现该项目数据存储，能够 100PB 级数据，能够满足该需求。

HDFS 与 Hbase 数据库等的关系结构如下图 17-4 所示 ：

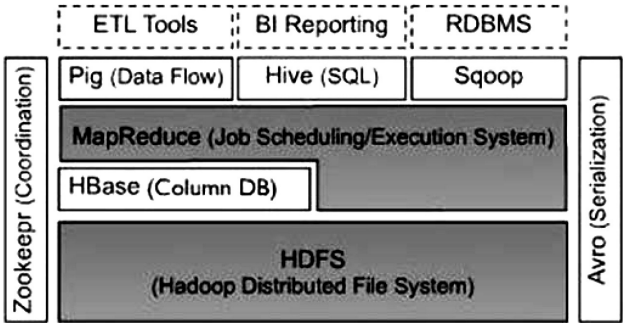


图17-4 Apache Hadoop项目的生态圈

其中 Hbase 位于结构化存储层，Hadoop HDFS 为 Hbase 提供了高可靠性的底层存储支持，Hadoop MapReduce 为 Hbase 提供了高性能的计算能力，ZooKeeper 为各个软件提供任务的协调和分发机制，如为 Hbase 提供了 failover 机制。

此外，Pig 和 Hive 还为 Hbase 提供了高层语言支持，使得在 Hbase 上进行数据统计处理变得非常简单。 Sqoop 则为 Hbase 提供了方便的 RDBMS 数据导入功能，使得传统数据库数据向 Hbase 中迁移变得非常方便。

在数据和系统保护上，使用的是 Kerberos 安全机制。Kerberos 是基于密钥认证的机理，来解决服务器到服务器的认证，以及 client 到服务器的认证。

（3）逻辑数据建模

图 17-5 是数据的逻辑模型 ：

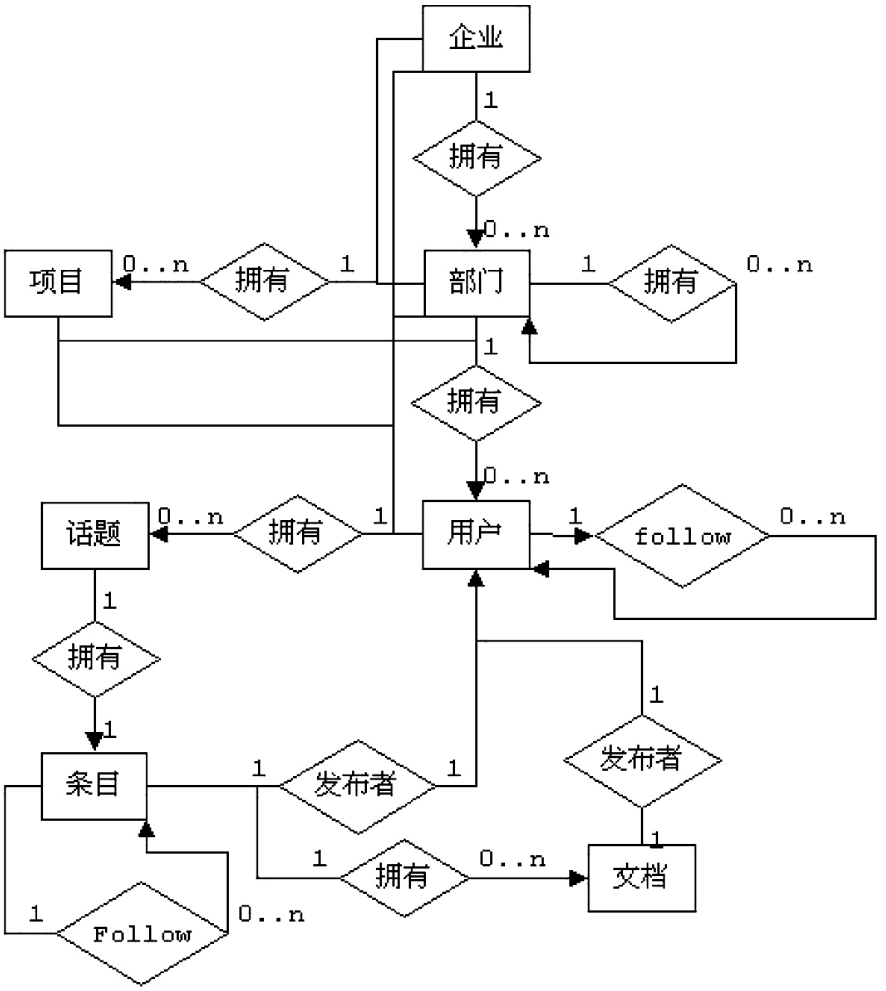


图17–5 数据的逻辑模型（案例）

（4）物理数据模型

Hbase 这样 NoSQL 数据库与 RDBMS 不同，它不支持 join。因此设计时要将关系作为行数据加入表结构中，支持关系搜索。为了提升性能，数据设计通过数据冗余提升搜索效率。

对于用户来讲，用户自身属性在一个列簇，包含用户的企业、部门等属性，之外，用户关注的用户以及用户被哪些用户所关注（following）关系作为数据库的一个列簇。在关注和被关注列表中包含姓名，这样在大多数场合，无需进行再次查找，能够正确显示在网页上。这些列簇中冗余数据是为了提升显示查找效率的，一般场合，在显示列表时，只需要姓名即可，这样就以数据冗余代替了关系数据的表 join，问题是数据一致性需要应用程序来保障，这点是 NoSQL 型数据库与关系型数据库极大不同。

此外，为了查询方便，将企业部门名作为数据直接放在该表中，在用户操作场合，使用该表支持用户数据，这里将一些不需要数据省略掉了。

Hbase 还不能支持两个以上列簇，在实际使用时，如果需要多个列簇，可以通过增加数据表方式来增加，相当于关系数据库的纵向切分（见表 17-1）。

表17-1　Hbase案例设计1：用户关系表

行	列簇						
	基本信息			联系人			
Key	姓名	部门	角色	Key	姓名	关系	
U_1	Aaaa	研发部	用户	U_2	Bbbb	双向关注	
				U_3	Cccc	关注者	
				U_4	Dddd	被关注者	
U_2	Bbbb	系统部	用户	U_1	Aaaa	双向关注	
U_3	Cccc	市场部	用户	U_1	Aaaa	被关注者	
U_4	Dddd	研发部	管理员	U_1	Aaaa	关注者	

在用户之外，企业关系数据也是一个重要部分，用于处理组用户权限查询，该表也面向企业管理使用，信息集中到一张表（见表 17-2）。

表17-2　Hbase案例设计2：部门关系表

行	列簇						
	基本信息			子部门		成员	
Key	名称	经理key	经理名字	Key	名称	Key	姓名
E_1	GE			D_2	研发部		
				D_3	系统部		
				D_4	市场部		
D_2	研发部	U_4	Dddd			U_1	Aaaa
						U_4	Dddd
D_3	系统部					U_2	Bbbb
D_4	市场部					U_3	Cccc

在企业管理之后，还有用户组合项目管理，对于该系统来看，两者区别很小，只是显示名称区别，因此可以放在一张表中（见表 17-3）：

表17-3 Hbase案例设计3：项目管理表

行	基本信息			成员	
Key	名称	管理者Key	管理者	Key	姓名
G_1	设计讨论组	U_1	Aaaa	U_1	Aaaa
				U_4	Dddd
P_1	项目1	U_2	Bbbb	U_2	Bbbb
				U_1	Aaaa
				U_3	Cccc
				U_4	Aaaa

其次是微博内容表，微博内容以及按照话题组织方式，注意数据冗余是为了加速显示（见表17-4）：

表17-4 Hbase案例设计4：微博内容表

行	列簇							
	内容				回复			
Key	内容	归属topic	创建者姓名	附件	Key	内容	附件	创建者
U_1_G_1_i_1	如何处理多键问题		Aaaa		U_4_G_1_i_2	要看具体场景		Dddd
					U_1_G_1_i_3	在内存map中	Multikeymap.cpp	Aaaa
U_4_G_1_i_2	要看具体场景	U_1_G_1_i_1	Dddd					
U_1_G_1_i_3	在内存map中	U_1_G_1_i_1	Aaaa	Multikeymap.cpp				

这样，对于用户初始化时，按照时间和 key（U_1_*_i_d*）以及 topic 不是（U_1_*）就能查到显示的所有类容，展现到网页，对于展开一个 topic，则使用 topic 的 key 查找，通过其内容，就能够展现整个页面。

这样数据冗余，对于显示显然是友好的，但对于数据插入显然是不利的，因此在程序设计时，要通过多次写操作，保持数据一致性。

此外，对于更复杂逻辑查询，需要引入 HIVE，使用 SQL 语句驱动 MapReduce，以获取最大性能。

（5）扩展性

对于 Hbase 及所有 NoSQL 数据库来讲，可扩展性一般通过分区来实现的，在 Hbase 中 region 是通过管理服务器设计实现的，一个 region 有一对 key（即起始和终了 key）实现的，对该应用来讲，自然分区最好以企业为单位，为了方便规矩定义，对上面 key 值做一个升级，增加一个 e_n_ 的前缀，同样在表中，企业列簇可以去掉。

用户键值是 e_2_u_1, 微博条目键值成为 e_2_u_1_i_1，这样就能很容易的将一个或多个企业配置到一个区域，通过硬件访问及前端访问负载配置，区域数据尽量在区域内访问，提升系统性能。

对于上传文件处理，每个企业可以建立一个文件目录，目录以企业的键值为目录名，将目录指定到不同区域，就能实现数据和文件一致分区，提升访问效率。

由于 Hbase 这样 NoSQL 数据库目标是面向巨大数据量访问，因此扩展相对简单，需要根据数据量及访问量，增加数据节点，即增加服务器数量，使用 MapReduce 提升性能，切分相对容易实现，无需额外程序开发。

（6）高可用度

由于分布式存储在存储底层采用集群及多备份方式存储数据，是一个高可用度集群，系统的可靠度比传统数据库有很大提升，因此应用软件无需担心。此外数据备份、包括灾备都能够在数据层处理，因此分布式数据系统本身就是一个高可用系统。

（7）性能

分布式文件系统性能相当于数据量较少的 ftp 访问，比本机访问性能性能要低不少，在大规模部署时，数据是分片并行处理的，这样使性能提升。因此在初期数据量较少时，适宜用关系数据库，在数据量增加到一定数量，使用 NoSQL 比较合适。

在部署时，数据节点，包含文件存储和结构数据存储，可以分布到 20 个节点上，随着用户增多，可以增加节点。region 备份已经存到物理节点上，因此也可以逐步建立。这样就能尽量降低初期投入，而在用户增加时扩展也相对容易。

17.2　云服务关系数据库工程

17.2.1　数据库架构设计的挑战

随着互联网的发展，应用软件已经发展到两层或多层架构。在过去 30 年里，涌现出了很多的程序语言、各种各样的架构和不同的理论，并且还在不断的向前发展。但数据库处于所有应用的最底层并为上层应用提供服务的情况却从来没有改变，数据库需要在同一时间为不同的用户和应用提供服务。许多用户和应用需要并发地修改数据库中同一数据，在这种情况下对数据库的设计便提出了更多的挑战。同时，不同数据库的架构差异也影响着数据库的设计。本章节将尝试通过通用的视角来描述数据库架构，对于具体数据库的细节问题则不作过多的涉及。

本小节通过以下几个方面来讨论数据库的设计。数据库这个领域非常的庞大，接下来的每一个主题，如果延伸下去都可以写成一本完整的书。所以，我们在这里的讨论只涵盖了原理性的东西。如果读者需要了解细节，则可以找相关的资料去做一些深入的研究。

17.2.2　数据库设计

对于很多开发人员来说，每当谈到数据库设计，大多数指的都是数据库模式（schema）的设计。模式设计包括两个方面 ：逻辑模式和物理模式设计。

- 逻辑数据建模

  逻辑数据建模指的是通过收集和分析业务需求并为它们建立数据模型的过程。这种方式更多关心的是业务流程而不是数据如何的存储。业务流程包括实体和实体之间的关系，数据流和业务处理的过程。逻辑建模的结果就是实体关系（ER）图，数据流图和流程

图。在项目实施之前，逻辑数据建模能够确保数据库的设计囊括了所有的业务需求。下图 17-6 是一个关于消费者、订单数据库的逻辑数据模型 ：

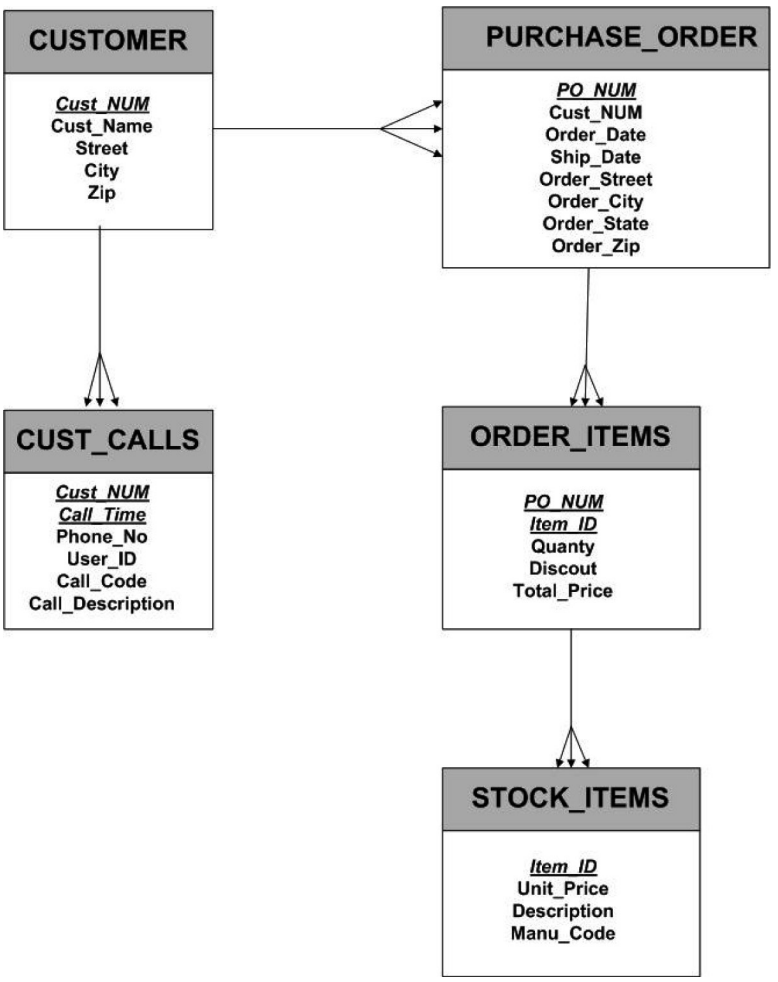


图17-6 逻辑数据模型（举例）

· 物理数据模型

逻辑数据建模可以转化成真实数据库里的相应关系，逻辑建模分别将实体或者说对象以及它们的属性对应到数据库里的表和列。数据本身的特性又决定了数据在数据库里存储的类型、精度、值的长度、数据是否可为空和数据是否可被查询。对于一些其他的业务需求，数据的属性和关系可以转化成数据库里的约束关系，比如主键 / 唯一性约束，外键约束和检查（check）约束等等。数据库视图类似于一种伪表，很多时候它能提高数据库的安全性、易用性以及数据库性能。大多数情况下，为数据库添加索引（indexes）能显著地提高数据库的性能。在数据库设计的过程中可以通过使用触发器（trigger）和存储过程（procedure）在数据库里定义一些业务规则。

上面的逻辑设计可以转化为如下的物理设计数据模型 ：

```
CREATE TABLE Customer (
  Cust_NUM     NUMBER NOT NULL,
  Cust_Name VARCHAR2 ( 200 ) NOT NULL,
  Street       VARCHAR2 ( 200 ) NOT NULL,
  City          VARCHAR2 ( 200 ) NOT NULL,
  State        CHAR ( 2 ) NOT NULL,
  Zip           VARCHAR2 ( 20 ) NOT NULL,
  CONSTRAINT Customer_PK PRIMARY KEY ( Cust_NUM )
  ) ;
CREATE TABLE Purchase_Order (
   PO_NUM        NUMBER,
   Cust_NUM      NUMBER,
   Order_Time    DATE,
   Ship_TIME     DATE,
   Order_Street     VARCHAR2 ( 200 ) ,
   Order_City       VARCHAR2 ( 200 ) ,
   Order_State      CHAR ( 2 ) ,
   Order_Zip        VARCHAR2 ( 20 ) ,
   CONSTRAINT Purchase_Order_PK PRIMARY KEY ( Purchase_Order_NUM ) ,
    CONSTRAINT  Purchase_Order_FK  FOREIGN  KEY  ( Cust_NUM ) REFERENCES
Customer ( Cust_NUM )
   ) ;
CREATE TABLE Order_Items (
  PO_NUM NUMBER REFERENCES PurchaseOrder,
  Item_ID              NUMBER REFERENCES Stock,
  Quantity             NUMBER,
  Discount             NUMBER,
  Total_Price          NUMBER,
  CONSTRAINT Order_Items_PK PRIMARY KEY ( PO_NUM, Stock_ID ) ,
   CONSTRAINT  Order_Items_FK1  FOREIGN  KEY  ( PO_NUM ) REFERENCES
Purchase_Order ( PO_NUM )
  ) ;
CREATE TABLE Cust_Calls (
  Cust_NUM     NUMBER REFERENCES Customer,
  Call_Time    Timestamp,
  Phone_No    VARCHAR2 ( 20 ) ,
  User_ID      VARCHAR2 ( 20 ) ,
  Call_Code    NUMBER,
  Call_Description  VARCHAR2 ( 512 ) ,
  CONSTRAINT Cust_Calls_PK PRIMARY KEY ( Cust_NUM, Call_Time ) ,
```

```
    CONSTRAINT Cust_Calls_FK FOREIGN KEY (Cust_NUM) REFERENCES
Customer(Cust_NUM)
  );
  CREATE TABLE Items(
    Item_ID NUMBER PRIMARY KEY,
    Unite_Price      NUMBER,
    Manu_Code NUMBER,
    Description  VARCHAR2(512),
    CONSTRAINT Items_PK PRIMARY KEY(Item_ID)
    );
```

以下是物理设计模式的规范

- 对于联机事务处理的应用（OLTP），大表应该拆分成多个小表。
- 对于一对一的实体关系可以做相应的合并。
- 保证对象命名的标准化和规范化（比如对象名的缩写等）。
- 按照规范化模式（schema）和非规范化模式设计方式无法在数据的完整性和冗余性上去求得平衡。
- 确保主键（primary key）和唯一键（unique keys）永久唯一性。
- 需要考虑到未来 ID 的可重用性。

17.2.3　数据库的可扩展性

什么叫数据库的可扩展性？数据库的可扩展性指的就是当我们花费双倍投入的时候，能够从数据库系统获得相应预期倍数的数据处理能力。事实上，作为应用的底层，数据库必须保证持续地为上层应用和用户提供服务的能力，这就使得数据库的伸缩性变得非常重要和极富挑战性。和应用程序不同的是数据库对外共享所存储的绝大多数的数据，这就使得数据库容易产生热点块（hot spots）以及导致相应性能瓶颈。尽管在过去的 20 年里，人们一直面临类似这样的挑战，但如今许多新技术和理论的出现使得对数据库的扩展不再只是一个神话。

数据库的扩展可以分为垂直扩展和水平扩展两种。

- 垂直扩展（vertical scale-up）
  - 垂直扩展是增加所用的某一个或几个数据库本身的硬件或软件性能以增加其数据库的处理能力。
  - 垂直扩展是传统意义上提高数据库的伸缩性的一种方式。相当多的大型企业如今都将数据库跑在昂贵的大型机上，对于垂直扩展这样的作法依然会受到限制。未来一旦大型服务器的资源被耗尽，如何扩展也变得不那么简单。虽然这种情况近些年来很少出现，但是如果预算充足的话，考虑到数据库的性能和可扩展性还是建议采购更多高性能的服务器，并且配备大容量的内存。垂直扩展包括扩展数据库系统的所有组件（component），如 ：数据库的存储容量，内存（以提高数据库的响应速度和支持更

多用户的并发访问），更多性能强劲的 CPU 等。

- 水平扩展（horizontal scale - up）
  - 水平扩展是增加所使用的数据库数目，让多个数据库为同样的应用服务。
  - 随着软件和硬件对并发访问支持的出现，很多数据库都采用水平扩展方式。水平扩展通常只需要普通的硬件设备，因此，水平扩展的花费比垂直扩展更加便宜。普通服务器的发展使得它们在处理企业应用和负载均衡方面的能力比以前更加的强大。

一般情况下，数据库系统的扩展只需要简单地通过添加更多的服务器来实现。在过去的 30 ~ 40 年里，有许多研究报告都对数据库服务器扩展的不同方式进行了论述，本章不打算对每种扩展方式的相关细节进行讨论。这里只列出各种扩展方式及相应的主流的数据库解决方案。

- 共享内存（shared memorg）：多 CPU 访问同样的共享内存池，这种方式多数出现在大型机上，DB2 数据库主要应用于该方面。
- 共享存储（shared disk）：在一个数据库集群里大多数的服务器都配置了独立的 CPU（可以是多核），内存和本地磁盘（主要用于数据库的二进制文件日志文件）。数据库文件（data files）则存储在 SAN（Storage Area Network）或裸设备（raw device）上 。所有的数据库实例则通过高速直连的方式访问共享存储上的数据。相对大型机来说即使使用普通高端机做数据库服务器依然显得很便宜。作为数据库最大提供商的 Oracle 公司，其 Oracle RAC 使用的就是共享存储的架构。
- 非共享（shared nothing）：每台数据库服务器对其他服务器不共享任何数据，每台服务器都有自己独立的存储、内存和 CPU，数据通过复制传输到其他服务器。复制这种方式比较简单直接并且适合于特定业务需求的数据库。很多数据库服务提供商都有采用了非共享的架构设计，比如 IBM 的 DB2，UDB，微软的 SQL Server，MySQL，Teradata DB，Informix 的 XPS ，Sybase 的 Navigation 等。

17.2.4　数据库的高可用性

正如之前所说，数据库处于所有应用的底层并为应用提供服务。任何数据库的中断将导致整个应用的停止。因此，保证数据库的持续运行就变得至关重要。

下面是两种不同的数据库高可用（HA，high availability）方案 ：

- Active - Standby 模式 ：一个数据库服务器（或集群）响应请求，其他数据库服务器作为备用服务器。备用服务器可以应用来自主服务器的所有更新操作。一旦主服务器不可用，备用服务器将转换为主服务器并响应所有的应用请求。
- Active - Active 模式 ：所有的数据库都是活着的（即都响应应用的请求），所有的数据库都承担负载。在数据库的上层需要使用负载策略用以均衡分散应用的请求。这种模式能使服务器更高效地工作，而不至于使一些服务器闲置而导致资源浪费（Active - Standby 模式下备用服务器一般不对外提供服务，故叫闲置）。

与数据高可用性相关的设计包括 ：

- 数据库监控

  数据库监控主要有两个目的。第一，确保负载均衡器、应用或数据库集群能够及时发布通知，以完成一些自我调整的操作（比如 Oracle RAC 集群的内部监控可以监控各个节点的健康状况，并且在出现问题的时候主动踢出有问题的节点）。第二，向数据中心的团队人员通知事故的发生（有些事故在用户的角度来说是感受不到的），类似这样的故障往往会导致服务级别的下降和服务器处于高负载状态。如果这种状况一直持续下去将会导致更严重的后果。

- 自动故障转移

  故障转移可以发生在不同的应用级别。负载均衡器可以将出故障的数据库标记为不可用并且将应用请求直接指向存活的数据库服务器。数据库集群故障转移则是将故障节点上的 VIP（virtual IP）漂移到活着的节点上。连接池管理器主动释放对故障节点的连接并且从新与活着的节点（live db server）建立连接。对于 Oracle RAC 集群服务（ORS），它能发现出现故障的数据库节点并将情况通知给其他节点，同时将故障节点上的 VIP 转移到其他活着的某个节点上，最后将情况告诉监听服务和 Oracle 应用服务（OC4J），连接池管理器会释放失效的连接并与新的活着的节点重新建立连接。

高可用数据库（HA DB）可以在数据库软件层实现，如 Oracle 的高可用集群数据库服务方案 ；也可以在系统层的集群（cluster）方案来实现，如 Veritas 集群方案和 HP 集群方案。

17.2.5　数据库的备份和恢复

所有数据库都需要有备份恢复策略，就像墨菲法则（Murphy's Laws）说的那样 “ 任何事情都可能出错并因此导致更多的错误 ”。这也告诉我们，随时需要做好应对故障转移和数据库异常的准备。

数据库备份通常用于数据库 crash 和数据不一致后的恢复。在少数情况下，数据库需要恢复到错误操作或应用出错的时间点。所有这些情况都说明了数据库备份的必要性。备份数据库通常在数据库服务器负载很低的时候进行（比如凌晨）。数据库有两种备份模式 ：增量备份和完全备份（有时也叫全量备份）。增量备份通常需要耗费时间和系统资源相对都比较少，因为它备份的是自前一次备份以来的所有对数据库的更改，但是恢复的时间将更长，因为在恢复的时候它需要读取或扫描更多的备份集。相对增量备份来说，完全备份需要消耗更多的系统资源和花费更长的时间，但是在恢复的时候则要更快。

在制定备份策略的时候，必须保留最近几天的备份集以备数据库进行基于时间的恢复。有时候备份需要同时存储在本地和远程数据中心（地理位置上不同），将备份数据存放在远程数据中心可以在发生不可抗拒的灾难事故时也可及时进行恢复。

在制定备份策略的时候，必须考虑到业务需求的因素，否则备份可能起不到应有的作用。

17.2.6　数据库性能优化

关于数据库性能优化这个话题如果要详细论述的话，完全可以写成好几本书。因此，本小节尝试从更高层次（即非细节方面）来描述数据库的优化问题。多数情况下，性能优化针对的都是某特定的数据库（因为不同的数据库优化方式通常不一样）。对于从事数据库工作的人员来说，花更多的时间去研究优化是非常必要的。建议读者可以通过尽可能接近真实环境的实验室环境下去做一些测试。一般的数据库都提供了现成的工具来检测消耗时间最大的查询语句或执行频率最高的查询语句。这样的分析往往可以检测出可能导致数据库性能瓶颈的潜在问题。

以下是数据库优化的一些方法和技巧 ：

（1）数据库层面的优化

- 索引 ：一个高效的索引可以显著地提高数据库的查询操作（这里的查询包括 select、update）性能。一个结构合理的索引可以使查询避免做全表扫描。有时候索引能极大提高大数据量下的排序性能。创建什么样的索引和在哪些列上创建索引往往取决于那些查询最为频繁的列。通常情况下，经验丰富的 DBA 能够通过分析查询语句或模式（schema）的定义来确定是否需要或如何创建合适的索引。而在一些更加复杂的情况下则需要收集查询语句详细的执行计划和相关表的统计信息来最终决定如何创建更高效的索引。当然，值得注意的是索引也不是在任何情况都能够提高性能，比如在有索引的情况下则更新和插入操作往往更慢。另外，有时候全表扫描可能要比通过索引查询速度更快。

  索引可以是单列索引也可以是复合（多列）索引。

  - 糟糕的表结构设计往往会给插入操作带来很大的挑战（即写入数据很慢）。在那种情况下，标准化的数据库设计往往显得格外地重要。
  - 数据分区 ：数据分区是一项将大量数据拆分成许多小子集，然后再提供查询的一种数据库的优化技术。同时，分区技术也能给数据的组织和管理带来方便。
  - 数据库参数的调整 ：调整一些数据库的参数可以提高数据库的性能。比如缓存（buffer）池的大小，共享内存的大小和高速缓（caches）。
  - 绑定变量 ：在许多情况下绑定变量用来替换字面值。
  - 通过在查询语句里使用 where 条件来减少结果集的大小。
  - 在很多表连接和关键字检索的查询语句中，适当地使用参照完整性和主键 / 唯一键能相应提高性能。

（2）应用层的优化

- 使用存储过程替代 SQL 语句可以显著地减少网络资源的消耗。存储过程的名字（这里指名字的文本长度，数据库调用存储过程的时候是通过存储过程的名字来调用的）比起冗长复杂的查询语句文本，相对来说要简单很多。
- 尽量避免在频繁访问的数据上产生热点块。解决的方法是增加磁盘分区并将数据尽量放在不同的分区上。如果热点块不可避免，则可以考虑通过数据分区或集群亦或是数

据复制来解决。

- 尽量在应用层缓存数据，这样能减少数据库服务器的负载并且应用访问数据也更加方便。一些数据甚至可以考虑通过消息服务（messaging service）在不同应用间进行共享。

17.2.7　数据库安全

在商业运营上，保障 IT 系统中客户数据的隐私性和安全性至关重要。客户重要数据的泄露不仅是服务提供商的一件公共关系事件，更严重的是在商务上还需要承担相应的法律责任。值得庆幸的是如今在数据库里有很多技术来保障数据的安全。

以下是在数据库层面上提高数据安全性的一些方法 ：

- 为不同的用户授予不同的权限（即严格控制对用户的授权）。
- 确保只有认证用户才能访问数据库系统并且只能访问其被授权访问的数据。
- 应用层面认证 ：在应用层面上可以使用基于数据记录的授权机制和基于业务逻辑的认证方法，如 Oracle 提供了可自动在 where 子句里添加私有数据库谓词和用户认证功能的功能。
- 不要将密码存储在可以通过 http 访问的脚本（比如 jsp）里。建议仅仅将密码保存在配置文件里。
- 使用数据库审计以跟踪对数据库所有查询、更新和删除操作。
- 数据加密 ：防止用户和用户密码被盗，用户可以使用现有的加密程序或用户自定义的加密程序并将其部署到应用里面。加密程序可以在数据插入数据库的时候对数据进行加密并且在取出数据的时候进行解密。这一般用于敏感数据如密码等数据保存。
- 防止 SQL 注入。

什么是 SQL 注入？举个例子，这里有一个 WEB 应用提供学生查询所有其已经注册了的课程。WEB 交互界面要求用户输入 学生 ID，通过用户输入的数据组装成 SQL 查询语句 如 “ select * from registerd_courses where student_id=100 ”。这时，如果一个不怀好意的用户输入一些其他数据如 ：'100;truncate table registered_courses'，加之 SQL 语句写得不够严谨（如对输入的数值不经过校验），这种情况将对数据造成严重威胁。

下面是一些防止 SQL 注入攻击的方法 ：

- 应用应该检查所有用户输入的值。应用应该在设计的时候加入对输入数据的检查逻辑，在上面的例子里，可以检查数据的长度或是检查否允许含有特殊字符以确保学生 ID 值的可用性。
- 在给用户执行 SQL 查询的权限时需要格外的小心，确保授予用户最小的权限（即如果是查询权限那么只给 select 权限，其他的都回收），如果上面例子里用户没有写权限那么就无法成功执行截断（truncate）表的操作，也就避免了对数据库的威胁。
- 使用预定义语句取代 SQL 查询字符串的拼接。
- 使用存储过程而不是直接使用 SQL 语句。
- 使用函数过滤 SQL 查询字符串里的特殊字符（如“ 号）。

在使用不同数据库时，数据库本身还会有各自一些保障数据安全的策略，如 Oracle 允许用

户使用自定义的认证策略，即使用户能登陆到数据库，它也必须提供由 DBA 随机生成的预定义变量清单信息，否则的话它依然看不到任何数据库里的数据。此外，Oracle 还提供了和 PL/SQL 存储过程类似的加密程序，该程序包具有一个关键字（key）该 key 被存储在数据库或被编译到应用程序里。这样即使入侵者侵入了数据库，但依然看不到数据库里的数据真实值。

要有效利用数据库提供的各种安全机制以保障数据安全。常用数据库都支持数据项或单元加密。对于安全性要求高的数据，采用程序加密是防止泄密的另外一个重要手段。

17.3　进一步讨论

由于云服务的多租户、大用户量的特点，海量数据存储与管理是一个重要技术课题。关系数据库是以保障强一致性为目标，它对数据分布式处理有局限性，没法支持海量数据。如果通过应用分库处理，当数据量过大时，效率及应用复杂度是个很大问题。

为了解决这些问题，目前的 NoSQL 数据库，牺牲了强一致性，换取分布式数据存储，并以无连接方式访问数据。NoSQL 这样分布式、多备份、多点读取方式，极大提升数据库容量、并发性能及可靠性，以最终一致性替代关系数据库的强一致性，以数据冗余替代关系数据库的联合查询，最终提升数据容量及数据可用性。目前 Google、Amazon、淘宝、eBay 等都各自根据自己实践，开发了各自的数据存储及数据库，在云服务数据平台支持中走在前列。

而目前国内，由于对 NoSQL 计算跟进不够，大多数企业仍然使用关系数据库来实现云服务。对于刚刚起步的新业务，关系数据库方法是一个不错的选择，但随着数据积累，数据的存取和备份，都会成为服务性能的瓶颈。在这时，服务数据构架就应该向分布式存储转移，以提高服务性能。

云服务数据存储是云服务能够成功的关键技术之一。技术本身需要长期投入的技术，同时，分布式存储和 NoSQL 也远没有成熟，没有统一标准，工具与开发手段也没有成型，这些也是数据服务企业的机会，而大数据时代一定会是在这些基础技术成熟基础上才能够抵达高潮。在这方面投入与成功，必然能够产生一批下个互联网时代的尖端企业。

第18章 云计算的服务： 面向服务的架构

随着互联网日益成熟，特别是接入技术，由最初电话线和 ISDN 接入，到 ADSL、Ethernet 网线等宽带技术，再到光纤、WLAN、3G、4G/LTE 等最新接入技术成熟，人们可以随时随地接入到互联网，享受互联网提供服务。人们也习惯于在互联网上建立自己的工作或生活的圈子、通信、购物等，日常活动越来越依赖于互联网，而随着云计算的进展，人们可以将自己出游的视频、照片直接存在云存储中，随时可以浏览、转发、共享，同自己认识或不认识的人分享自己的体验。同时企业也开始转变，IT 不再重要，IT 服务可以通过互联网获得，不再需要购买大量机器和软件，不需要投入大量现金、人力来维护自己的基础设施，而是通过租用云服务，降低成本，提升自己竞争力。这样云计算服务越来越普遍，最后必将像电、水等消费品一样，只要接入，支付租金，便能享受到各种需要的服务。

对于云服务而言，一旦被用户认可，用户访问量和数据量都会呈现爆发性的增长，这就对技术架构提出了更高的要求。实践证明，除了功能需求外，云服务架构必须考虑以下因素 ：

- 高可用 ：7x24 小时不间断服务，在发生软件或硬件故障时，也能提供稳定的服务。
- 高配置性 ：通过配置实现客户个性化需求。
- 高性能 ：高效的处理业务请求，提供优质的用户体验。
- 可伸缩 ：负载增加时，能方便扩展以满足需求，且不降低服务质量。
- 可维护 ：提供方便快捷的管理和技术运营途径。
- 服务支持 ：能够支持服务团队，支持客户使用。

本章将从这些基本需求出发，分析云服务平台的技术架构，并结合实例分析，讨论构架云计算服务平台的实践。

18.1 7x24的云计算服务的挑战

云服务有它独特的优势，与传统软件相比，它由专业技术、服务人员集中提供服务，在保障服务质量同时，由于其专业和规模效益，成本更低，使用更加方便 ；但也正是由于这一特点，使得构架云服务软件与传统软件有着本质区别。

18.1.1 比较传统企业服务软件与云服务软件

传统 IT 服务软件，其对象是一家企业，使用用户是一家企业的雇员，有十万以上员工的企业就很少见了，而云服务，目标是服务于以万计的企业，用户量以千万计，如腾讯、360 等成功的云服务提供者，用户数量是以亿计，因此大用户量，高并发是云服务软件要考虑的首要需求。

大量用户同时在线对云服务的稳定性提出了非常高的要求。对于一个企业内部软件，中断服务影响可以预估，而对云服务，中断服务的影响是数以万计的企业，其影响范围扩大、处理难度都极大提高，这就对可靠性提出了更高的要求，需要系统达到 99.99% 或更高的可用度。而目前，互联网免费网站的稳定性一般低于 99.9%，如果对于 B2B 的收费服务，则必须达到 99.99% 以上。表 18 - 1 是服务的可用性指标 ：

表18-1　可用性指标

可用性指标	一月不可用时间
99.999%	<1min
99.99%	<5min

多租户的特点，也决定了云服务软件要具有与传统软件比较的另一个不同的特点，即功能、流程的高度可配置性 ；传统 IT 服务软件，软件为特定客户部署，可以根据客户需求配置或二次开发，以满足不同客户的特定需求，而云服务软件集中部署，同时供许多客户使用，系统配置不再是静态的，而要根据不同用户，选择不同配置，系统功能要方便地根据用户需要配置，同时满足数以万计客户同时使用。

此外，云服务用户是初步累积起来的，这就要求随着用户增加，系统能够通过增加部署，提升系统容量，传统 IT 服务软件也有当企业规模扩大，通过部署提升容量的需求，但一般通过系统升级来做到，而云服务系统，则要求更高，首先是系统扩张的频度比企业 IT 服务软件要高出很多，其次是数据量随着企业增加，呈现爆炸性提升，因此如果不在构架时考虑扩张，随着系统用户增加，并发压力升高，必将导致灾难性结果，因此如何构架高伸缩性也是云服务构架一个关键点。

互联网云服务，许多企业也许是相互竞争的对手，共享一个服务，数据都存在服务提供者处，用户通过互联网访问，这与传统软件数据存在企业内部，访问的仅仅是内网，相比较安全要求极大提高，如何保障用户数据安全，保证只有授权用户可以访问相应数据，是互联网云服务成败的关键条件之一。

与传统软件相比，云服务的最大特点就是面向多客户，与传统软件比较，简单总结如下（见表 18-2）：

表18-2　云服务与传统软件比较

	传统软件	云服务
容量	较低	非常高
可靠性	一般	非常高
可配置性	一般	非常高
伸缩性	一般	非常高
数据安全和保密性	分行业	非常高，不同租户之间不可见
技术运营难度	一般	非常高

18.1.2　特性化与统一服务

云服务是在互联网面向不同用户提供服务，是通过一个统一服务软件系统，服务于大量客户，而每个客户对服务有着不同的要求，这要求在统一服务的前提下，需要支持用户个性化需求，同传统软件比较，软件使用于一家企业，在部署是配置功能和流程，满足该企业需求即可，这些配置是系统级的、静态的。

云服务对于不同客户、用户，配置各不相同，客户、用户可以根据自己喜好，配置服务，这些配置往往是每个客户、用户有不同要求，在该用户使用过程中有效，而且随时可以更改，因此配置是分级的、动态的。

云服务配置分层为系统级配置、客户级（企业级）、用户级配置，相较传统软件复杂度明显不同。

相较于传统 IT 软件，企业有自己的 IT 人员服务、支持单个部署运行，而云服务系统是集中部署，服务于数以万计的企业客户，因此对于服务支持系统要求较高，如果是简单将各个企业 IT 人员统一集中，显然不能为服务企业提供更专业而且低成本的服务，因此在设计系统时，同时要考虑到运营服务支持的需求。

18.1.3　面向运营及服务系统功能

云服务要求的是电信运营级（carrier grade）的服务，因此运营维护和服务需求是软件基础需求的一部分。相较于传统软件，要设计专门的服务、技术运营系统，有专门运营、服务团队，通过软件提供的技术运营、服务功能，快速响应用户请求。这与传统软件有着不同要求，传统软件用户维护，通过提供系统诊断数据，用户遇到问题时先是自己 IT 人员自己解决，然后是请求支持，因此服务一般面对的是 IT 人员，属于专业人员，云服务的服务团队直接面对的是用户，一般不是专业人员，对问题描述是表象性的，要尽快定位，需要更多数据和证据，以提升服务、技术运营团队的效率。

此外，技术运营系统还要面对客户，如前面提到的，一家服务提供者整合云服务面对最终客户，在服务过程中出现问题，其服务人员要有方法定位问题，需要整合服务和维护到它们的服务系统。除此之外，云服务的 PaaS 和 IaaS 服务，当用户部署服务后，通过服务平台对其部署服务进行维护和问题分析，也是必不可少的功能。

18.1.4　传统IT管理与服务监控

云服务同传统软件一样，需要 IT 管理与监控系统，但其要求比传统软件要高许多，主要是用户量和接入复杂，在判断问题时因素更多，因此监控本身就是一个大系统，数据海量，分析复杂，而且用户访问行为记录不仅仅对技术运营判断问题有用，对于分析用户行为，改进系统也是非常重要的。

快速问题处理与错误恢复能力的要求，特别是对错误恢复时间要求更高，这需要在设计时考虑系统失败环节恢复方式及对技术运营执行工作的支持。

18.2　云服务构架

云计算软件构如同大多数软件构架一样，是对实际需求、扩展性、可靠性及性能等需求方面因素，专业因素如 CRM、物流管理、财务管理等，还有人力成本、开发周期、运营预期及升级周期等重多因素平衡取舍的结果。我们在第 8 章“ 云服务的生产设计 ”中谈到了云服务的高可用性、管理性、扩展性等设计理念和规范，这里主要是描述这些理念在构架设计中的技术实施模式。

18.2.1　设计的基础模式

（1）一致而稳定的抽象层

如前面论述，云计算是通过规模集中及服务专业化来提高服务效率，达到降低使用者成本目标，因此一般来讲，对于云服务系统，需要整合多个不同业务，以满足用户多方面需求。多个服务是否能够整合，而且是否能够根据用户需求，更换不同服务提供者，关键的一点是该服务能否抽象成一个高级别的组件，迅速组合，构成满足用户需求的服务，同时也为系统高度可配置奠定基础，以适应在一个平台上快速满足众多企业用户需求。

据此，云计算提高了抽象水平，所有组件都抽象化或虚拟化，并可用来迅速组合较高级别的应用程序或平台。如果某个组件不向其客户或同行提供一致而稳定的抽象层，该组件就不适合于云计算。常见如单点登录，将用户认证、用户关系组织成一个公有服务，用户在互联网上登录时，无论在哪个服务上登录，都能够提供身份，保障一次性登录使用所有应用，同样的，目前社交网络服务（SNS）中，使用人人网或 Linkedin 网的 ID 进行身份认证，将服务提供给更多用户，这是一个典型的用户认证及关系抽象的例子。对于该服务自身来讲，并不处理用户登录，仅仅通过虚拟登录概念，通过不同适配，达到同样目标，但极大扩展了服务使用人群。

云计算 SaaS 服务，大都能够采用 WEB 的服务封装，提供标准服务接口，这样各个客户及其他服务提供者能够简单集成，一起向最终用户提供服务，如上面所说的人人网及 LinkedIn 的服务接口，这样既能保障客户通过人人网或 LinkedIn 获取更多服务，也能够让网站在开通初期，尽快扩大用户群体，是一个双赢局面。

IaaS 应用标准部署单位是虚拟机，它本质上可运行于抽象硬件平台。人们很容易过度关注构建虚拟机映像（image），而忽视用来创建虚拟机映像的模式。在云计算中，维持该模式而非映像本身非常重要。该模式是保留下来的，而映像则是从该模式产生的。虚拟机映像将始终在变化，因为虚拟机映像内的软件层将总是需要修补、升级或重新配置。不变的是创建虚拟机映像的流程，而且这是开发人员所应重视的。开发人员可以通过把 WEB 服务器、应用程序服务器和 MySQL 数据库服务器层叠在一个操作系统映像上，应用补丁程序、配置更改，以及互连各层组件，来构建虚拟机映像。可以说虚拟机抽象是 IaaS 服务基础，但本质是虚拟机维护、创建、监控、资源配置与隔离及可靠性保障等抽象与封装，使得 IaaS 能够一致稳定地向不同客户提供服务。

PaaS 平台构建可以建立在 IaaS 之上。使用虚拟机系统隔离每个用户部署在平台上应用，虚拟机产生映像时，将平台 API 相关调用库，认证设置等一次部署到位，而且平台安全、可靠等相关部署事项都可以通过虚拟机方式隔离，让用户开发、部署应用的难度极大降低，这是 PaaS 服务抽象的常用模式。同样，通过抽象建立业务运行的隔离环境，除业务 API 外，部署、开通、计费等服务系统必备 API 都应该是 PaaS 平台 API 的一部分，这样就能够通过一个稳定抽象层，隔离各个用户发布的服务，当一个服务故障时，不影响其他服务正常使用。

（2）标准化

对于云服务，需要同其他服务提供商、企业服务集成，才能满足企业用户的不同需求，而不同企业用户对不同服务有着各自不同需求及选择，这就要求对服务的抽象和封装尽量满足行业标准，这样就能快速替换不同服务，满足用户需求。

云计算首先重视效率，因而采用少数标准和标准配置有助于降低维护和部署成本。拥有可简化部署的标准比拥有用于作业的最佳环境更重要。80/20 规则就在这里发挥作用 ：20% 的标准可

以支持 80% 使用案例。从成本效益来看，这样的工作可以让成本高的一次性产品开发转变为可以复用的构件开发，从而降低多个项目的综合成本。

云服务一般采用 WEB Service 标准接口，包含 RESTful 和 SOAP 两大阵营，随着封装级别提高，RESTful 日益兴盛。SOAP 由于历史原因，还有大量服务和系统支持，并且由于 SOAP 工具更加成熟，它仍然是许多服务提供接口的选择。考虑到效率、耦合度及开发分发等一系列因素，RESTful 可能是大并发系统的最佳选择。

18.2.2　设计的结构模式

（1）虚拟与封装

对于云计算服务，将服务接口封装成一个虚拟服务，在创建自己服务时，使用这些虚拟服务，在虚拟服务实现层，使用 proxy 模型，封装多个同一服务，在配置中体现使用不同服务，这样就是在增加更多的应用服务时，最多完成封装适配工作，这样能最大限度减少不同服务集成带来的影响。

虚拟和封装技术将实现细节隐藏起来，并使开发人员重新重视组件之间的接口和互动。这些组件应该提供标准接口，以便于开发人员方便快捷地构建应用程序，同时利用与性能或成本所要求的相似的功能来使用替代组件。

云服务开发是有计划地完成的，甚至用来部署服务的程序也要封装，以便于利用和重新利用。可以封装部署三层式 WEB 基础设施的程序，这样，该程序的参数就会包括指向用于 WEB 服务器、业务逻辑和数据库层的虚拟机映像的指针。然后就可以执行此设计模式，以便于部署标准应用程序，而不必重新设想或甚至重新考虑支持每层所要求的架构。

已经实践了多年的模型总线其实就是一种高层次的虚拟与封装过程。如一个有限状态机模型，是对很多有状态系统状态迁徙与行动的封装，通过将状态行为数据化，能够在多个不同部署上，同时对同一业务的不同过程进行处理，这样能极大提升系统并发能力。同时在处理过程中，由于一个服务可能会使用多个模型，模型总线概念是将这些模型及之间通信封装成一个松散的、以数据驱动为主的抽象实现，降低在其上实现、部署服务的难度。

（2）松散耦合、无状态、原地失败

松散耦合（loose coupling）、无状态（stateless）、原地失败（fail-in-place）的模式的根本动机是降低耦合。这种耦合是包含程序信令处理的逻辑耦合、状态依赖和计算上下文依赖关系等。当各个部分之间采用松散逻辑耦合、无状态依赖和无上下文依赖这种方式后，每个分片处理都能并发在不同机器，由于没有状态和上下文，哪个计算单元处理结果都是相同的，就能够避免因大量用户并发与分布处理导致的状态不一致等错误。同时，每个分片错误发生后可以直接重新开始也可以在原地处理。这样为提升系统可靠性、可管理性及大并发提供依据。目前，基于 WEB 的应用程序已在向松散耦合和无状态转变。

在云计算中，这些特征更加重要，因为云计算具有更加动态的性质，服务组件越来越动态，而且可能是分布在不同地域，任何组件发生故障都不会影响整个应用程序的可用性，一个组件应该做到“原地故障”，极少甚至不会对服务可用性产生影响，这既方便系统伸缩，也提升系统可靠性。

由于服务组件越来越具有临时性，应通过将状态推出软件来尽可能使服务运行处理无状态，从而尽可能地将处理与数据分开。能够做到这一点的技术包括 ：

- 将状态下推至后端数据库。
- 维护数据的补充副本，Hadoop 就使用这样的一个策略。
- 以 cookie 的形式将状态推出到用户，或者将状态代码编入 URL。
- 使用基于网络的持久性技术，例如，GlassFish 应用程序服务器的 Terracotta 或 Shoal。

“ 原地失败 ” 计算对操作活动的影响是，即使是硬件也应该是无状态的，以便于云正常工作。硬件配置应存储在元数据中，这样，在发生故障时就能够恢复配置。

（3）水平扩展

云服务用户往往会爆炸性增长，服务在设计和部署时，一定要考虑可伸缩性，当用户数量增加时，能迅速部署，扩展系统容量，同时能够保障系统性能，支持用户快速扩张。水平扩展模式是支持系统提升容量的重要模型。设计服务以在水平扩展环境中顺利工作的趋势，意味着越来越多的服务能够很好地适应云服务方式。

利用水平扩展的服务重视假如个别组件发生故障整个服务的可用性，如 IaaS 云服务平台是在一个虚拟资源池上构建的，其中，如果任何一个物理服务器发生故障，该服务器托管的虚拟机只需要在一个不同的物理服务器上进行重构。把无状态和松散耦合的应用程序组件与水平扩展技术结合在一起，可促成一种 “ 原地失败 ” 策略，这种策略与任何一个组件的可靠性都没有关系。

水平扩展不必局限于单个云服务。根据服务数据的大小和位置，“ 超负荷 ” 计算可用来扩展一个云服务的能力，以适应临时性工作负荷增加。在超负荷计算中，云服务的应用程序可能会根据需要吸纳来自公用云的额外资源，如使用人人网登录，将用户信息暂时存储在人人网，等到系统负荷降低时再将数据同步到自己服务中。

如果在一个较长周期，“ 超负荷 ” 计算是一个常态，那么可以规划将资源、计算通过其他云服务长期负担，如用户的 profile 的操作，直接存在人人网中，将用户的文档直接存储在用户使用地网盘中，这样就尽快扩展了自己服务能力。

水平扩展是云服务协同其他服务，通过使用其他服务的数据、计算能力，提升自身服务容量的一个有效手段。

18.3　构建高可靠性

18.3.1　可靠性理论与云计算平台需求实现

1. 可用性与可靠性

一般来讲，对服务系统考察的是系统总体的可用性，即系统不间断持续提供服务的能力，而对终端系统，一般采用可靠性来评估系统稳定提供功能，不出现错误的能力。如前所述，对于云服务系统，系统可用性要求比传统软件要高，业界要求的最高水平是到 99.99% 的可用度。

由于系统可用度计算的是正常系统提供服务时间占系统服务时间的比例，对于 99.99% 的可靠度指标，即一个月中断服务时间少于 5 分钟，中断服务时间包含维护、升级时间，对于一个 6 个月维护一次的云服务系统，如果维护需要中断服务，系统完成维护周期为一个小时，那么系统可用性指标降低到 99.97%，如果系统升级也中断服务，那么系统可用性指标将不会超过 99.9%，因此在云服务构架时，根据可用性指标要求，要考虑到系统维护、升级等对系统可用度的影响，要考虑升级与扩展、甚至维护都能不间断服务，这样才能确保系统达到可用度指标。

2. 系统可用性模型

在系统构架时，首先要建立可靠性模型，首先要区分串联和并联因素。假设系统 V 有两个子系统，其可用度分别是 V1 和 V2，并且 V1=V2=99.99%。

- 如果系统中两个子系统，无论哪个子系统失败都导致系统失败的，在可靠性模型中是串联的。

$$系统可用度\ V = V1 \times V2 = 99.98\%$$

- 如果系统中两个子系统，其中一个失败并不导致系统失败，只有两个同时失败，才导致系统失败的，在可靠性模型中是并联。

$$系统可用度\ V = 1 - (1 - V1) \times (1 - V2) = 99.999999\%$$

因此，在实际部署中，一般使用的是并联方式，如备份、集群等方式，以提升系统可用度。

以下面一个传统 WEB 服务系统为例，在可靠性模型中（见图 18-1），互联网接入、路由器、交换机、前端 Apache 服务器、应用服务器和数据库服务器，是一个串联系统，一般来讲，路由器、交换机有 99.9996% 以上的可用性，互联网单接入网络可用度 99.9%，而 WEB、app、database 服务器可用达到 99.9%，系统可用度是 ：

$$V = 99.9\% \times 99.9996\% \times 99.9996\% \times 99.9\% \times 99.9\% \times 99.9\% = 99.6\%$$

如果以 99.99% 可用度为目标，直接建立一个备份系统，系统可靠度 ：

$$V = 1 - (1 - 99.6\%) \times (1 - 99.6\%) = 99.9984\%$$

可见其系统可用性指标可达 99.998%，就能达到系统可用度指标，其方式表示如图 18-2 所示 ：

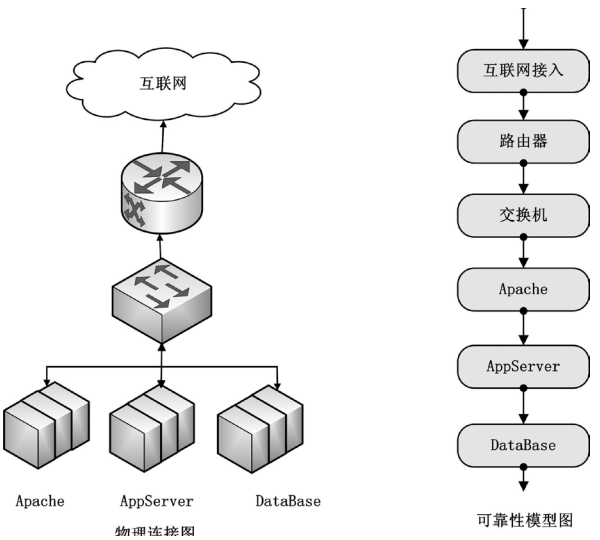


图18-1　云服务可靠性模型图（1）

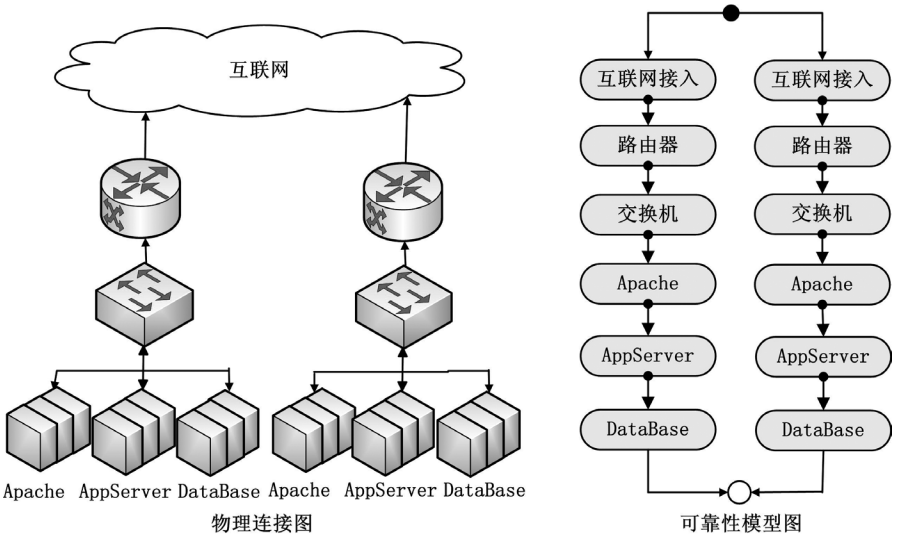


图18-2 云服务可靠性模型图(2)

这使得服务系统能够达到要求，但确实不是一个很经济的模型。同时，这种模式也引入不少问题，如数据库同步，用户如何同时接入等等。因此在常见系统中，采用混合（meshed）接入方式，让接入有备份，而且用户看到地址是统一地址，路由和交换有备用系统，能够尽快回复，Apache 采用 HA 热备，AppServer 在 Apache 作用下双机集群，而数据库采用 HA 热备模型，系统模型就变成了图 18-3 所示的模式 ：

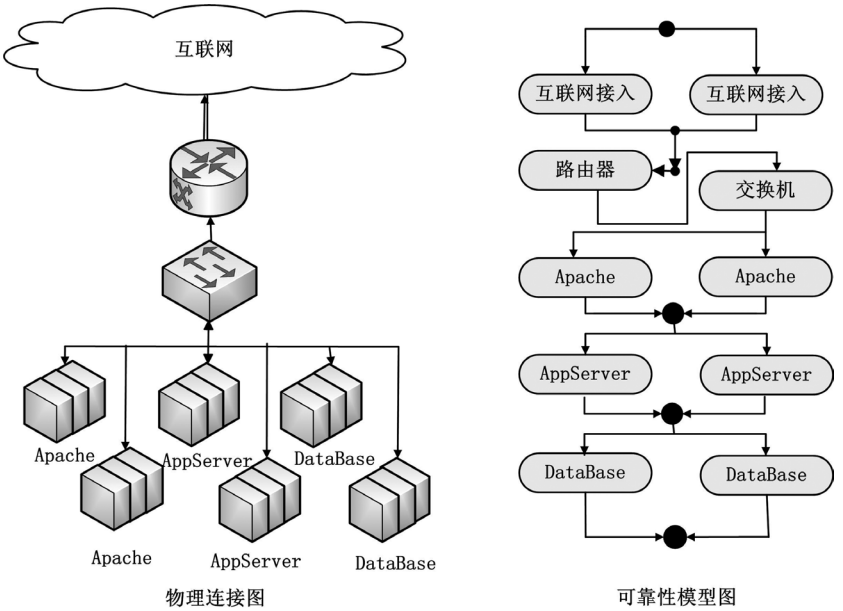


图18-3 云服务可靠性模型图(3)

其计算相当复杂一些 ：

V = [1-（1-99.9%）x（1-99.9%）]x99.9996%x99.9996%x [1-（1-99.9%）x（1-99.9%）]x[1-（1-99.9%）x（1-99.9%）]x[1-（1-99.9%）x（1-99.9%）] = 99.998%

这个模式也达到可用度指标，但更实用。

在构造高可用度云计算服务系统时，首先考虑各个实现环节的关系，一般在构架设计时，根据可靠性指标，综合考虑系统可靠性、成本等诸多因素，先要从理论上保障可靠性指标在各个环节分配合理性，主要考虑关键节点，通过多冗余保障系统可用性，设计时还要考虑在升级与维护时，系统仍然可以提供服务。

3. 云服务平台可用性设计

（1）建立模型

在系统设计时，对可靠性首先是建立相应可靠性模型，在模型层面，将指标分配到各个子系统，再考虑各个子系统能否完成可靠性指标。要照顾每个环节的实际能力来分配可用性指标。

比如一个单机程序，持续一个月不停止服务的概率是很低的，假如其一个月需要维护一次，比如服务重启，需要停止 2～3 分钟，而半年需要服务器重启，需要 4～5 分钟，这样来看，光是因为维护，服务的可用度就降低到 99.99% 的水平。加上各种故障时间，包括软硬件故障，网络故障，电源故障等，单机系统能够到 99.9% 就已经非常高了。

在分配指标时，由于仅仅有一个总体可用度指标，而系统环节很多，变量很多，例如上面的 WEB 服务，有互联网接入、路由、交换、Apache、应用服务器、数据库六个环节，仅仅知道要达到 99.99%，根据可用性模型，还是计算不出每个环节的可用度，对最后一个模型，可以知道方程式是

$$99.99\% = [1-(1-V_{互联网接入})\times(1-V_{互联网接入})]\times V_{路由}\times V_{交换}$$

$$\times[1-(1-Vapache)\times(1-Vapache)]\times[(1-(1-Vapp)\times(1-Vapp)]$$

$$\times[1-(1-Vdatabase)\times(1-Vdatabase)]$$

首先排除技术运营积累因素，比如路由和交换环节，公司使用路由服务累积数据时 99.996%，而 Apache 和数据库是公用服务，可靠度也是有累积的，比如单机是 99.9%，这样带入以后，剩下因素可以求解出为 ：Vapp 至少达到 99.6%。

但如果一个系统开发环节不是一个，问题就再度复杂，那么要判断这些系统可靠度，就需要有合理模型了，常见方式包含 ：

- 经验模型 ：如对应用服务器如 Tomcat 或 JBoss 为基层的应用服务，根据积累的经验值，比如单机在同等规模时，可以达到 99.8%。
- 规模比例 ：预估开发规模，一般换算成 ALOC，其可用度与规模成反比。
- 等值法 ：假设自己开发系统可用度值相等，算出相应值，对于实际不能达到的，通过将可能的值带入，再求解其它值，直到每个值都能满足要求。

建立可靠度模型，可能不能一次完成，如果计算出可靠性无法实现，则需按照前面提到的增加并联，降低节点可靠性指标的方法，重新调整模型，再进行计算，直到系统需求得到满足，而各个节点的可靠性指标合理、可实现。

（2）实施的技术因素

云服务平台可用性设计要考虑到以下因素，才能完成高可用系统指标 ：

- 任何一个环节中的任意一台服务器的失败，仅仅影响系统容量，不会引起系统失败，这就要求在系统设计中不能出现单故障点，所有环节都有备份或冗余。
- 当一个服务器出现失败，用户能够被其他服务器接管，保障系统服务能够持续。
- 在维护时，对每个服务分别按照时间计划维护，如同一台服务器失败一样，不会引起系统失败，确保系统维护时间内服务仍能照常提供。
- 在升级时，应该考虑逐个升级，避免服务中断，因此新旧版本之间的兼容性要能够很好解决，保证连续两个版本之间的兼容性。
- 容灾考虑，如果有容灾备份，在升级时，系统先转向灾备系统，先对服务系统升级，再升级灾备。同样，在升级灾备系统前，切换到实际系统。这样能实现在系统升级过程中不间断服务。同样方式可以在生产线出故障时使用。

18.3.2　负载均衡与集群

提升系统可用度，使用较多的方案就是集群和负载均衡，该方案一方面可以消除系统中的单故障点（SPOF，Single Point of Failure），一方面可以提高系统的扩展能力。通过将服务部署到多台机器上，每台机器都能对外提供相同的功能，这样多台机器就组成一个集群（cluster）对外提供服务（见图 18-4）。

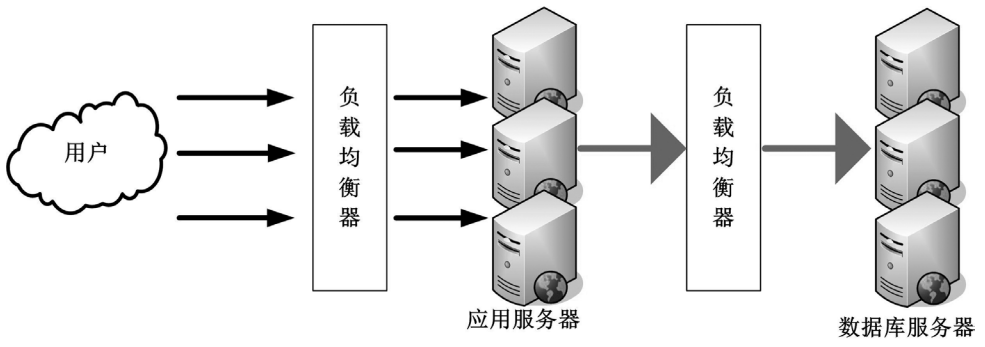


图18-4 系统负载均衡示意

集群环境下，通常会引入负载均衡技术，用户的请求不直接发送到集群中，而是由负载均衡机器接收，并根据一定的策略转发到集群中的业务机器上，其优势在于 ：

- 用户请求不直接和业务机器交互，系统的伸缩性通过在服务集群中加入和删除一个节点来达到，理论上可以无限扩展。
- 负载均衡器可以将请求动态分配到业务机器上，整个集群结构对用户透明，技术运营人员可以对节点进行实时监控并及时进行故障处理，使系统达到高可用。

负载均衡实现可以分为硬件方式和软件方式 ：

（1）硬件方式 ：通常称为负载均衡器，安装在服务器和外部网络之间，具有以下特点 ：

- 性能好。
- 负载均衡策略多样化。
- 流量管理智能化。
- 价格昂贵。

在硬件产品领域，有一些知名的产品可以选择，比如 Alteon、F5 等，能够提供非常优秀的性能和灵活的管理能力。

（2）软件方式 ：通过软件来实现负载均衡，具有以下特色 ：

- 成本低廉，多为开源产品，比如 LVS。
- 成熟稳定、配置简单、使用灵活。
- 效率较硬件方式低，消耗部分系统资源。

18.3.3　双机热备

双机热备（hot - standby）概念上来讲，属于集群的一种。为了保障业务的连续性，不允许服务存在单点，但建立集群又太复杂，这时候可采用双机热备技术。

双机热备技术（见图 18 - 5）是一种软硬件结合的服务容错方案，通常由两台服务器系统和一个外接共享磁盘队列柜及相应的软件组成。“ 故障隔离 ” 是双机热备的工作原理，通过主动转移故障点来保障业务的连续性，双机热备本身不具有修复故障的功能，只是将服务转移到备用服务器上。“ 故障检测 ” 是双机热备的一项任务，采用 “ 心跳 ” 方法来保证主系统和备用系统的联系。

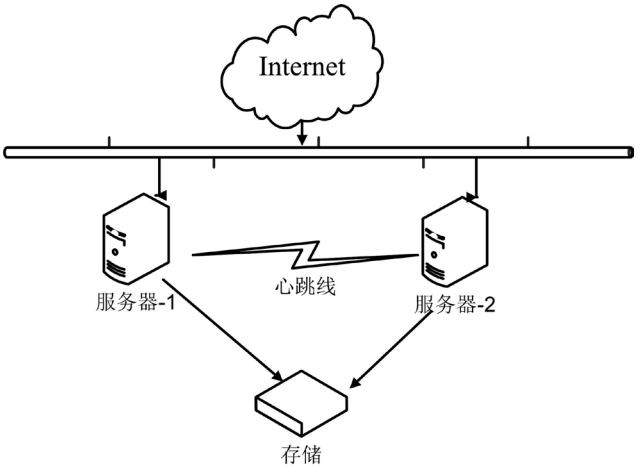


图18–5 双机热备

双机热备的工作模式主要有主备模式、双机互备模式和双机双工模式。

双机热备的数据共享的实现方式主要有 ：基于共享存储（disk array）和基于数据复制（data

replication)。

18.3.4　异地灾备

容灾系统（DR，disaster recovery）是指在相隔较远的异地，建立两套或多套功能相同的系统，互相之间可以进行健康状态监视和功能切换，当一处系统因意外（如火灾、地震等）停止工作时，整个应用系统可以切换到另一处，使得该系统可以继续正常工作。如图 18-6 所示。其中的 GSLB 是一个网络层的异地高可用技术。

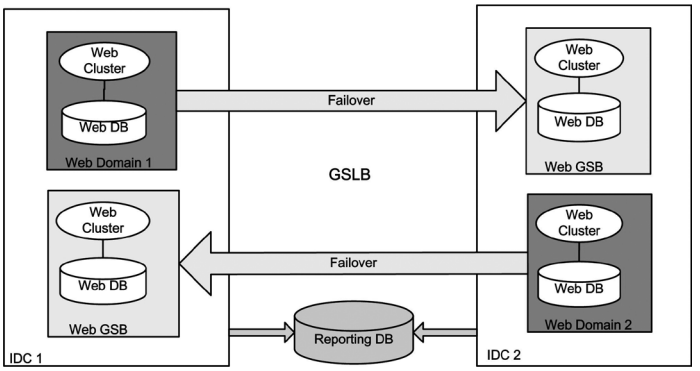


图18–6 灾备系统示意

IDC 即数据中心，比如 IDC1 表示北京数据中心，IDC2 表示上海数据中心。如上图 19-6 所示，各 IDC 之间互为备份。

容载系统从实现的层次来分，可以分为数据级容灾和业务级（也称应用级）容灾。

- 数据级容灾的关注点在于数据本身，在灾难发生之后要确保原有的数据不会丢失或者遭到破坏，但在数据级容灾这个级别，发生灾难时应用是会中断的，数据级容灾在技术上、流程上还不能保证应用、业务的高可用性。
- 业务级容灾是在数据级容灾的基础之上，在备份站点同样构建一套相同的应用系统，这样可以保证应用在允许的时间范围内恢复运行，尽可能减少灾难带来的损失，让用户基本感受不到灾难的发生。

容灾除了需要成熟的软件、硬件解决方案外，关键在于容灾与业务的紧密结合，这对投入和管理能力要求都比较高。

下面做一些具体的讨论。

（1）容灾指标

RPO（Recovery Point Objective）是指灾难发生后，容灾系统能把数据恢复到灾难发生前时间点的数据，它是衡量租户在灾难发生后会丢失多少生产数据的指标。

RTO（Recovery Time Objective）则是指灾难发生后，从系统宕机导致业务停顿之刻开始，到系统恢复至可以支持业务部门运作或业务恢复运营之时，此两点之间的时间。

RPO 可简单地描述为租户能容忍的最大数据丢失量，RTO 可简单地描述为租户能容忍的恢

复时间。理想状态下，希望 RTO=0，RPO=0，即灾难发生对用户毫无影响，既不会导致生产停顿，也不会导致生产数据丢失。但显然这很难，能做到的是尽量减少灾难造成的损失。

（2）数据级容灾

数据级容灾主要是指数据同步，比如文件、数据库数据、内存状态等。数据同步实现的程度，决定了容灾方案最后的效果和能够实现的程度。

对于文件资源，实时地将主站（active）的数据同步到备用站点（standby），如图 18-7 所示 ：

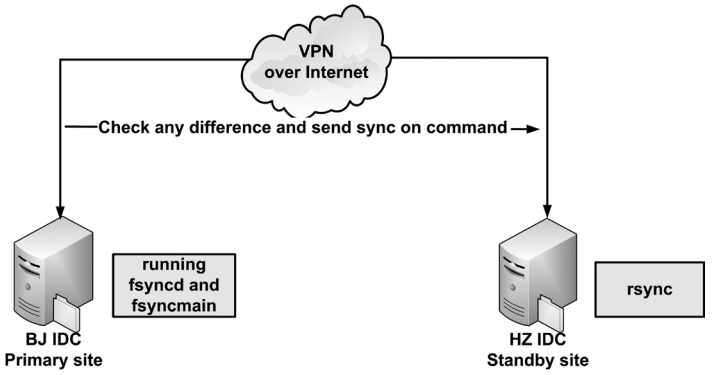


图18-7 系统层（system level）数据同步系统

数据库同步是将一个正在运行的数据库系统中的数据实时或准实时地同步到另外的一个或者多个数据库上，相对于数据库备份，它的实时性更强，如图 18-8 所示 ：

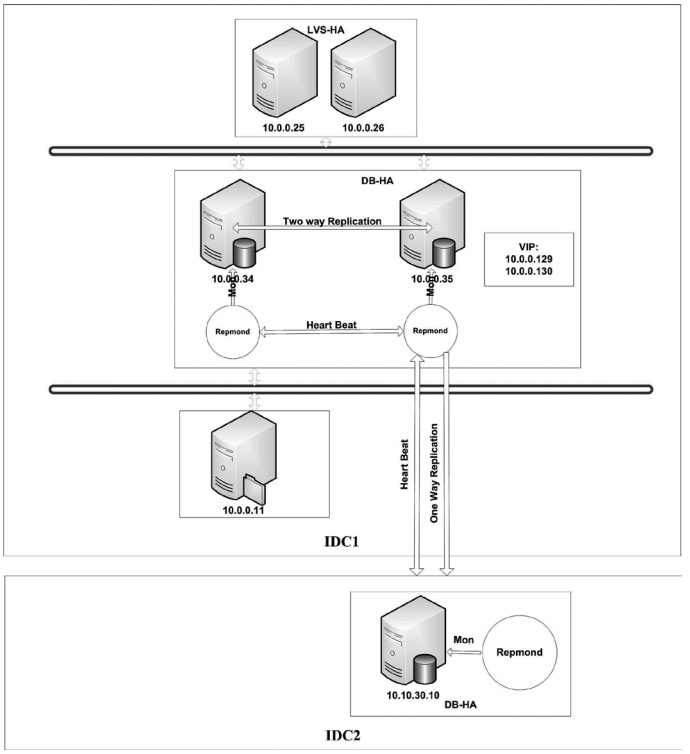


图18-8 数据库层（database level）数据同步系统

Replication Monitor 包括两个部分：site 内部 DB-HA 的 MySQL Server 状态监控，自动 failover（维护 VIP 跟 real IP 对应关系）；site 之间 MySQL replication 状态监控和维护。每个 DB cluster 可以有一个或者多个 MySQL server 之间的数据同步通过循环复制实现。应用通过 VIP 来访问数据库，VIP 和 real IP 所对应的物理数据库的对应关系是 1:1 或者 1:n 的关系。通过维护 VIP 和 real IP 的对应关系来实现自动的 failover，对应用服务透明。负载均衡通过不同的应用连接不同的 VIP 来实现，对应用不透明。

（3）应用级容灾

应用级容灾是在数据级容灾的基础上，在异地建立一套完整的与本地生产系统相当的备份应用系统（可以是互为备份或者全局负载均衡模式）。图 18-9 为硬件 GSLB 的实现示意：

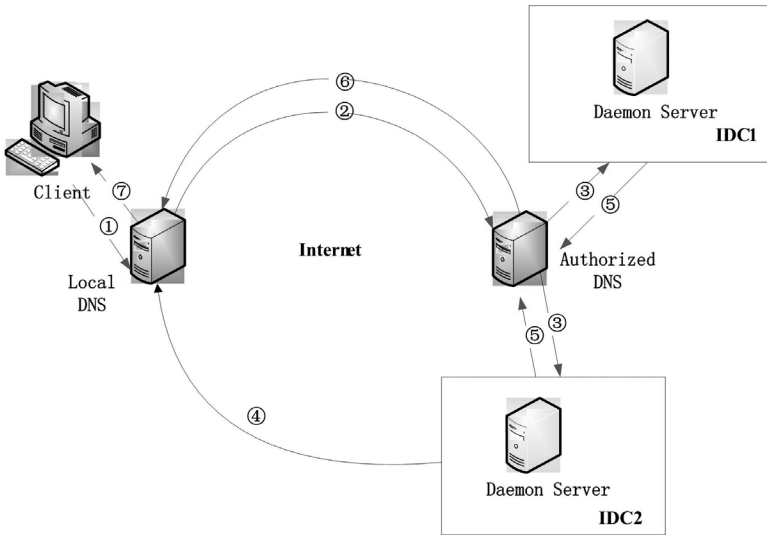


图18-9 应用层容灾架构（Application Level Disaster Recovery Structure）

18.4 构建高性能

“用户速度体验的 1-3-10 原则”，如图 18-10 所示[1]：

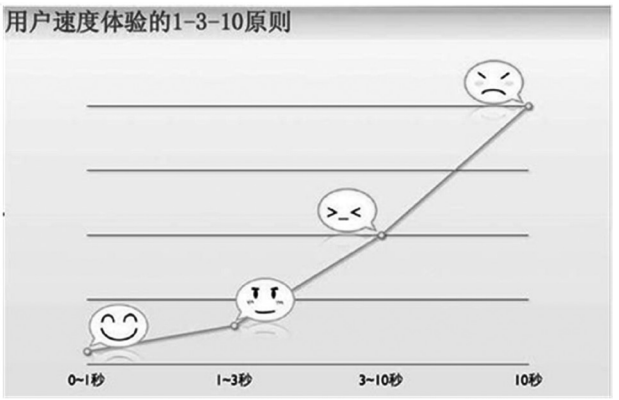


图18-10 用户体验1-3-10原则

分析表明，服务的响应时间如果超过 3 秒，用户的心情会急剧变化，由此可见，高性能是服务平台一个非常重要的指标，是优质用户体验的基础。

在面对大量用户访问和高并发请求方面，使用高性能的服务器、高性能的数据库、高性能的 WEB 容器无法从根本上解决问题，而且投入成本也很高。架构设计需同时考虑高性能、可扩展性和低成本，通过合理优化策略，实现在大容量、高并发条件下的最优用户体验。

18.4.1　系统容量与性能瓶颈

云计算服务是通过互联网向所有客户提供服务的，一般在构建系统时会根据单机容量决定各个环节集群数量，对于 WEB 服务，一般会分为 WEB 接入层、服务应用层和数据层，对于不同服务，系统性能瓶颈可能不同，如对于数据逻辑复杂的应用，应用层可能是系统性能瓶颈，而对于数据集中型服务，数据层决定着整个系统性能，而在互动与接入访问频繁，显示复杂的服务，接入层可能是系统性能的瓶颈，因此对于云计算设计者来讲，首先要评估系统容量对各个环节的压力，然后找出瓶颈，有针对进行优化，如果不能很准确通过评估找到性能瓶颈，就需要搭建系统，通过测试了解系统瓶颈，然后有针对性进行系统性能及容量优化，下面对一些常用的系统性能优化方法进行分析 ：

18.4.2　接入与WEB层容量与性能设计与优化

相对传统的软件产品，互联网服务提供的是一种在线 WEB 应用，界面元素较为丰富，并且用户可以从不同地区，通过不同的网络访问服务。相对于应用层和数据库层的调优，WEB 层的优化相对容易。

（1）采用高效、稳定的 WEB 服务器

选取目前比较流行的几种 WEB server 进行比较 ：

- Lighttpd ：Lighttpd 是一个开源的轻量级 WEB server，CPU 占用率和 memory 开销都比较低，效能较好，模块也很丰富。
- Apache ：Apache 是目前使用最为广泛的一个 WEB 服务，具有跨平台和高安全性等特性。
- Nginx ：Nginx 是一个轻量级、高性能的 HTTP 和反向代理服务器。

主要特性比较如表 18-3 所示 ：

表18-3　几种WEB server的性能比较

Server	Apache	Nginx	Lighttpd
Proxy代理	非常好	非常好	一般
Rewriter	好	非常好	一般
Fcgi	不好	好	非常好
热部署	不支持	支持	不支持
系统压力比较	很大	很小	比较小
稳定性	好	非常好	不好
安全性	好	一般	一般
技术支持	非常好	很少	一般

Server	Apache	Nginx	Lighttpd
静态文件处理	一般	非常好	好
Vhosts虚拟主机	支持	不支持	支持
反向代理	一般	非常好	一般
session sticky	支持	不支持	不支持

对静态页面的请求及响应性能比较如表 18-4，表 18-5 和表 18-6 所示 ：

表18-4 Lighttpd性能数据

n/-c（ab参数）	cpu%	Mem	Requests per Second	Time taken for tests
100000/100	64	60	462.75	21.6
100000/200	67	60	312.07	32.4
100000/500	83	60	137.24	72.8
100000/1000	94	60	126.6	78.9

表18-5 Nginx性能数据

n/-c（ab参数）	cpu%	Mem	Requests per Second	Time taken for tests
100000/100	34.6	140	943.66	10.597
100000/200	35.6	110	924.32	10.818
100000/500	34.3	110	912.68	10.956
100000/1000	37	160	832.59	12.106

表18-6 Apache性能数据

n/-c（ab参数）	cpu%	Mem	RequestsperSecond	Time taken for tests
100000/100	40.6	170	690.72	14.47
100000/200	41.1	180	685.39	14.59
100000/500	42.3	190	633.64	15.78
100000/1000	43.1	200	547.53	18.26

从这些数据的比较可以看到 Nginx 作单纯的 WEB 服务器，性能上比 Apache 要好，特别是在承受压力、带宽及资源消耗上都要优于 Apache。如果以 3 秒的响应时间作为标准，Nginx 能应付不超过 10,000 的并发连接数，如果以 10 秒响应时间作为标准，Nginx 能应付 15,000 以下的并发连接。

（2）从应用服务器中分离静态资源

应用服务器主要处理核心的业务逻辑，对于静态资源，应尽可能的分离，让高效的 WEB server 来处理，从而降低应用服务器的压力。常见的静态资源有图片、JavaScript、CSS、html 等。

（3）动态内容静态化

静态化的 html 页面效率最高、消耗最小，因此在能够静态化的地方，应优先考虑该方案。

一些大型的门户网站和信息发布类网站，都会采用 CMS 进行信息管理，并通过一定的策略自动生成静态页面。对交互性较高的网站，尽可能的静态化也是提高性能的必要手段。

html 静态化也是某些缓存策略使用的手段，对于系统中频繁使用数据库查询但是内容更新很

小的应用，可以使用静态化 html 片段来实现，这些内容在后台更新的同时进行静态化，避免了大量的数据库访问请求。

通过 URI 编码，将网页请求的状态尽量保存在 URI 中，通过相同 URI 获得相同页面，可以通过从 cache 机制，将动态网页静态化，最大限度提升系统访问能力。

（4）开发优化

前面提到的优化方法对性能提高能起到很大的作用，但从开发的角度来看，高质量的代码才是根本之道。在这方面，Yahoo 团队从实践中总结出了 34 条基本的黄金守则 [2]，在这里选取了前 10 条与大家一起分享 ：

- Minimize HTTP Requests
- Use a Content Delivery Network
- Add an Expires or a Cache - Control Header
- Gzip Components
- Put Stylesheets at the Top
- Put Scripts at the Bottom
- Avoid CSS Expressions
- Make JavaScript and CSS External
- Reduce DNS Lookups
- Minify JavaScript and CSS

18.4.3　服务层容量与性能设计与优化

互联网应用是整合所有资源的综合服务，从网络到服务器，从操作系统到应用软件，任何一个环节出现瓶颈或者异常都将降低服务质量，严重的可能会直接导致系统崩溃。

在架构设计上，需要综合平衡各方面资源，比如前面提到的 “ 页面静态化 ”，就是充分发挥 WEB server 的能力，减轻对应用层和数据库层的压力，而这里主要对应用层的一些优化方法进行介绍 ：

（1）缓存

缓存是提高性能的重要措施，合理的应用可以起到平衡资源的作用，比较典型的一些场景有：

- 缓存整个页面或页面片段，比如 HTML、JS、图片等。
- 缓存那些耗时运算结果数据。
- 缓存那些耗时的查询结果和业务数据。
- 缓存上下文相关的数据，比如应用级配置信息和会话级用户的信息等。

这里说的缓存是贯穿 WEB 层、应用层和数据库层的一种综合优化方式，主要面向的缓存对象是相对稳定，但访问频繁的数据。

缓存命中率是衡量缓存设计的一个重要指标，主要影响因素有 ：

- 时效性 ：业务决定的缓存过期时间。
- 容量 ：缓存的大小，受限于硬件结构，比如内存。

· 粒度 ：缓存的健值算法设计。
· 架构设计 ：分析出业务中的真正热点。

缓存的应用在于减少 CPU、I/O、Network 的消耗，从而提高相关资源的效率，常见加载模式如图 18-11 所示 ：

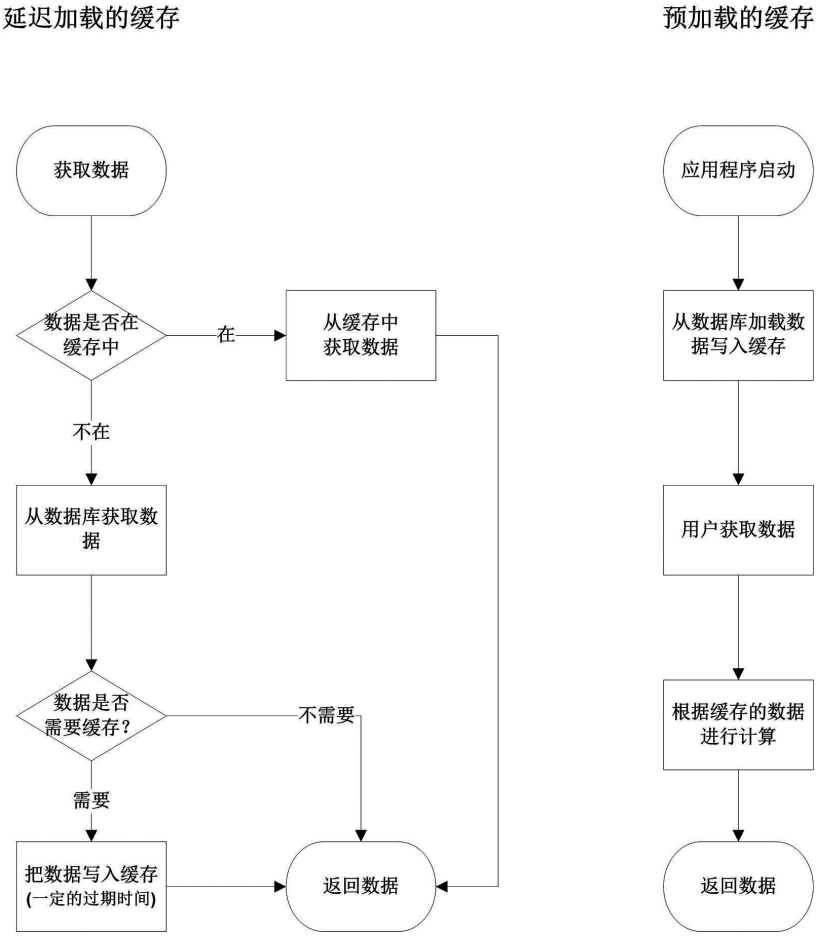


图18-11 缓存加载模式

对于选择集中式还是分布式缓存，需要根据具体情况进行分析，这里不做进一步分析。

在实例介绍部分，将介绍 Memcached 的详细应用。

（2）算法优化

在计算机基础数据结构中，对算法复杂度与系统开销有精确描述，2n、n^2、nlog(n) 等不同算法复杂度，对系统性能影响是巨大的，因此算法优化能极大提升系统性能，这是在设计系统时需要认真分析并仔细优化的地方。

在设计时，首先要考虑运行环境，如现在服务器一般是多核 CPU，如果使用单线程，CPU 性能不能完全利用，系统性能一定不理想。一般来讲，使用线程数量约是 CPU 数量的两倍为佳，太多线程会因为调度开销，降低整个系统的性能。此外，线程安全与同步，往往会引入线程锁，

频繁的线程锁会极大降低系统性能，因此应该在算法中避免频繁使用线程锁，将线程锁范围尽量缩小，能较大程度提升系统性能。

对于搭建系统，可以通过观察各种资源的消耗情况，并结合 profiler 工具进行分析和定位，对程序进行一定的优化，可以充分的利用各种资源，提升服务的整体能力。

下面先介绍两个实际调优的例子 ：

例 1 ：这个例子是关于 CPU 使用的，代码的主要功能是生成一批静态页面，优化之前任务执行时间是 30 分钟，优化之后只需要 2 分钟。优化之前观察到服务器是 4 核的，但该任务是单线程的，仅使用到一个核，单核消耗 CPU 在 30% 左右，同时由于执行时间长，对公共资源的消耗也很大，导致整体服务不平稳。将任务改为多线程，并让单线程的任务片段间适当的 sleep，最终性能大幅提升。

例 2 ：这个例子是关于锁的问题，在代码中使用到 OgnlRuntime 的一个方法来做页面渲染，在 OgnlRuntime 的实现中该方法整个加了同步锁，而代理的方法需要使用数据库资源，观察到有的线程已经占用了 OgnlRuntime 但在等待数据库连接，有的线程占用了数据库连接而等待 OgnlRuntime 的线程，最终导致大量的线程被 blocked，系统无法响应。新版本的 OgnlRuntime 对方法锁进行了分拆，缩小范围，另外也对代码进行了调整，避免发生死锁。

通过前面的例子可以看到，在程序调优上应尽可能发挥各类资源的优势，对耗时任务进行分拆，合理的使用多线程技术，非实时任务采用异步实现，减少锁的使用。

18.4.4　数据层容量与性能设计与优化

应用层的扩展相对容易，数据库层的扩展比较困难，数据库层的合理优化将会极大地提升平台性能，这里分析一些常用的策略 ：

（1）数据库设计：垂直分区、水平分区、读写分离

垂直分区（见图 18-12）是指将不同模块数据存入不同的数据库中，而水平分区（见图 18-13）是指将相同模块数据存入不同的数据库中。无论是垂直分区还是 分区，对应用都是相对透明。

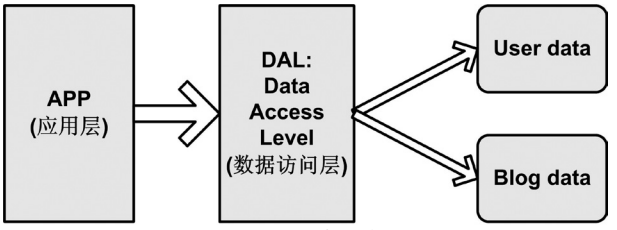


图18-12 数据库垂直分区

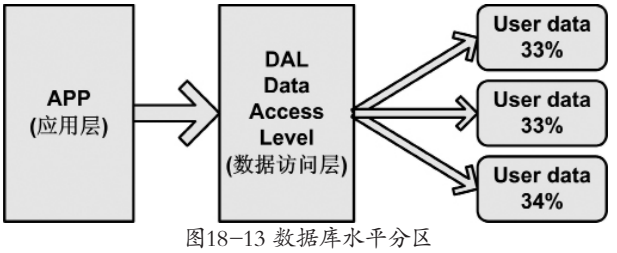


图18-13 数据库水平分区

另外，通过读写分离也可以增强数据层的扩展能力。

（2）数据库表设计

- 建立合适的索引 ：加速数据检索。
- 分离大字段 ：将一些数据特别大的字段从数据库表中分离出来，使用单独的表存放或者是使用文件系统存放，减少 I/O。

（3）SQL 书写

- 避免使用复杂 SQL 语句 ：减少 join 操作。
- 精简字段 ：减少 I/O，仅获取需要的字段数据。
- 避免大数据表连接 ：减少 join 成本。

18.4.5　应对高并发

云平台中存在一些特别高并发的模块，这些模块，使用常规的数据库管理系统，没有办法达到它需要的写的性能要求，需要引入了内存数据库（非结构化存储数据库），来解决这个问题。比如在考试系统中，大家可能都集中在考试结束前的小段时间来提交试卷，但是不会马上来查询结果，这个时候把数据先存放在内存数据库中，在后期再把数据写入业务数据库。因为存储结构的不同，内存数据库可以有更好的并发能力。内存数据库的使用模型如图 18-14 所示。在内存数据库的使用上，内存数据库只是用来处理对高并发写的支撑，在后期的结构化查询中，系统还是从业务数据库中。所以对于程序来说，只要处理内存数据库的写，然后系统会根据内存数据库复制方案把数据复制到业务数据库。

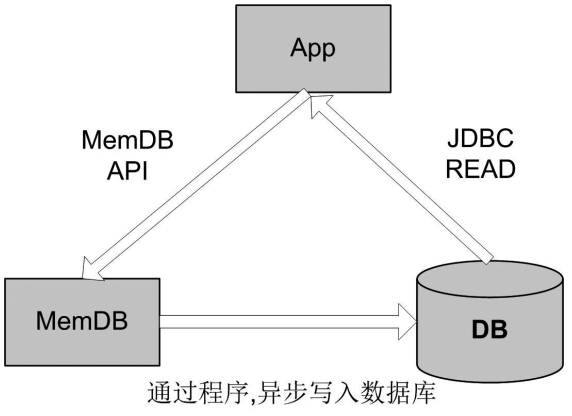


图18–14 内存数据库机制

18.5　构建高伸缩性

对于互联网云服务而言，访问量和数据量随时都可能呈现几何级增长，在架构设计中，需要考虑以最简单的方式来应对这些变化，这就是高伸缩性（scalability）。表 18-7 对两种扩展方式进行对比 ：

表18-7 垂直扩展与水平扩展的比较

	垂直扩展	水平扩展
方式	升级机器硬件，比如增加CPU、内存、网络	增加机器
优点	升级简单，技术运营简单	理论上可无限扩展，硬件成本相对较低
不足	硬件成本较高、存在瓶颈，只能有限扩展	技术要求较高，技术运营难度增加

18.5.1　设计规则扩展与性能

由于云服务扩张较企业服务要快，在构架时，尽量坚持“无状态”和“就地失败”原则，也只有这样，才能适应系统伸缩性的要求。

构架的另一个原则是针对系统瓶颈，在设计时就要有针对性设计，如 WEB 层采用 Apache 集群，集群能力取决于负载均衡器能力，加入采用 LVS，可以支持百万人访问，这是系统集群的最大能力。在构架时，要考虑在能力不足时，采用多个 LVS 一起工作的模式，如采用 DNS 将不同区域访问，指向不同 LVS 地址，这样一般认为可以将系统能力扩展到无限制。

因此，在高伸缩性设计时首先要遵守云计算访问设计基本准则，还要识别系统性能瓶颈点，有针对性进行设计，确保在系统用户急剧增加时能够快速增加部署，确保业务稳定及性能。

18.5.2　并发访问量

水平伸缩存在无限扩展的可能，为了支撑大的访问量，目前大型的互联网服务在架构设计上做了很多努力，实现了在系统用量增加的情况下，仅需增加硬件设备，无需调整架构。

为了做到水平伸缩，应用应该是无状态的，架构设计上通常将有状态的信息集中存入到缓存或者数据库中，业界也称为 SNA（Share Nothing Architecture）架构。比较常用的方案是分布式缓存，比如开源的 Memcached 方案。

18.5.3　并发数据访问与I/O

访问量的增加会提升对数据库的访问频率，而数据量的增加会直接导致数据库读写性能的下降，互联网服务很大的一部分瓶颈来自数据库。垂直伸缩能起到一定的缓解作用。通过增加机器，进行水平扩展，能有效地提升数据库读写数据的能力。表 18-8 对几种扩展方式做了比较 ：

表18-8 几种常用数据库扩展方式的比较

	读写分离	垂直切分	水平切分
方式	将相同的数据复制到不同的数据库中，对数据的读写可访问不同的数据库	将不同功能模块的数据分别存放到不同的数据库中	将原来存放在一个数据库表中的数据，按照一定的规则，分别存放到不同的数据库中
优点	降低数据库读的压力		
提升数据库读的相应能力	易实现	扩展能力强	
不足	对业务有一定要求，比如读多写少，延时限制等	扩展有限，可能引起分布式事务	增加技术运营成本

18.6　构建高可配置性

云服务面向的是多租户，对服务系统存在较多的个性需求，比如界面、功能、流程等。在云服务设计上，通过对典型需求的分析，需要将这些部分设计为可配置（coufigurability），以满足不同的租户的需求。

传统应用和云服务都存在可配置的要求，但云服务对配置要求更为复杂，而且由于可靠性要求，在更改配置时，服务要不中断，能够立即生效。

为了适应云服务面向多租户的需求，一般来讲将云服务配置分为系统级、客户级和用户级，客户级又可称为站点（site）级，这些配置各有侧重点，对运行系统影响各有不同，下面一一予以说明 ：

18.6.1　系统配置

传统服务系统配置以客户应用功能为主，比如对界面、功能、流程不同需求，而在云服务设计时，系统配置一般以技术运营、部署参数为主，一般不会涉及功能性配置。这些配置以配置文件形式存在，设计时要考虑监控配置文件改变，保障配置改变能及时生效。

18.6.2　站点配置

云服务的站点是对特定客户服务的入口，一般站点配置相当于传统业务的业务配置，不同于传统模式，这些配置是不同客户使用不同配置，是系统运行中根据用户所属的客户动态调整的，因此，一般不再采用配置文件方式，而是使用数据库来记录配置，用户登录不同站点加载不同配置项。

站点配置包括了功能、流程、界面等，在构架时，要考虑不同客户需求变化，尽量将各个功能，能够动态组合，根据配置来决定功能、流程和界面，以同时支持不同客户需求。

18.6.3　用户配置

一般来讲，站点配置决定企业用户统一功能集、流程和界面中企业相关的商标等显示需求变化，是企业用户对供应商的统一要求，而用户配置则是用户根据自己的习惯，在企业需求基础上，设置应用功能、流程和自己界面，以提升自己工作效率。

在设计用户配置时，首先要分析不同用户可能的习惯，将界面、功能和流程尽可能按照不同组合予以对比，排除不可能选择，按照剩余方式提取配置，提升系统适应性，以最大限度满足用户需求。

18.6.4　服务配置与技术运营关系

可管理性是云服务基本要求之一，云服务配置层次较多，配置复杂度很高，因此需要根据系统规模，设立专门的配置服务器，对系统中系统配置集中管理，可以查看、修改站点与用户配置，这是云服务系统不同一般企业服务配置的需求。

18.7　构建高可管理性

云计算服务数以万计的企业客户，可靠度要求很高，而系统可靠度来源于系统维护，系统可管理性是系统成败的一个关键因素，从另一个角度来讲，系统可管理性（managability）决定着系统可靠度实现，比如一次维护需要中断服务 1 分钟，那么要达到 99.99%，一个月系统可以维护 4 次，系统只要保障运行一周多不出问题，系统就能达到 99.99%，而如果一个系统维护一次需要中断服务 20 分钟，系统需要保障运行 5 个月才能达到 99.99%，这对于系统实现来讲，难度可想而知。因此，可管理性是系统高可靠度的一个关键点。

18.7.1　系统维护周期

在做构架设计时，要考虑系统维护周期，维护周期长短、维护时服务中断时间等因素是决定系统可靠性的时间指标，因此在构架时需要考虑以下因素 ：

（1） 系统中各个服务器维护流程
（2） 系统中各个服务器维护周期
（3） 最长维护路径
（4） 各个环节需要时间
（5） 通过何种方式保障维护时系统服务中断时间最短或不中断服务。

根据这些需求及构架设计的部署设计，使用路径覆盖方式，估计出系统维护中断服务时长，一般来讲，UNIX/LINUX 服务器需要在 3 个月到 6 个月重新启动一次，而设计服务程序要根据实际情况决定服务重新启动时间，以保障系统服务可靠和安全。

在系统设计时明确维护周期，每次维护服务中断时间，并根据维护需求设计维护系统，提供给技术运营人员合适的系统接口，以确保维护实施一致性，减少人为因素导致系统服务中断，对于达到系统可靠性和降低技术运营成本都很重要。

18.7.2　系统维护与服务中断

生产线运营的理想情形是在不中断系统服务的状态下进行维护，这一般可以通过备份系统实现，如前面提到的灾备系统。当系统需要升级或维护时，先将系统切换到备份系统，然后对当前系统进行维护或是升级，这样服务中断时间就降低到切换时间，如 30 秒，极大提升了系统可靠度，这是云服务常用的维护模式。在构架时要考虑设计这样的系统及操作，最好能无缝切换，不中断服务。

18.7.3　系统可配置性

云服务系统配置相对复杂，升级与更改配置都不是一个简单的事情，因此，在构架时尽量设计用于系统配置与升级的 API 与界面，并建立集中式的配置管理系统来管理不同层次配置，来提升系统可管理性。

集中式的配置管理系统的讨论可以参考本书的第二部分 “ 技术运营管理 ” 中的 “ 云服务的生产设计 ” 章节。

18.7.4　系统监控能力

好的医生总是建议病人提前保健，而不是看病吃药。而对于云服务生产运营来讲，是要将系统问题消灭在萌芽状态，而不是等到系统出错、服务中断、导致客户不满或流失，那么如何检测系统处于健康状态，也是维护服务人员更关心的事情，对于人来说，使用各种仪器体检能够检测健康隐患，而系统监控如同各种医疗器械一样，提供系统各种参数，以诊断系统隐患，因此构架时设计系统监控系统，对于云服务来讲是非常重要的。

监控应该是覆盖平台的所有资源，包括应用服务的状态、应用服务涉及的相关软硬件资源等。对于云服务，几乎所有功能都涉及网络，网络状况监控，可以使用 SNMP，对流量、丢包、抖动等基本网络参数，以及路由、交换、服务器等物理设备运行情况及拓扑变化情况都能通过 SNMP 监控工具进行监控。

服务器运行情况监控，主要是服务器硬件如 CPU、内存、硬盘等情况监控，都能通过如 Nagious 这样监控工具，提供实时数据。

这里是监控系统的几种展现方式如图 18-15、图 18-16 图 18-17 所示。

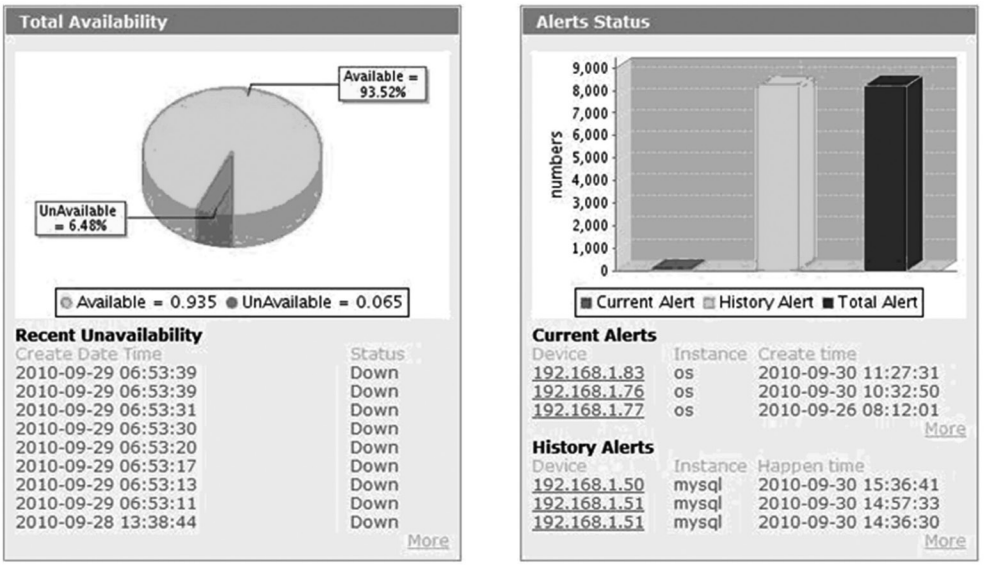


图18-15　服务系统运行状况图（1）：可用性数据及报警统计

	Host Status	Host Name	IP	Enabled	Mail Alert Flag	Sms Alert Flag	SystemInfo
		Teacher-DB110	192.168.1.110	Yes	Yes	Yes	
		VM-WEB-78	192.168.1.78	Yes	Yes	Yes	
		VM-WEB-84	192.168.1.84	Yes	Yes	Yes	
		VM-WEB-81	192.168.1.81	Yes	Yes	Yes	
		Demo	192.168.1.95	Yes	No	No	
		Beta	192.168.1.96	Yes	No	No	
		VM-WEB-74	192.168.1.74	Yes	Yes	Yes	
		OnlineReg	192.168.1.88	Yes	Yes	Yes	
		VM-WEB-75	192.168.1.75	Yes	Yes	Yes	
		QuartzDB	192.168.1.118	Yes	Yes	Yes	

图18-16　服务系统运行状况图（2）：按照时间体现的运行状态

图18–17 详细警报日志

应用层监控设计到的因素更多、更为复杂，需要在设计应用程序时就考虑监控要求。比如系统需要通过连接池同数据库连接，连接池健康程度，如当前连接健康情况，需要提供专门监控接口。服务负载，包含处理并发等情况也需要监控，这些都需要在设计时考虑，而服务一旦构建，再开发难度就相对较高。

因为监控的重要性，本书的第 19 章“ 服务运营的技术监控体系 ”专门对监控做了讨论。

18.7.5　日志记录与错误处理

应用监控的因素很多，而采用日志方式，将系统运行信息输出，用于检测系统情况或判断问题都是一个简单易行的方式，因此日志系统是需要在构建时设计，一个完善的健全的日志处理机制对系统维护与排除都十分重要。

（1）重要性

了解日志机制，首先要理解日志的重要性，即在云服务应用中，日志能起到怎样的作用。

- 系统运行和服务状况分析，比如通过 Nginx 的访问日志可以粗略地估计访问量，对 Jboss 的错误日志进行分析，可以及时发现潜在问题。
- 用户行为分析，通过日志跟踪，可以挖掘出用户的使用习惯和操作轨迹，为设计测试用例和提供附加服务提供直接依据。
- 生产环境问题分析，用户使用过程中会反馈各种各样的问题，日志为快速地定位和分析这些问题提供最直接的信息。
- 提供历史证据，对一些关键业务操作进行日志记录，用户在需要的时候可以对这些操作进行查证，最典型的就是用户的误操作。

（2）日志种类

基于互联网服务架构的应用，其生产环境都相对复杂，每个层次都有很多日志，有的日志由各类服务器提供，简单配置即可，有的日志需要开发实现 ：

- 操作系统日志 ：如 Linux kernel 日志（系统的 dmesg）。
- 基础服务日志 ：如 Linux xinetd、ssh、smtp 等基础服务日志。
- 专用服务日志 :如 Nginx 的访问日志和错误日志、Jboss 的应用日志、DB 层 SQL 日志、各类资源的运行日志等。
- 业务服务日志 :开发业务的操作日志、错误日志、服务开销性能日志等，一般需要专门开发。

（3）日志级别与配置

完善的日志机制必须支持不同级别日志，如 Linux log 系统支持 emergency、alert、critical、error、warning、information，debug 等 7 个级别 log，用于不同情况，在设计 log 系统时必须考虑区分不同级别，以方便控制输出 log 输出大小，一般服务器在运行时会打开到 information 级别 log，debug 级别 log 一般用于研发排查问题。

Log 系统必须能支持配置输出到不同位置或系统，如输出到文件、数据库，或是输出到 log 监控系统。此外通过配置可以实现不同级别 log 到不同系统，以保障系统运行效率。而 log 文件的切分、备份等功能都是要考虑的因素。

Log 系统另外一个因素是对服务系统性能的影响，要采用 relay 系统保障不影响系统正常运行，在系统有空闲时才考虑将 log 最终写入，典型的如 Linux syslogd-ng 就通过 relayfs 实现这一功能，在保障可靠情况下，同时保障运行系统性能需求。

（4）日志管理

在原始日志采集到之后，需要有一定的策略对日志进行管理。原始日志一般会非常庞大，这时候需要进行历史归档，或者是进行二次加工，为日志分析提供更加直接的依据。

（5）日志分析

日志采集是为了进行分析，并最终为各种决策提供有效的参考依据。日志分析方式有很多，这里不深入分析。

（6）日志监控

运营系统都会输出大量日志，如果能够监控日志信息，对于判断服务健康有着非常重要的意义，因此，对日志实时监控，并更加不同级别日志做不同动作，对于确保系统稳定运行，服务有效稳定是十分必要的。

常见做法是将日志输出给 Nagio Nagios 等监控系统。

18.7.6　用于服务的配置、监控与日志系统

对于云服务中的 SaaS 服务来讲，一般用户使用功能，不太关心配置、监控与日志，但对于 PaaS 和 IaaS 服务，用户在其上使用的是基础设施和平台的服务，对于相关网络、服务器及服务状态监控、日志、配置都有不同需求，这些功能既是内部服务与支持所需要的功能，又是外部客户需要的，因此，在设计时还要区分这两个方面需求。

18.8　案例研究：云培训服务平台

本节将以一个云培训服务平台为例，围绕技术架构关注的几个方面进行详细分析。

18.8.1　业务介绍

主要业务场景如下 ：

- 以在线云服务的方式，为全国近 1,000 万家企业提供在线培训服务。
- 每个企业可同时开展多个培训项目，培训参与者可以是企业内容员工，也可以是外部人员。
- 培训项目的周期不固定，有的项目可能超过 1 年，而有的项目可能是 10 天之内结束。
- 培训项目人数不固定，大的项目人数超过 10 万，而规模小的项目可能少于 100 人。
- 培训学员使用服务的时间不固定，平台需持续可用。

主要的业务模块构成如图 18-18 所示 ：

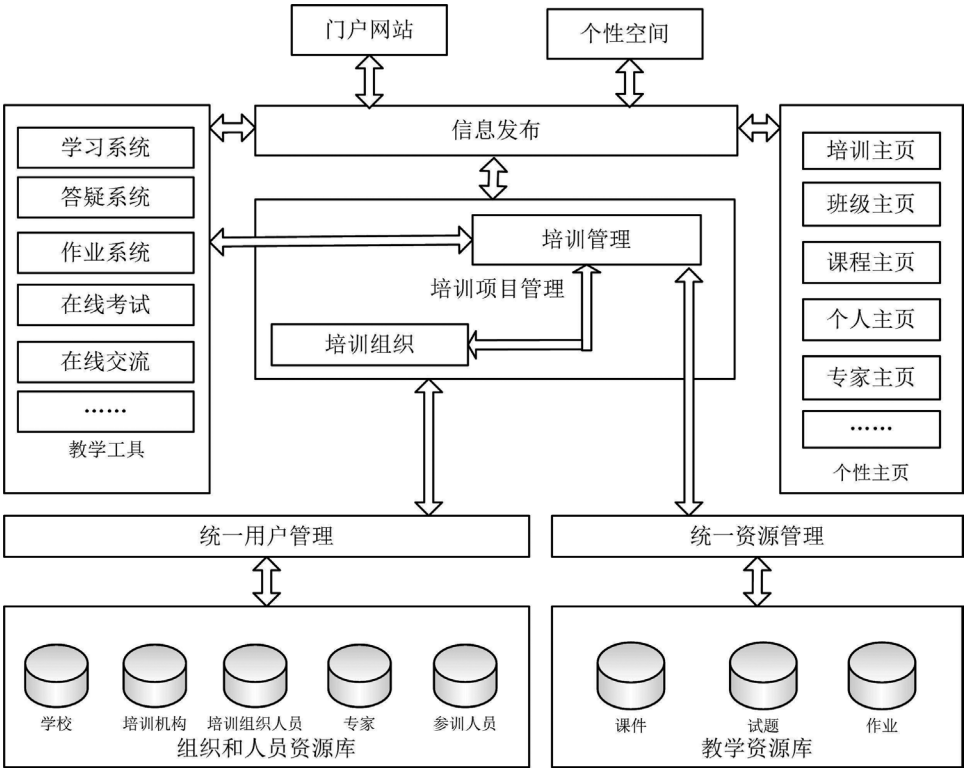


图18–18 业务模块构成（案例）

系统模块划分 ：

- 平台管理中心 ：对教学资源、师资资源等基础资源进行统一管理。
- 项目管理中心 ：培训项目创建、培训项目组织、培训过程支撑、培训结果考核和成绩管理。
- 教学工具包 ：提供丰富的培训工具，比如课件学习、在线作业、在线答疑等，每个项目可选择使用。
- 信息系统 ：对培训项目过程性信息的集中展示。
- 认证中心 ：对平台用户进行统一认证。

18.8.2　设计难点

从业务需求来看，该平台功能丰富，稳定性和性能方面有着非常高的要求，潜在的并发访问

很大，如何提高平台的可伸缩性将是架构设计要考虑的主要问题之一。表 18-9 是一个典型的培训项目的数据，单个项目的数据达到百万和千万级别 ：

表18–9 容量需求分析（案例）

基础数据：（w表示万，k表示千） 参训人员 10w（辅导员+学员），每个班级学员30人 学习中心分4级，每级下设5个子中心和10个班级，约1w 每个班级辅导老师数量2，每个学习中心辅导老师数量1 开设10门课程, 每个学员选择5个课程	作业： 每个课堂2个作业 10x2=20 作业数 10w学员 x 5个课程 x 2个作业＝100w 作业批改记录数 100w 作业评论数 100w x 10 ＝1000w
在线学习： 学习时间记录数 10w学员 x 5个课程 = 50w 学习时间明细： 50w x 40个记录点 = 2000w 学习笔记 50w x 20次 = 1000w	在线考试： 考卷考题 5个课程 x 2次考试 x 20个题目＝200 答卷和批改数 10w x 200 = 2000w 考试成绩 10w x 5 x2 = 100w
文章： 发表数量 10w x 5 x 5＝250w 文章评论数 前40k x 100 = 400w	学习简报： 简报数 2w 简报评论 前1k×1000＝100w
答疑 ： 答疑数量 10w x 5 x 3= 150w 答疑回复数 150w x 10=1500w	学习总结： 学员心声 10w×5=50w 培训总结 10w×5＝50w 班级留言 10w*6=60w
积分： 答疑积分数 1500w + 150w = 1650w 文章积分数 250w + 400w = 600w	通知： 公告 1k，动态 1k，服务栏 100

这里重点分析三个场景 ：

- SSO 和 session ：平台中存在多个服务，必须实现单点登录，对单个培训项目，20 分钟内 10 万人要求登录完成，相当于 100 次 / 秒的登录压力。登录之后，10 万人 1 小时内 100 次左右的页面点击，相当于 3,000 次 / 秒的页面点击压力。对此，架构设计需要考虑用户登录信息的存放和访问，比如支持 site failover 和 load balance 等。
- 在线学习 ：在项目培训过程中，学员在线学习课程资料，平台需要记录学员详细的学习时间，这是学员考核和成绩计算很重要的组成。比如有的培训要求在线学习时间合计 40 个小时，平台每隔 15~30 分钟自动记录一次。可以预见，对架构的挑战在于大数据量和纵向的集群扩展。
- 在线考试 ：在项目培训的中后期，学员需要参加在线考试，考生的考卷是根据规则从海量题库中生成，不能重复，同时参加考生的数量数以万计，尤其在提交答卷的时候，短时间内并发访问量非常之大，并且在答卷提交之后需要对客观题进行自动评卷，给出考试成绩。对架构的挑战在于如果应对高并发写和运算。

18.8.3　架构设计

为解决这些问题，在架构设计中综合应用了静态内容分离、统一认证和集中式 session 服务、集群、缓存、数据库分片、数据库读写分离、内存数据库等技术，成功地应对各项挑战，下面详细进行介绍 ：

（1）总体架构图见图 18–19

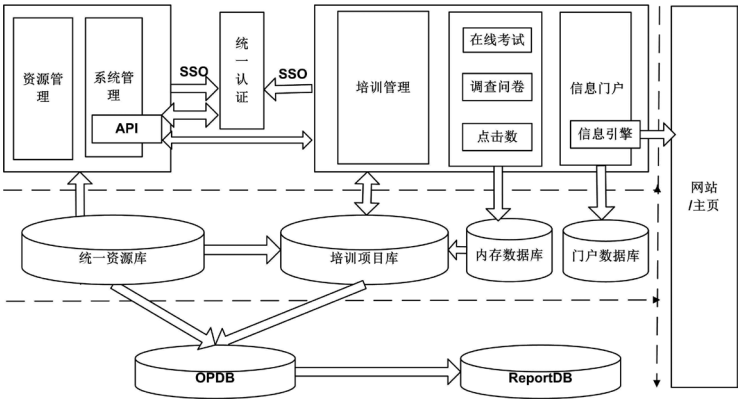


图18–19 总体架构图（案例）

（2）总体部署图见图 18–20。

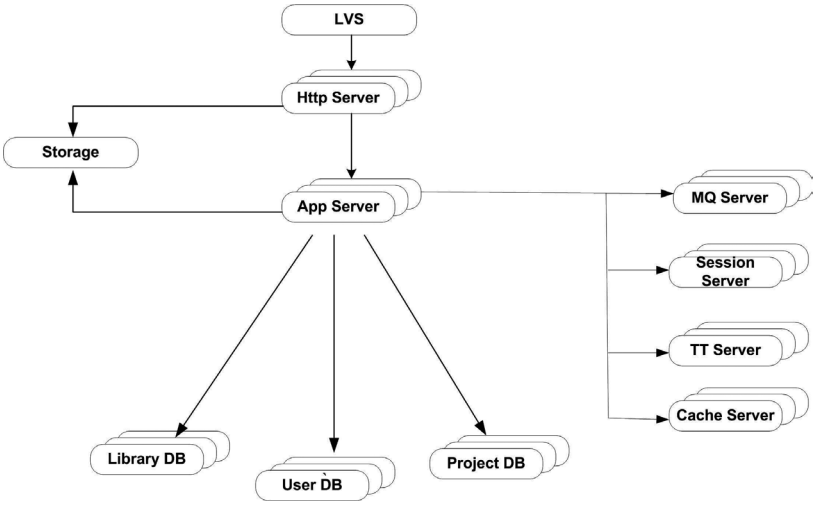


图18–20 总体部署图（案例）

（3）架构综述

· 缓存设计

这是一个典型的数据库应用，在这样的应用中，数据库的性能往往决定了整个应用的性能，而减少数据库的访问，就成了提高应用性能的一个非常重要的手段，在设计中，采用 Memcached 作为缓存服务，从多个层面提供缓存 ：(见图 18 - 21)

- 关键性缓存 ：比如认证信息等，用户登录之后不会变化也不允许丢失的信息。
- 可重复加载的应用缓存 ：用户登录之后的权限、菜单等类似信息，无须每次都从数据库加载，缓存失效之后，也可以重复加载，最终用户不会有任何影响。
- Hibernate 的二级缓存和查询缓存 ：当第一次数据库访问之后，把对象存放在访问速度高于数据库的缓存中，下一次来取相同的数据时候，就直接从缓存中返回。

· Page 缓存 ：较少变化的内容生成静态页面或者使用 page cache。

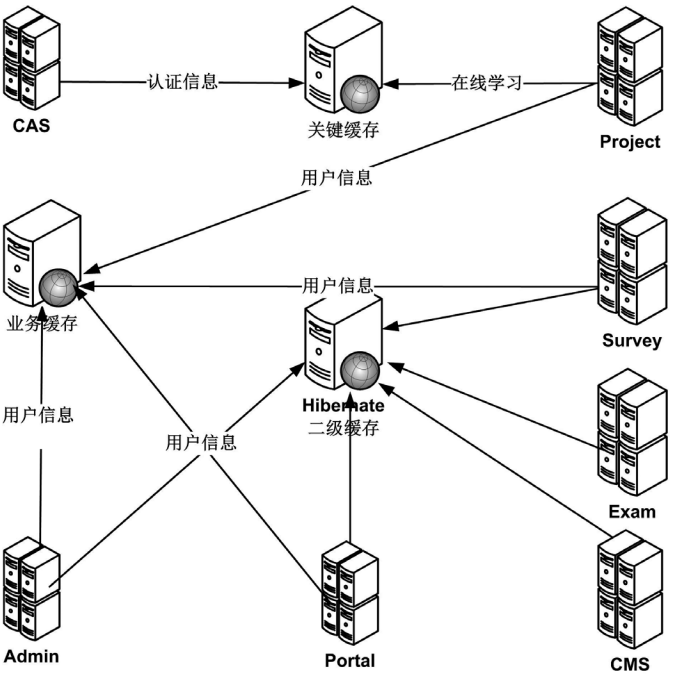


图18–21 缓存分布图（案例）

· 集群设计

从业务模式来分析，平台需要满足无限扩展的可能，在可伸缩性方面有着强烈的需求，各个层面的集群支持则属必然 ：

· 应用集群 ：应用设计成无状态，系统采用 CAS 统一认证，session 通过 session server 来统一管理，session server 同样支持集群部署，如果应用服务器在性能上成为瓶颈，我们可以添加服务器来支持。

· 数据库集群 ：数据库设计首先支持水平和垂直切分，在切分之后，通过主备的方式做读写分离，如果属小规模的培训服务，可以把不同的 ProjectDB 部署在同一个数据库实例中，节约成本。如果是较大规模的培训服务，则可以创建主备集群并且使用读写分离来处理。

· 缓存集群 ：所有的缓存服务都设计为支持 HA 和集群部署。

18.8.4　架构进一步的改进

平台设计之初，对于 HA 的考虑不足，曾经出现部分服务无法使用，在后续的架构设计方面，通过深入的分析，做出了进一步调整 ：对于每个子模块，比如每个教学工具，都能够做到集群部署，而在监控方面，更是深入到每个业务层面。

随着时间的推移，新的数据在不断增加，历史数据和不活跃信息也在不断累积，平台架构也将

随之进行调整，比如引入全文检索机制，改变检索方式，并将历史不活动数据与活动数据分离，将其存入历史库，保持及时服务库效率，同时要提供离线库查找机制，保障系统服务性能最优。

随着培训项目的增加，新的挑战随之来临，除了数据量的剧增，由于采用了切片的方式，数据库的数量也在不断地增加，对于技术运营和监控难度也在不断提升，而对数据挖掘和分析方面的能力却在递减。随着架构的演进，引入了 Big Table，一种全新的分布式数据存储方案，在 Big Table 集群基础上，结合缓存策略和相关调优方案，平台在性能、可靠性和扩展性方面都有了很大地提升，而技术运营的难度也逐步降低。随着云计算相关技术的不断成熟，架构设计有了多样的选择，而在不同的阶段采用不同的方案也成为构架的一种乐趣。

18.9　本章总结

本章结合在线培训的实例，围绕云服务技术架构，重点关注的几个方面进行介绍，即 ：

- 高可用
- 高性能
- 可管理性
- 伸缩性

云服务面临诸多挑战，技术架构为业务服务提供强有力的保障，随着业务的发展，技术架构也在不断的演变，在合适的时机采用适合的架构才是设计的真谛。

在实际工作中，还会碰到诸如 “ 多版本共存 ” 和 “ 实时部署 ” 等相关问题，都需要在构架时考虑。这些考虑与实际系统的可用性、可管理性、软件版本兼容性等相关。这些内容相对单一，因此本章不做重点讨论。

参考文献 ：

[1] “ 海量 SNS 网站的柔性运营 ”，2009，邱跃鹏

[2] “ Best Practices for Speeding Up Your Web Site ”，Yahoo developer,

http://developer.yahoo.com/performance/rules.html

第19章　服务运营的技术监控体系

19.1　监控体系简介

19.1.1　概述

监控系统是服务运营体系的重要组成部分（就像航空运营中的雷达系统一样）。如果没有监控，生产环境的状态将不能被有效展现，甚至失去控制。

完善的监控服务能够快速地发现故障、定位故障点、诊断故障原因、帮助制定解决方案，从而缩短服务停止时间和提高客户满意度。

有效性（effectiveness）和高效性（efficiency）是监控体系的两大挑战。具体地说 ：

- 有效性（effectiveness）：所有的生产线上问题都应该被报出。
- 高效性（efficiency）：不应该出现问题误报。

19.1.2　云服务所面临的挑战：应用为中心

与传统的 IT 产品如软件不同，云服务有着更大的挑战。那就是，云服务是以应用程序为中心的（application centric）。监控需要覆盖到应用程序层。这样的要求带来的新的挑战是 ：

（1）应用程序的复杂性

每一个云服务提供商都有其专有的应用程序。没有通用的工具或软件来监控他们。必须根据各自业务服务的逻辑，单独建造应用层的监测。

（2）客户体验的监控

即使所有的应用程序模块都启动和在运行中，应用程序服务仍然可能无法正常工作。例子之一是线上的支付交易。客户通过信用卡进行付款，基础的监控系统显示所有的网络连通、服务器、数据库、应用服务都在运行中，没有报警。但是，后台的计费系统没有得到支付信息。事后发现，这是由于数据库的一个子模块的执行程序在变更时被误删除，导致传输的数据无法处理而被丢弃。这个应用层的问题，在一般的日志中是无法体现的。

19.1.3　讨论

在这章节将首先对监控进行一般性讨论，然后再将焦点移到应用程序级别的监控。

讨论将使用一个在线支付公司的监控系统作为一个案例研究。对于生产线管理来说，运营难度最具挑战性的三个行业中，在线金融业是其中一个。在线金融业的运营，由于其金融财务的敏感性，比普通的云服务的运营困难大得多。希望这个案例的研究能够对一般的云服务公司的监控体系提供参考。

其他两个运营难度很高的行业是（1）实时通讯应用系统，如网络会议。任何的系统和网络

不稳定性都会马上带来音视频流畅度、文档共享等一系列的用户体验问题。另外一个是（2）在线游戏系统，它需要处理海量用户的同时在线和同步。这三个行业的难度在于：它们都需要实时和双向的应用运营，而且需要 99.99% 级别的服务可用性水平。

19.2　监控的技术架构

19.2.1　云服务监控体系的架构

当客户通过互联网或移动网使用云服务的时候，他们所关心的就是云服务的作为整体的功能和性能体验。但是，对与云服务提供商而言，监控系统是有两个部分的：

- 内部环境监控：如 IDC 和自建的网络骨干网。
- 外部环境监控：如互联网，用户所在 LAN 的环境等。

整体结构如图 19-1 所示。

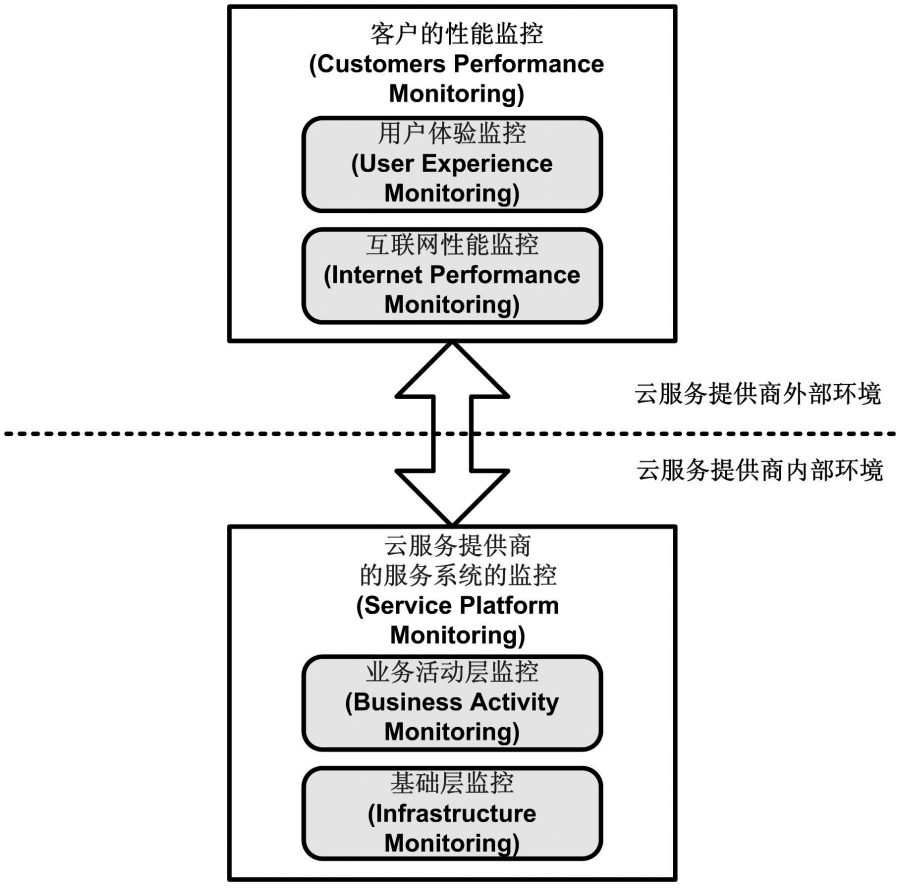


图19-1 云服务的监控架构

（1）云服务提供商的服务系统的监控（Service Platform Monitoring）

这主要是监控云服务商自己拥有的设施的监控，包括在自己数据中心和网络上的设施。这种监控分为两个层次：

- 基础层监控（infrastructure level monitoring）：包括网络、系统、数据库、应用服务等。
- 业务活动层监控（business activities monitoring）：这里是业务逻辑层的监控，如业务流量等。这会在后面的案例中做详细地讨论。

（2）客户的性能监控（Customers Performance Monitoring）

这主要是监控运营商自有的设施之外的环境监控，比如互联网。对云服务商的挑战是，当用户通过自己公司内网，跨过互联网或移动网来使用服务时，一旦出了问题，他们是希望云服务商来解决的，而不管这些是不是云服务商的设施。这就像是航空服务，如果因为天气不好，乘客被耽误飞机。但是乘客会把天气耽误飞机的原因归咎于航空公司一样。因此，航空公司要严密监控天气，做好防范和应急措施。这就是云服务运营商要做的同样的事情。用户体验监控包括：

- 互联网性能监控（internet performance monitoring）
- 用户体验监控（user experience monitoring）

19.3　云服务运营商的服务系统的监控

19.3.1　基础设施的监控

1. 基础设施监控目标

在计算机诞生的时候就一直伴随着宕机、硬件故障、软件故障等等问题，虽然计算机的硬件和软件制造商不断地改进优化，但是从理论上来讲，无故障（BUG-free）的硬件和软件是不可能实现的。

为了让故障发生时，服务商能第一时间知道，并且处理问题，将损失降低到最小，需要对计算机的系统、网络设备、数据库、存储等进行监控，这些就是基础设施的监控。 表 19-1 是基础设施监控的层次。

表19-1 基础设施监控的层次

定义		监控目标
应用服务程序	应用程序进程	每个应用程序所调用的进程
第三方应用程序	第三方应用	Apache、JBoss
系统、数据库	数据复制	Oracle/DataGuard、 GoldenGate、Shareplex
	数据表	Oracle、MySQL
	操作系统及硬件	Linux、Windows、 Linux
	存储	DAS、 NFS、 SAN 等
网络	安全设备	防火墙、IDS、 等
	网络	路由器、交换器

具体的例子如下：

- 系统是否出现磁盘满、网络中断、CPU 或内存出现高负载，指定进程是否在运行、端口是否绑定、HA 服务是否正常。

- 网络设备是否出现网络中断、线路负载过高、CPU 或内存出现高负载。
- 数据库不仅需要对其进行系统级的监控，还需要对数据库服务进行监控，进程是否存在，是否可以提供正常的查询 / 写入服务、数据同步是否正常。
- 存储也需要对其进行系统级的监控，另外还需要监控器是否可以提供网络服务、磁盘空间，高可用性（HA）是否正常工作。

2. 基础设施的监控方法

目前对于基础监控的方法基本上有两种，一种是基于 SNMP 协议开发的监控软件，另一种是通过 C/S（client/server）的方式开发的监控软件。

SNMP 协议出现时间早，且覆盖面广，硬件厂商基本都对其支持，基于 SNMP 开发的监控软件有，Cacti、Zabbix、Zenoss、Ganglia 等等，能大部分满足上面提出的基础监控的需求。对于网络设备，这些监控软件都自带模板，可以满足上面提出的对网络设备监控的需求。但是对于一些特殊的监控，像数据库的查询写入和数据是否同步，这些需要单独开发模板。模板提供远程访问 MySQL 的功能，能将 MySQL 的各种状态获取，然后 Cacti 分析这些数据，根据预定义的法制进行报警，这些不是依赖 SNMP 实现的。这些模板大部分监控软件都已经提供，但是像对存储的 HA 监控，以及系统上的进程是否存在，端口是否被绑定，这些需要单独开发监控脚本，部署到被监控的主机上，然后绑定一个 OID 号，配置到 snmpd.conf 中，在服务端通过指定的 OID 号即可对其监控。

C/S 方式也有很多支持厂商。各大商业监控厂商都有自己的一套 C/S 软件，像 HP 的 OpenView，开源的也有很出名的 Nagios。对于上面的监控需求都能满足，而且 Nagios 灵活的架构，对脚本支持广泛。对于 MySQL 和 Oracle 已经有了一套监控模板，对于 SQLserver 在其 Exchange 站点中也有很多共享的监控脚本可以选择。对于像进程监控如果自己需要单独开发一个监控模板或者脚本，只需要按照 Nagios 接口的规则方式返回 return code 即可，比如

“ 0 ” 代表正常
“ 1 ” 代表 warning
“ 2 ” 代表 critical
“ 3 ” 代表 unknown

通过 NRPE（Nagios 的 Agnet）中配置一个指令，在 Nagios 中直接调用即可实现对其监控。现实起来比使用 SNMP 要更加的灵活简单，且后期维护要更加简单方便。但是对于网络设备的监控则可以通过 check_snmp_plugin 模板实现，实现方式同 Nagios 其他模板一致。

要点和难点的讨论如下 ：

- 要点 ：要收集和整理一份需要监控的硬件资源及 OS 系统对应表，依照此表再结合业务的特性，以及每台服务器承担的角色不同，指定监控项，在根据业务的特性设定报警法制，制定一份最终的监控列表，依照此列表实施监控部署。
- 难点 ：现在一个云系统少则几十台服务器，多则几百台上千台服务器都是很正常的，如此多的服务器加上网络设备都需要进行监控，这将是一项非常庞大的工作。而且每一个服务器或者每一组相同功能的服务器，他们硬件资源和 OS 系统可能都不一样，还有不同业务的特性，因此对他们的监控项，监控方式和阀值也会不一样，要收集一份详细的

监控项目及阀值表也是一项很繁琐的工作。

3. 虚拟化监控

虚拟化能大大节约硬件资源的投入，提高硬件的资源使用效率，可以更合理、更简单的调整和分配资源，使 IT 管理成本也得到很大地降低。对这些虚拟化的宿主机监控也是基础监控的一部分，每台组主机都工作数台甚至数十台虚拟机，一旦出现故障影响面将十分巨大，因此对宿主机的监控也是监控工作的重点。

目前使用的较多的虚拟化技术有 VMware、Xen、KVM、Hyper-V。以 VMware 为例，VMware 提供了一套监控接口，通过 VMware 的监控 SDK，即可获取宿主机运行的各种状态数据，VMware 对外发布了 check_esx3.pl 脚本，此脚本是专门针对 Naigos 开发的，能很好地与 Nagios 结合，此脚本能监控到 Nagios 的宿主机运行的状态，也能监控虚拟机的运行状态，比如监控宿主机的 CPU 和内存信息。

要点和难点的讨论如下 ：

- 要点 ：首先，要为 Nagios 监控添加监控用户，如果没有监控用户的权限 Nagios 无法完成监控，这个只要在宿主机的设置中添加用户权限 ；其次，宿主机监控脚本需要读懂，否则在调整监控的阀值时将无从下手。
- 难点 ：虚拟化监控主要的难点在于，每个虚拟化技术对监控这块的支持不一样，像 Xen 和 KVM 这些开源的技术，当需要对其监控时，需要投入研发资源，针对性的进行开发，其工作量和难度也较大。商业产品，像 VMware 和 Citrix Xen，提供监控接口，而这些接口如何和监控软件结合也是一大难题。

19.3.2　业务活动监控（BAM）

业务活动监控 BAM（Business Activity Monitoring）的核心是应用程序层（Application Level Monitoring）的监控。

三个主要步骤组成有效的业务活动监控 BAM（Business Activity Monitoring ）执行 ：

- 首先，以有效及时的方式收集足够量的相关数据来提供有意义结果。
- 然后，处理数据来识别分类特定关系相关的因素。
- 最后，分析数据并以清晰、简洁的方式展示结果，所以工作人员能采取适当措施。

在风险管理（Risk Management）中运用业务活动监控 BAM（Business Activity Monitoring）的实例之一就是防欺诈的信用卡交易。如果一项大笔的预付现金由信用卡支付，而这并不符合卡持有者一般的消费习惯，那么银行安全人员可以给卡主打电话并核实这笔交易是计划中的。另一个可疑行为实例是频繁地在一台自动取款机上以最大允许金额或者接近的金额重复提取现金。

商业智能 BI 与业务逻辑（活动）监控 BAM 比较如表 19-2 所示 ：

表19-2 商业智能BI与业务逻辑（活动）监控BAM比较

商业智能BI	业务逻辑（活动）监控BAM
基于历史数据与关键指标KPI	基于当前的数据与KPIs

商业智能BI	业务逻辑（活动）监控BAM
基于大量的成批数据	基于相对较小的实时数据
主要用于计划与分析用途	主要用于运营管理

在后面的章节中将讨论基于 BAM 思路上的监控体系的设计和实施案例。

表 19-3 是 BAM 的一些目标举例。

表19-3 BAM的一些目标

BAM监控	适用场景
业务数据异常监控	适用于监控业务数据是否正常
业务流量监控	用于监控流量是否正常
业务逻辑（活动）监控	适用于对各个业务处理节点或软件模块上的上的故障进行报警。 频繁的二进制软件更新，配置更新，流量变化，外界依赖的服务行为变化都有可能导致业务工作异常，但是系统层监控、流量监控不能报警或快速定位业务处理故障点

BAM 的监控要开发者需了解业务逻辑，数据流图等。同时各个业务软件在设计和实现时，需要提供有对业务对象在各个处理节点的日志信息。

19.4 客户的性能监控

客户的性能监控（Customers Performance Monitoring）是指从终端用户的使用角度来看，服务的性能是怎么样的。这里包括两个部分，互联网性能监控和用户体验监控

19.4.1 互联网的性能监控

1. 总体概述

互联网性能监控（Internet Performance Monitoring）是指针对一个企业产品运营所在 IDC 运行状态进行 7x24 的网络链路质量及性能的监控，通过监控数据分析反应 IDC 网络链路运行效率及客户使用企业产品的感知，从而对企业产品的网络运营的改进提供帮助。

为什么要做互联网监测呢？这与产品运营有什么关系呢，首先来看这样一组数据统计 ：

- 网站运营不佳将影响品牌和用户忠诚度。
  - 33% 的被调查者不满网站速度。
  - 28% 的人不满收到错误信息。
- 网站表现差更导致购物者一系列强烈反应。
  - 75% 不满者将不再访问网站。
  - 28% 的人对公司产生负面影响。
  - 27% 的人会将他 / 她的经历告诉周围的亲朋好友。
- 网站访问速度至关重要，影响网站客户的忠诚度。
  - 33% 的受访者希望网站有快的响应速度。
  - 具有两年以上网络经验的受访者中有 42% 的人持同样观点。

网站用户体验对于客户忠诚度、企业品牌及口碑都具有极大影响。

不仅仅是互联网网站运营型企业，其他的一些产品运营型企业也需要进行互联网的应用监控。监控的主要目的表现为以下两点 ：

- 了解本企业网络在运行状态。比如企业的网络运营商是否在某个事件点出现故障导致网络中断、延时等，如果有互联网监控，它能够在第一时间能通知网络工程师立即启动应急预案，确保网络可持续运行。
- 提高用户使用产品的感知。客户分布在全国甚至世界的各个角落，他们使用的网络情况千差万别。比如，客户因网络情况不好导致客户无法正常的使用服务，服务商是等着客户一次又一次的电话抱怨，还是积极主动的将他们的信息提前获取，然后，针对这些情况给客户出具具体的解决方案呢？答案是明确的，服务商最好是主动的是面对现实一个个的去解决这些问题。

2. 监控要点

互联网监测主要有以下几个指标 ：

（1）网络丢包率监控

网络丢包率是指测试中所丢失数据包数量占所发送数据包的比率，通常在吞吐量范围内测试。 丢包率主要与网络的流量及硬件设备，准确地说是与从用户计算机到产品运营服务器之间每段路由的网络拥塞程度及之间的路由交换设备好坏有关。由于交换机和路由器的处理能力有限，当网络流量过高来不及处理时就将一部分数据包丢弃造成丢包，另外如果之间的设备损坏也会造成丢包。由于 TCP/IP 网络能够自动实现重发，这样发生丢包后不断重发，将造成更大量的丢包。

（2）网络延时及抖动监控

网络延时指一个数据包从用户的计算机发送到产品运营服务器，然后再立即从服务器返回用户计算机的来回时间。通常使用网络管理工具 PING（Packet Internet Group Protocol）来测量网络延时。由于互联网络的复杂性、网络流量的动态变化和网络路由的动态选择，网络延时随时都在不停地变化，不停的变化称为网络延时抖动。网络延时和网络延时的抖动越小，那么网络的质量就越好。

3. 监控难点

（1）监控覆盖面有限，只能在少数的互联网节点进行必要地监控。

（2）问题定位比较困难，互联网的路由较复杂，具体在哪个路由节点出现的问题、出现的什么问题不能很快地定位。

19.4.2　用户体验的监控

1. 总体概述

传统的监控，只关注服务平台端（server end）的状态，重点确保服务端能够提供正常的服务，对于服务端之外一般不做监控，主要是因为服务端外的部分不受控制。比如，互联网的传输

质量，受各大运营商提供的基础网络限制、各个运营商之间互联互通、以及客户到电信运营商之间的接入带宽等等不可控的因素太多。还有用户端的 LAN 的状态、计算机的硬件配置，系统是否能正常工作等等。

在云服务的环境下，服务商需要提供给用户一个最优的用户体验，不管用户处于何种网络，以及何种地理位置，都能得到最好的服务，因此传统的监控已经不能满足需求，需要监控到每个用户使用云服务的情况，也就是用户体验的监控（user experience monitoring）。这包括 ：

（1）监控用户端（client end）到服务端（server end）的网络质量状况，当客户到某个云服务点出现网络问题，根据当时情况，可以调整客户到他最优的另外一个云服务点。

（2）监控用户端（client end）的使用体验，如用户的 LAN 的状况、client 软件的运行情况、客户使用模式等。监控体系可以根据收集到数据做用户行为分析，为以后产品的更新和重构提供基础。

2. 监控要点

根据云服务产品的特点，针对性设计出网络质量和业务特性的监控，这包括 ：

- 定义好那些关键的、需要被监控的点和指标。
- 确定这些指标的阀值。
- 开发相应的应用接口。

3. 监控难点

用户体验的监控是超出传统监控的范畴，因此用户体验指标的定义和传统监控如何结合，如何报警及显示，还有需要在云服务软件上提供接口等等，都需要研发和设计的巨大的投入。相应的协调资源等也都是巨大的考验。

- 用户体验监控很多操作或动作都会涉及到用户，有一些还需要用户参与，如果使用不善，不仅不能提高用户的用户体验，反而会引起用户反感。
- 监控用户到服务端的网络质量，或者业务的使用情况，这些会影响服务器的性能，如果使用不当，会造成服务性能低下，用户网络质量下降等。
- 上传 BUG 的日志，这会占用用户带宽，如果频繁地上传，上传日志过大，都会降低用户体验。

19.5　案例研究（1）：业务活动监控

这个案例的目的是讨论业务活动监控的具体设计要点和实施步骤。[1]

B 公司是在线的支付服务公司，对业务的可用性和可靠性要求非常高。B 公司的应用层的监控是按照业务活动监控（BAM）的思路进行的。这里通过平台的设计和实施两个方面来讨论。

19.5.1　背景介绍

B 公司已有的监控系统包括 ：

- 系统层监控：例如网络、存储、操作系统、数据库系统、第三方软件（例如MEMcached）等服务监控。
- 业务流量监控（usage monitoring）。
- 业务数据异常监控：例如发现网关商户通知加签后字符串为空，则报警。

在实现上，目前的监控功能主要是由以下几个方面组成的：

- 数据监控：即使用 SQL 语句定时执行的方式。可以利用数据来监控出当前是否正常。如收单过程中数据状态是否正常，支付完成时数据状态是否正常等等。如不符合报警规则的就以邮件形式由监控中心通知到相应责任人处理。
- 图形监控：可以对银行接口、网关产品、商户交易量，成功率等进行实时监控，另外也有对设备监控，如数据库延时等。但图形监控还是已监控交易量为主。
- 交易日报：由数据库工程师每日定时从数据库中获取交易数据，生成每日报表。

19.5.2　现有服务监控存在的问题

目前的问题主要有两个：(1）各个业务的监控是分散的，(2）业务层的监控能力不足，如图19-2 所示。

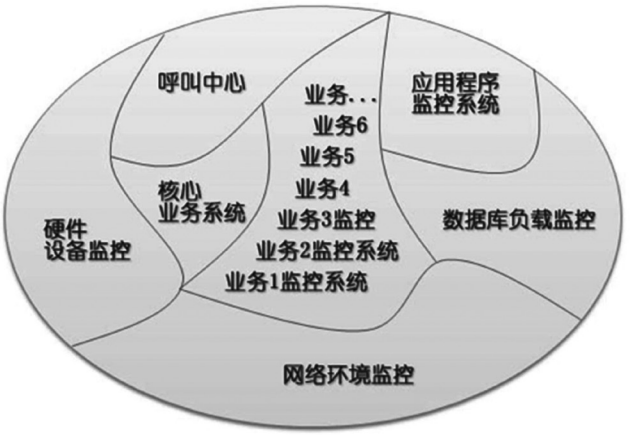


图19-2　重构前的分散的业务监控（案例）

B 公司已有的监控系统只能对某个特定业务的特定状态进行监控。现有监控系统分散在数百个应用程序和服务中，软件、硬件和网络的监控也都自成体系，相互独立不沟通，给维护和监控工作造成很大的困难。故障发生后无法迅速定位问题，特别是无法迅速定位业务层面上的故障点。

B 公司现在没有对业务拓扑逻辑、数据流图（logic topology/data flow）方面进行监控。频繁的二进制软件更新、配置更新、流量变化、外部依赖的服务行为变化等，都有可能导致业务处理节点工作异常，但是系统层监控、流量监控不能及时报警或快速定位业务处理故障点。

19.5.3　平台总体设计思路

1. 系统构思

整个监控体系的主要功能模块如图 19-3 所示：

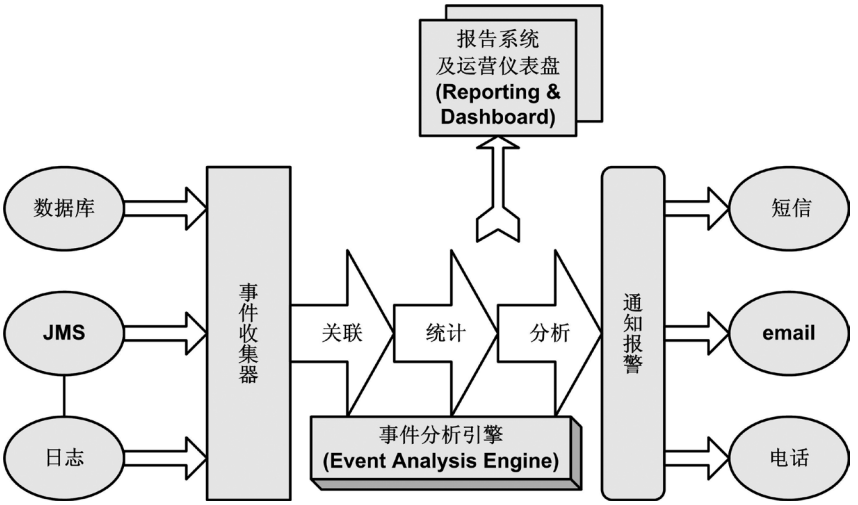


图19–3 新的整体架构图（案例）

2. 集中式的监控平台

业务活动监控系统采用中央节点控制，通过部署在“业务站点”的中间设备采集信息。“业务站点”可以是一个物理的区域，如机房，也可以是一个虚拟的业务集合点。所有的事件将传递到中央节点进行统一处理和输出。

这个设计的目的是使得（1）服务监控的一体化和标准化，（2）NOC（Network Operation Center）可以在单一系统中完成 7x24 的事件管理（Event Management）。

3. 控制流程和处理流程

在设计中，一般用流程图或活动图（Activity Diagram）来表示表示监控系统的主要控制流程和处理流程。

图 19-4 所示的数据流的流程图表示了应用服务监控的主要阶段，包括主要数据通路、数据流向、消息转换等。

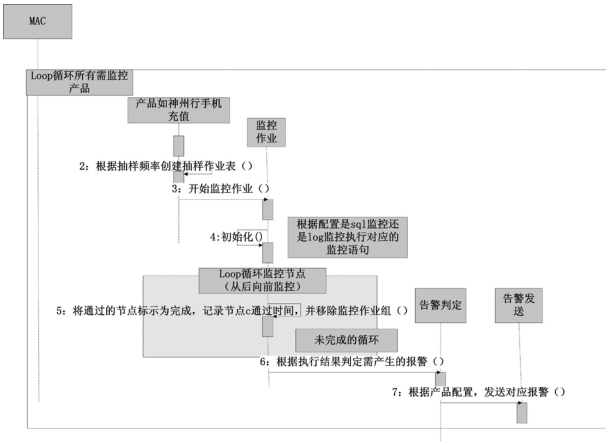


图19–4 控制流程图

抽样方法根据不同业务采用不同方法 ：

· 在单位时间（例如 1 分钟）抽取最大个数为配置值（例如 10）的订单 ；

· 在单位时间根据不同类型、不同商户级别抽取最大值为配置值的订单。

另外，对于流量较少的业务，根据完成支付事务的流量监控方法不能解决故障报警问题。例如正常情况下，单位时间完成支付事务流量为 0～5 个，流量为 0 时，很难判断是否故障已发生，但通过业务逻辑监控可报警该种故障。

19.5.4　业务逻辑监控的设计与实现

1. 确定业务逻辑

业务逻辑的确立是业务逻辑监控设计的第一步。这里以 B 公司的神州行手机充值为例说明。

B 公司的神州行手机的业务是为运营商完成一个网上接受支付的功能。其业务流程如图 19-5 所示 ：

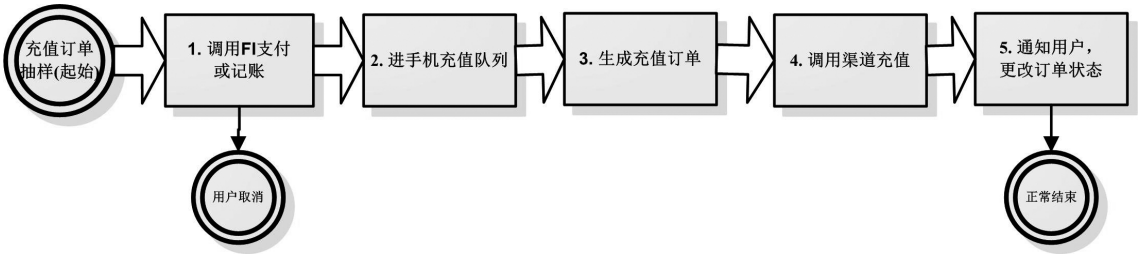


图19-5 手机充值业务流程逻辑图（案例）

2. 实施细节

以神州行电话充值卡做网上支付为例的详细设计 ：

（1）各个节点的状态判断

① 节点 1 判定 ：调用支付

• 正常状态 ：调用支付成功，当 num>=1 时候该节点为通过。

```
Select count(*) as num From szx.mobilebookingorder a Where
        a.orderid=? And a.status=111
/* 111: 调用支付成功 */
```

• 异常状态 ：调用支付平台失败。

```
Select orderid  From szx.mobilebookingorder  a Where
        a.orderid=?  and a.status=101
   /* Error code=101: 网关支付失败，可能原因 用户金额不足等 */
Select orderid  From szx.mobilebookingorder  a Where
        a.orderid=?   and a.status=112
/* Error code=102: 调用支付平台失败   */
```

② 节点 2 判定 ：进手机充值队列。

状态 ：

· 当手机充值队列的排队任务数 num>=1 时候该节点为通过。
· 超时查不到记录为异常。

③ 节点 3 判定 ：生成充值订单。

状态 ：

· 正常 ：生成充值订单当 num>=1 时候该节点为通过。
· 异常 ：超时查不到记录为异常 ：注普通订单 30 分钟可以生成订单（90% 以上）。慢充订单有可能 2 个小时才生成订单（10% 以下）。

④ 节点 4 判定 ：调用渠道充值。

状态 ：

• 当有效渠道充值数 num>=1 时候该节点为通过。
· 超时查不到记录为异常。

⑤ 节点 5 判定 ：通知并更新订单状态。

（2）报警策略

• 只对进入开始状态的事务进行判定。如果没有任何订单进入开始状态，若故障发生，该系统不支持报警，有系统级别监控系统负责。
· 根据处理时间分级别报警。对于一个抽样订单，如果从开始时间到进入结束节点的时间若超过配置时间，则触发不同级别的报警。
· 根据单位时间的成功率报警。由于依赖的服务其成功率不是百分百的，可根据不同级别设定的成功率报警。

（3）抽样业务对象运行状态判断

从效率考虑，对于一个业务对象，判断它当前的处理节点次序是从后往前判断，也就是从业务结束处理节点到起始节点，如果该业务对象正处于某一节点，则结束判断。

3. 监控实施的基本方法

业务拓扑逻辑的监控可以由以下两种方式来支持 ：

（1） 利用一个虚拟用户（机器人）模拟真实用户的行为，让该虚假用户的数据在业务拓扑逻辑图中遍历一遍，经过每一个环节时，其输出是可预见的。如果某一环节的输出并不是所期望的结果，则可判定该环节出现故障。例如为了监控邮件过滤系统，在单位时间，一个虚拟用户发出一组包括各种类型的邮件，不同类型的邮件过滤后的结果是可预见的。经过邮件解析、附件杀毒、邮件内容分类等环节，如果邮件过滤的结果在某一环节的输出并不是所期望的结果，则可判定该环节出现故障。对于第三方支付系统，利用机器人模拟真实用户的行为，这种方式不适用，因为本身支付系统有一特征是，反机器人，反伪造用户的。例如支付系统通过图形方式的动态验证码来阻止一已知用户名、密码的机器人来代替自然人支付 ；通过银行卡支付时银行对每一交易生成一动态密码，通过短信方式告知自然人。如果机器人能够破解图形方式的动态验证码，那么该支付系统应该改进识别难度，避免该安全漏洞。

（2） 基于事务日志（transaction log）的业务拓扑逻辑分析 ：对于金融服务系统，为了保证交易可跟踪，每一比交易，每一环节，存在事务日志记录。当一笔交易数据从一个处理节点流向另一处理节点时，事务日志记录着该笔交易数据的状态跳转。例如网关支付中逻辑图（图19-5）中的 1、2、3 环节。对于每一环节的跳转，如果事务日志在上一环节出现，但在下一环节消失了，说明下一环节的工作已出现故障。

19.6　案例研究（2）：大型监控体系扩展性设计

在大型云计算中心的监控中，监控平台本身的部署有两个挑战 ：

（1）网络架构复杂 ：各自服务运行在不同的网段上。

（2）监控平台要处理的数据量大，带来监控服务器（monitoring server）的负载高。

在这个案例中[2]，G 公司是通过分布式的监控服务器（distributed monitoring servers）的部署来解决这个问题。

19.6.1　背景介绍

G 公司是互联网服务型企业，业务分部在多个区域，服务器主机拥有近千台，拥有多条产品线，近有一万多个监控服务点。此案例主要以海量基础监控及内部业务监控使用市场上主流的 Nagios 监控工具为例，简要的描述如何进行 Nagios 监控的调优工作。此案例的其监控覆盖范围如下 ：

（1）基础部分 ：

- 虚拟机，操作系统负载、CPU、内存、磁盘利用率、I/O 读写、网卡流量。
- 宿主机，系统 CPU、内存及其他资源利用率。

（2）业务部分 ：业务逻辑、syslog、业务 log。

19.6.2　现有服务监控存在的问题

目前使用的监控采用 Nagios NRPE 与 Cacti 结合的方式，由于生产线的业务逐渐增多，其所要监控的服务也将增多，Nagios 监控点在多余 3,000 个后，其性能将会降低。一般 Nagios 监控时间间隔为每 5 分钟监控一次，随着服务的增加，其监控的间隔时间将会越来越大，导致生产线服务有问题的时候延缓告警邮件和短信的发送，影响故障处理的时间。现有监控的技术架构如图所示

19.6.3　分布式监控架构设计

为了适应批量应用服务的模式，监控的分布式也随之成为必要，大量的密集的服务器，和多区域的分布，成为监控的难题，传统的中央集中监控模式也捉襟见肘，越来越无法适应，分布式能很好地解决这一缺点，通过在每个区域部署一个监控采集点，然后向中央的监控传送监控数

据，中央监控汇聚监控结果统一展示和报警。这样既解决了中央监控因被监控服务器数据巨大，而监控系统自身无法承担负载，又解决了中央监控到各个区域的网络带宽，和由于专线或者互联网连接质量导致的监控不准确。

目前很多监控软件都支持分布式，大部分都是通过监控采集点 / 中央展示的方式，以 Nagios 为例，Nagios 的任务队列的监控任务，下发给 mod_gearman 模块，此模块接受到监控任务，将任务发送至 gearman 服务端，又 gearman 将任务转发至监控采集点 worker，worker 可以有多个，可形成分组或者负载均衡，worker 将监控到的结果又返回给 gearman，gearman 再返给 mod_gearman，Mod_gearman 将结果发送给 Nagios，最后由 Nagios 负责展示和报警。见下面的逻辑图（见图 19-6）：

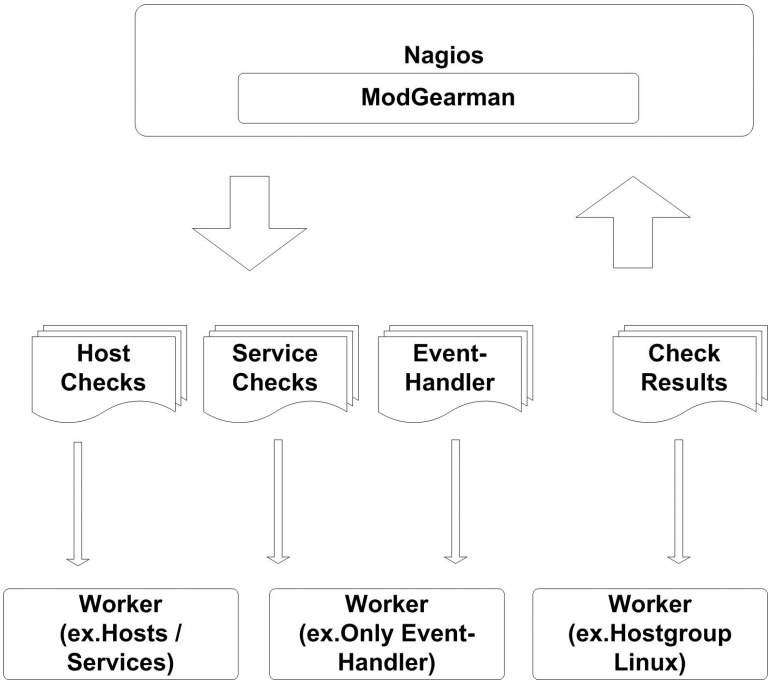


图19-6 分布式监控总体架构

19.6.4　分布式监控的实施

Nagios 分布式系统一共有三台服务器，安装 VMware，虚拟出多台虚拟机，这几台虚拟机共组成 Nagios 分布式系统。其中一台为 Nagios master，为中心服务器，Nagios 的所有配置在此服务器上，提供 Web 的展示，收集所有 worker 的检测结果 ；另一台为 Nagios slave，为 Nagios master 的备机和 master 之间每 5 分钟同步一次配置文件和 pnp 数据。具体架构设计如图 19-7 所示。

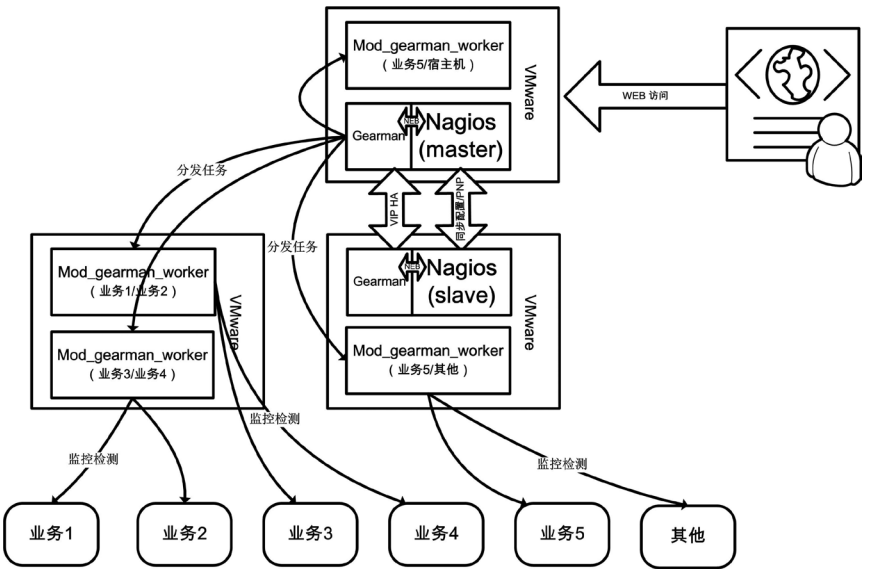


图19-7 分布式监控的实施

另外，worker 机器采用负载均衡，并采用分布式方案，其中每台上的业务类型不同，根据业务的特点选择指定的 worker 服务器。此种设计主要是既有冗余，将来为故障定位减少时间。

19.6.5　分布式监控的要点与难点

（1）要点

在设计中，要结合当前的监控的现状，需求，整体部署是否简单，以后维护和扩展是否方便等。在监控层级上，要考虑基础监控、互联网性能监控、虚拟化监控、业务监控、用户体验监控是否能满足，是否能很好地整合和管理等。

（2）难点

分布式监控，目前每个监控软件都有自己的分布式方案，方案的设计也都很有大的不同，需要选者一个适合自己的。分布式方案需要投入很大的资源取筛选，而且有些监控软件有好几套分布式方案。

例如像 Nagios，本身有三套分时方案，分别是 OSCP、DNX 和 Mod_gearman，OSCP 需要同步采集点和中央监控的配置，一旦配置不同步将造成监控误报或者无法监控等问题，而且每个监控采集点都是单点。而 DNX 由于代码维护较少，其稳定性也相对于 Mod_gearman 有很大的差距，很多 BUG 得不到及时的修复。而 Mod_gearman 可以解决前两者的缺陷。而那款分布式适合自己，需要看当前监控现状、已经业务的特点进行综合考量后才能决定。

19.7　本章总结

本章中，介绍了云服务的监控体系。监控的目标是要在客户发现问题之前，或至少同时，发

现服务的问题。

在基础架构的监控相对比较成熟的情况下，业务层的监控就成了各个互联网应用服务公司，包括云服务公司的难点。

业务逻辑监控系统的主要功能是能快速定位出业务故障点，发现问题是出现在哪个模块或者是数据在哪个模块的流动（data flow）中出现异常。

业务逻辑监控系统依赖产品的业务逻辑。

在本章的案例中描述了介绍了整个业务监控的设计思想和实施。

另外，如本章前面所讨论的，作为互联网上的应用，互联网的连接及其性能是另外一个需要监控的地方。在这方面，CISCO/WebEx 的 WeatherMap 和 GlobalWatch 就包括了用户从互联网的客户一端连接到云服务提供商的数据中心的应用平台上的应用层监控，达到了点到点的实时网络会议服务的性能监控。具体的可以查阅 CISCO/WebEx 的技术白皮书。

参考文献：

[1] 此案例研究选用了 B 公司的技术运营监控设计资料

[2] 此案例研究选用了创想空间通信公司的技术运营监控设计资料

第20章 云服务平台的质量工程

最早通过简单的手工检验来进行控制质量，发展到以统计学为基础的控制理论和控制技术，及后来的质量保证手段、全面质量管理思想等等，质量的管理水平不断得到提升。软件开发是以个人智力为基础的有组织的团队性生产活动，使软件质量变为一项复杂系统工程问题，我们必须用系统方法来研究。借助系统工程学、管理学等理论，把质量控制、质量保证和质量管理有效地集成在一起，形成现代软件质量工程体系，是当今质量管理的发展趋势，也是真正改善软件质量的最彻底、最有效的方法。

上个世纪末，软件正从产品的销售模式转化到在线服务的模式，即诞生了软件即服务（SaaS）模式，随后又发展到今天的云服务模式。这些给软件质量管理带来新的挑战，软件质量的管理不在局限于软件开发阶段，而是一直延伸到技术运营阶段，技术运营的质量管理也成了重中之重。

20.1 服务质量保证的基本原理

要建立软件服务质量工程体系，首先要了解什么是软件服务质量、传统的质量管理体系有哪些内涵，然后基于先进的质量管理思想，结合系统工程、软件工程等学科以建立现代的软件质量工程体系。传统的质量管理体系能够帮助组织增强顾客满意，鼓励组织分析顾客要求，规定相关的过程并使其持续受控，以实现顾客能接受的产品。质量管理体系能提供持续满足要求的产品，向顾客提供持续的满意服务，而且能提供持续改进的框架，以增加顾客和其他相关方满意的机会。从这个意义看，质量管理体系实际是质量管理过程成为一个持续改进的过程，这也是系统工程学的一个基本目标，通过设定顾客满意度作为管理体系的质量目标，顾客的需求则是系统的约束条件，对系统中的资源再分配、质量功能调节等，以便寻求质量管理体系越来越优化的结构和功能。为了使组织有效地运行这个持续改进的过程，必须识别和管理许多相互关联和相互作用的过程。由国际标准 ISO9000 或国内标准 GB/T19000 所表述的、以过程为基础的质量管理体系模式如图 20-1 所示。该图表明在向组织提供输入方面相关方起重要作用。监视相关方满意程度需要评价有关相关方感受的信息，这种信息可以表明其需求和期望已得到满足的程度。

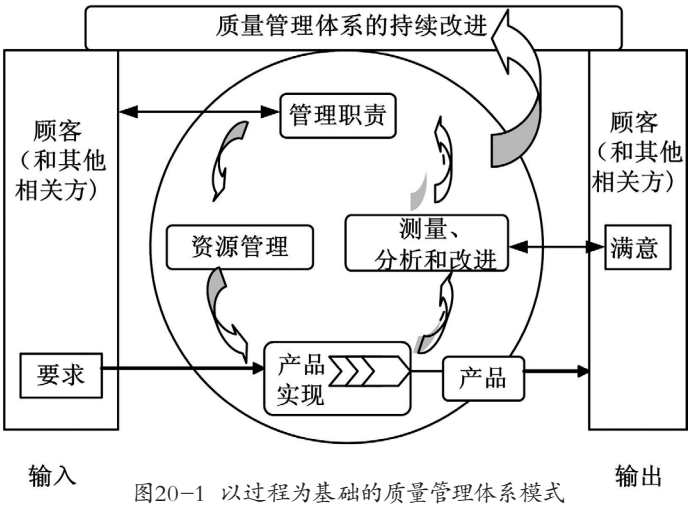


图20-1 以过程为基础的质量管理体系模式

采用上述方法的组织能对其过程能力和产品质量树立信心，为持续改进提供基础，从而增进顾客和其他相关方满意并使组织成功。建立和实施质量管理体系的方法，主要包括以下内容 ：

- 确定顾客和其他相关方的需求和期望 ；
- 建立组织的质量方针和质量目标 ；
- 确定实现质量目标必需的过程和职责 ；
- 确定和提供实现质量目标必需的资源 ；
- 规定测量每个过程的有效性和效率的方法 ；
- 应用这些测量方法确定每个过程的有效性和效率 ；
- 确定防止不合格并消除产生原因的措施 ；
- 建立和应用持续改进质量管理体系的过程。

20.1.1　软件服务质量

不管是软件产品，还是软件服务，都是为了满足客户或用户的某种需求，也就是为了让客户满意，这就是经常强调的产品质量或服务质量。质量是产品或服务所满足明示或暗示需求能力的特性和特征的集合，即满足用户明确需求的能力。质量高的软件应同时满足用户的需求和软件企业自身的需求。

- 满足外部用户的需求，就是要使软件系统能够满足用户在功能、易用性、可用性、可靠性和安全性等各个方面的要求或期望。
- 满足软件企业自身的需求，就是要降低软件系统的复杂性，具有可扩充性、移植性等，使系统更容易维护。对于云服务，软件质量需求的焦点在于系统的可用性（可靠性）、安全性和可维护性等。

软件产品，不仅要满足其运行时的用户需求，还要满足用户需求的变化，要对软件产品进行修改，增加新的功能、增强已有的功能或修正不正确的功能。软件产品的质量需要考虑其开发、运行、维护等不同方面的诉求。例如，对开发者具有重要意义的软件质量诉求是使产品易于修改、扩充、测试和验证，并易于移植到新的平台上，从而可以更及时地满足客户的需要。而当产品运行时，用户则希望容易安装、容易理解、操作简单方便、随时可用，而且系统运行要稳定可靠、快速响应，可立即提供结果，也经得起折腾，系统决不会崩溃。而在软件的维护时候，数据能够备份和恢复、功能容易扩充等。

软件质量指标是衡量那些可识别的软件质量特性的项目，有助于软件质量进行度量，选择软件工程方法来达到特定的质量目标。在一个理想的范围中，每一个系统总是最大限度地展示所有这些属性的可能价值。在 ANSI/IEEE 中提到以下了六个质量要素 ：

- 正确性（Correctness）：实现的功能达到设计规范，并满足用户需求的程度。
- 可靠性（Reliability）：规定的时间和条件下，仍能维持其性能水准的程度。
- 易用性（Usability）：用户掌握软件操作所要付出的时间及努力程度。
- 效率（Efficiency）：软件执行某项功能所需电脑资源（含时间）的有效程度。
- 可维护性（Maintainability）：当环境改变或软件发生错误时，执行修改或恢复所做努力的程度。
- 可移植性（Portability）：从一个系统 / 环境移到另一系统 / 环境的容易程度。根据这些软

件品质要素和前面的质量属性的分析，可以确定一系列的软件质量指标。根据这些软件质量指标来定义用户和开发者的目标，软件的设计者可以做出合适的选择。

（1）功能性的质量指标

- 功能的正确性（Correction）：系统功能和用户的实际需求、已定义的产品规范一致，没有出错，能正常运行。
- 功能的准确性（Accuracy）：系统所产生的结果在精度允许的误差范围之内（如±0.001）。为了达到这个精度，函数变量、数值算法、数值转化等都要保留足够的数位，如长整型、双精度的浮点运算等。
- 软件功能的完整性（Completeness）：所有功能及其定义应该清楚、可用，满足所需功能的每一个输入 / 输出数据项、功能、接口、文档等都已具备。

（2）可用性的质量指标

- 可操作性（Operability）：容易使用和操作系统，包括易理解的用户界面、大多数场合使用很少的敲键、适应一些特殊用户的可选项等。
- 通用性（Commonality）：数据显示、网络通讯接口和用户界面等都遵守已有的软件标准，包括采用单一模块来进行数据格式的处理、通讯接口等。这一质量指标也和可维护性、扩展性相关。
- 一致性（Consistency）：在软件开发整个生命周期中，建立和使用相同的标准，保证设计、实施的流程、方法等一致、保证全程变量、数据类型、出错处理等命名和使用的一致性。

（3）可靠性的质量指标

- 系统自我恢复能力（Autonomy）：当系统的某个功能失效发生时，系统在当前环境下能实现故障自动转移，重新自动配置、继续执行的能力。软件系统具有自我检测、容错、备份等机制，也尽量做到独立于硬件的编码、硬件设备之间的通讯协议独立等。
- 健壮性（Robustness）：各种恶劣环境（大数据量、大量用户同时访问）下系统能正常工作。
- 系统的分布性（Distributivity）：软件系统的某些子系统或部分被定位于不同的处理主机、存储设备上，提高系统的可靠性和性能、容量、可维护性和可扩展性。分布性包括网络结构分布性、系统接口分布性、系统功能分布性和数据分布性、用户操作分布性等。在不同设备上运行相同的关键功能或对数据的储存和复制，是解决单点失效问题的有效方法。

（4）性能的质量指标

- 有效性（Efficiency）：系统在通讯、处理、存储等方面占有很少资源或对所使用的资源进行了最优化。是不是每个通讯的字节、每个存储的数据、每个处理的步骤都是必要的？
- 安全管理 / 完整性（Safety management）：系统及其子系统（或任何部分）具有良好的安全管理，能防止不安全存取系统、防止数据丢失、防止病毒入侵等。完整性要求不能犯任何错误，即数据和访问必须通过特定的方法完全保护起来。在系统设计时候，能将系统的关键组件和非关键组件隔离开来，并对系统安全性、失效恢复等足够测试，是实

现完整性的有效方法。

- 易存取性（System Accessibility）：对系统的存取权限设置清楚，存取操作方便，存取操作有记录。易存取性和完整性相关性较大。

（5）可维护性的质量指标

- 模块化（Modularity）：是指将一个复杂的软件系统分解为分别命名并可寻址的、最小的耦合性、很强的凝聚性、结构化的组件，将模块综合起来又可以满足问题的需求的性质。
- 增强能力 / 灵活性（Augment ability）：容易为系统增加一个新功能或新的数据，而不需要进行大量的代码修改或设计修改。用户的需求肯定是要变化的，所以软件的设计事先就应该假定系统的功能是不完备的，留有接口或空间使系统有良好的增强能力。
- 可测试性（Testability）：测试软件组件或集成产品时查找缺陷的简易程度。如果产品中包含复杂的算法和逻辑，可测试性在系统设计中的实现就很重要。
- 可追溯性（Traceability）：对一个特殊需求容易找出相对应的代码，也容易找到特殊代码所实现的某个需求。可测试性和可追溯性对系统需求验证，也是重要的。
- 简单性（Simplicity）：系统设计、功能实现简单到直接了当、轮廓鲜明、清晰，如尽量少用公共数据块、分离输入源和输出结果、单元的单入单出、避免重复代码等。
- 自我描述性（Self Descriptiveness）：设计、代码容易被理解，包括每个功能如何被执行，即软件可以通过文档描述自己。自我描述性通过代码中有效的注释行、程序语言或设计语言等实现。
- 系统兼容性（System Compatibility）：系统之间操作协调，硬件、软件和外部系统可以象一个完整系统一样正常工作。
- 文档质量（Document Quality）：文档齐全、符合标准、遵守模板要求、逻辑清楚、描述准确、用词恰当、易理解、分类归档等。例如，当系统出错时，可以通过文档追溯到设计、程序某个准确位置，容易解决问题（Fix Bug）。用户的文档质量也有助于可用性质量指标。

（6）可移植性的质量指标

- 独立性（Independence）：系统不依赖于环境，也就是说系统不做修改或做很少的修改就可以运行在其他环境中。
- 可重用性（Reusability）：表明一个软件组件除了在最初开发的系统中使用之外，还可以在其他应用程序中使用的程度。可重用软件必须标准化、资料齐全、不依赖于特定的应用程序和运行环境，并具有一般性。
- 互操作性（Interoperability）：表明了产品与其他系统交换数据和服务的难易程度。
- 虚拟性（Virtuality）：通过相同的逻辑或虚拟接口代表不同的物理组件，这种代表性体现在软件用户接口、单元相互调用、软件和设备接口。有了很好虚拟性，可以不需要修改接口，就替换功能、设备。
- 一般性（Generality）：系统的某个单元具有一般性时，可以向多个功能提供服务，或可以被多个组件调用。

软件质量因素很多，而且对软件质量影响的程度、深度是不一样的。比如“正确性与精确性”应该排在这些质量因素的第一位，因为软件运行首先要能正常运行、结果正确，才能为用户

服务，才能向用户提供最基本的服务。否则，软件就没有价值，给用户造成损失，更谈不上性能、可靠性、兼容性等等。即使一个软件百分之百地实现了需求规格说明书规定的功能特性，还不能断定这个软件的正确性与精确性，因为需求分析还可能会出错。所以，软件的正确性与精确性需要用户的验证、确认。

其次，对软件产品质量影响的因素是易用性和性能，性能和易用性在一定程度上紧密相关。在保证软件的正确性与精确性之后，就要保证用户能方便、快捷使用产品。不仅产品要好用，而且系统反应、速度要快，去赢得更多的用户。产品只有在越来越多的用户使用下，才能得到用户的大量、足够的反馈，才知道用户的需求，从而软件产品才能不断改进、提高，成为优秀的系统。

再者，对软件产品质量影响的因素是容错性和可靠性的设计。从某种意义上说，完全消灭软件中的缺陷几乎不可能，所以不得不假定系统存在着一定的不正确与不精确的因素，然后设计相应的措施，来保证这些因素的影响下，系统还是可靠、安全的，即容错性设计。

在软件在产品修改、移植方面的质量影响因素，最重要的是模块化或结构化设计、面向对象设计，其次包括软件的程序的可读性、简洁性、复用性、独立性等。

20.1.2　软件过程质量

相对软件产品来说，影响软件产品过程的质量因素更多，也比较复杂，这里从开发的整个过程包括计划、设计、实施和维护等分别做简单地介绍。

在软件项目计划过程中，要完成需求分析、软件产品特性、规格的定义、项目计划书（包括软件质量计划、测试计划等）等工作，这些工作相对复杂，工作需要做细、做到位，其影响因素很多，其影响这个过程的主要因素有 ：

- 和客户的沟通能力 ：良好、充分地和客户沟通决定需求分析的结果。
- 软件产品特性定义的方法 ：分析的结果还需要通过一种有效、清晰的方式来描述。
- 项目计划策略 ：策略会影响质量计划、测试计划等。
- 评审的流程、范围、方式和程度 ：主要对软件需求分析书、计划书的讨论、审查的力度。
- 协同工作流程。
- 合同和用户管理流程和方法。
- 文档编写、管理等的规范和流程。

在软件项目设计过程中，要完成系统的设计、程序设计、测试用例的设计等任务，设计相对来说是一个纯技术工作，相对封闭，与客户的关系较小，其结果取决于软件产品质量指标的定义。软件质量指标，尤其在系统构架、模块及其接口设计、可靠性设计、兼容性设计、界面设计等方面的要求，是否得到很好的理解、正确的实现、全面地贯彻等，直接关系到这个过程的质量，其影响因素有 ：

- 软件产品指标的定义和解释。
- 设计流程 , 包括知识交换、结果评审等流程。
- 设计标准改进流程。
- 协同工作流程。

· 文档编写、管理等的规范和流程。

在软件项目实施过程中，要完成系统的编程、测试脚本开发、单元测试、集成和系统测试、用例的设计等任务。理论上，如果计划、设计过程执行得很好，实施相对来说比较容易，如果按照计划严格执行，和良好的变更控制（Change Control）流程结合起来，结果会更理想。这个过程的质量影响因素有 ：

- 变更控制流程。
- 执行过程跟踪方法、流程和相适应的系统。
- 缺陷处理流程。
- 文档编写、管理等的规范和流程。

软件维护是软件过程中的很重要一个过程，通过不断改进、增强来实现软件的成熟、稳定和可靠。这个过程的质量影响因素有 ：

- 变更控制流程。
- 用户反馈、相应处理机制。
- 回归测试流程。

软件商业环境也可以看作软件过程的一个扩展，也可以看作软件产品的扩展。软件产品的一些特性会直接在商业环境中体现出来，如产品界面设计上循序渐进的变化容易被客户接受，而跳跃式、突破式的变化不被接受。产品的定义还要考虑对市场、销售的支持。软件质量在这方面影响的质量因素有 ：

- 软件改进的策略，如是否遵守持续不断、逐步改进的原则？
- 产品开发模式，是否很好采用增量模式，保证产品发布的最佳时机？
- 市场定位，不同的软件产品在用户群定义上是否有区别、有没有冲突？
- 产品标准，是否符合国际标准和业界标准？是否引导业界的进步？
- 文档内容和形式，是否通过 Flash、视频动画教学内容帮助客户的自我培训？是否通过在线帮助有助于客户支持？
- 软件的后续服务模式。

20.1.3　质量管理体系的构成

当试图用系统工程学的方法来进一步分析软件质量的管理体系、而建立一套现代的软件质量工程体系之时，首先就必须将软件质量管理作为一个系统来管理，也就是运用系统科学的方法，通过有关质量的各种信息反馈与调控，对软件质量进行全面、综合的系统性管理，包括软件质量计划、组织、协调、控制，以实现软件质量目标。有关软件质量的各种信息，来源于软件质量属性及其变化、客户对软件的需求及其变化、软件技术现状及其发展趋势等，这些信息被归纳为软件的质量指标、影响因素，是软件质量工程体系的输入。对这些系统的输入进行加工、处理，是通过软件质量的计划、实施、控制、调整等来实现的，其达到的质量目标就是系统的输出结果。

从系统方法论来说，系统工程学是系统的结构方法、功能方法和历史方法的统一。也就是说，软件质量工程体系，不仅体现在对质量系统的结构划分、结构分析、质量功能展开上，而且基于质量的历史数据，可以建立质量的预测或评估模型，包括软件的可靠性评估模型，从而实现软件质量的可预见性，也可以进行有效地软件质量风险的控制和管理。

如果质量目标不可验证，那么就说不清是否达到这些目标。在合适的地方为每一个属性或目标指定级别或测量单位，以及最大和最小值。如果不能定量地确定某些对项目很重要的属性，那么至少应该确定其优先级。软件质量度量方法的提出，就是为了解决这一软件质量需求。

概括起来，如图 20-2 所示，从系统工程的角度来描述质量管理体系，软件质量工程体系的思想是从系统工程学、软件工程理论出发，沿着逻辑推理的路径，对软件质量的客户需求、影响软件的质量因素、质量功能结构、问题根源等进行分析，以建立积极的质量文化、构造软件质量模型，基于这些模型研究相应的软件质量标准和软件质量管理规范，并配以相对应的质量分析技术、工具等，把质量控制、质量保证和质量管理有效地集成在一起，降低质量成本和质量风险，从而系统地解决软件质量问题，形成现代软件质量工程体系。

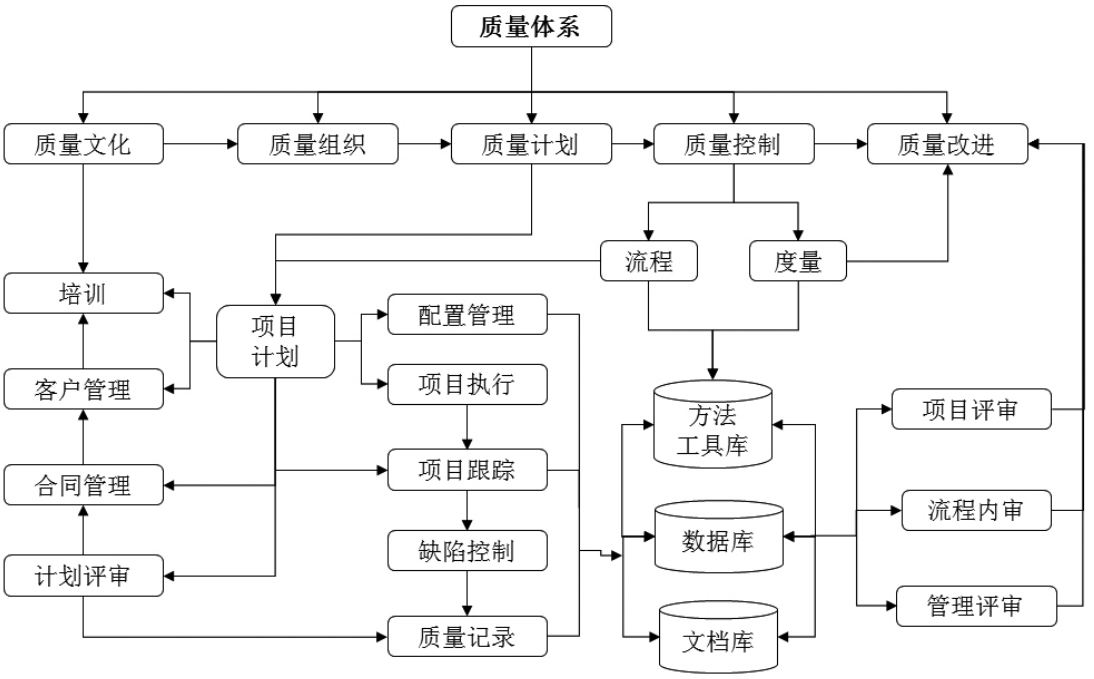


图20-2 从系统工程的角度来描述质量管理体系

现代软件质量工程体系，通过相应的系统方法，对软件质量管理体系进行实施并有效地控制、优化，而形成优化的质量结构和组织功能体系，最终达成预期的质量目标，即按时按量地、高质量地完成软件项目。软件质量工程体系揭示了软件质量计划、质量标准、质量控制、质量保证之间的关系，从对软件质量指标分析开始，直至客户满意度，其内容丰富，其构成的层次清楚，共分为 5 层如图 20-3 所示 ：

（1） 软件质量指标、软件质量影响因素
（2） 软件质量模型，软件质量度量
（3） 软件质量标准和规范，并与之配套的培训体系、技术、工具、模板等
（4） 软件质量方针、软件质量控制、软件质量保证和软件质量管理
（5） 软件质量成本的控制、软件质量风险的控制、客户满意度

软件质量指标是衡量软件质量的标准，有了这些指标，就容易判断软件质量的程度。接着就

可以找出影响这些指标的因素，也就是确定了软件质量的影响因素。对每一个软件质量的影响因素，都可以试图找出对策，降低某个因素的消极影响或提高某个因素的几积极影响。当然，这种对策也是不容易的，因为这些因素之间还存在相互影响。所以还得借助软件质量模型，才可以对质量问题分析透彻，并解决问题。

软件质量标准和规范，是相对软件质量模型而存在的。软件的管理工作，就是在软件质量标准和规范指导下，涵盖了软件质量计划、质量风险管理、质量成本控制和质量计划的实施等内容。质量计划的制定又受质量文化影响、质量方针指导，通过对影响质量各种因素的分析，了解可能存在的质量风险，从而加以回避、控制。通过对软件产品、过程的测量和质量的度量，不断改进软件开发过程，以达到软件质量预先设定的规程，可以较合理地达到目标。

软件质量工程规范，规定了一系列的质量活动，这些活动又得遵守相应的流程，也就是说，软件质量工程规范约束了软件开发人员的行为模式，软件开发人员的行为通过一套规范进行有效的控制。

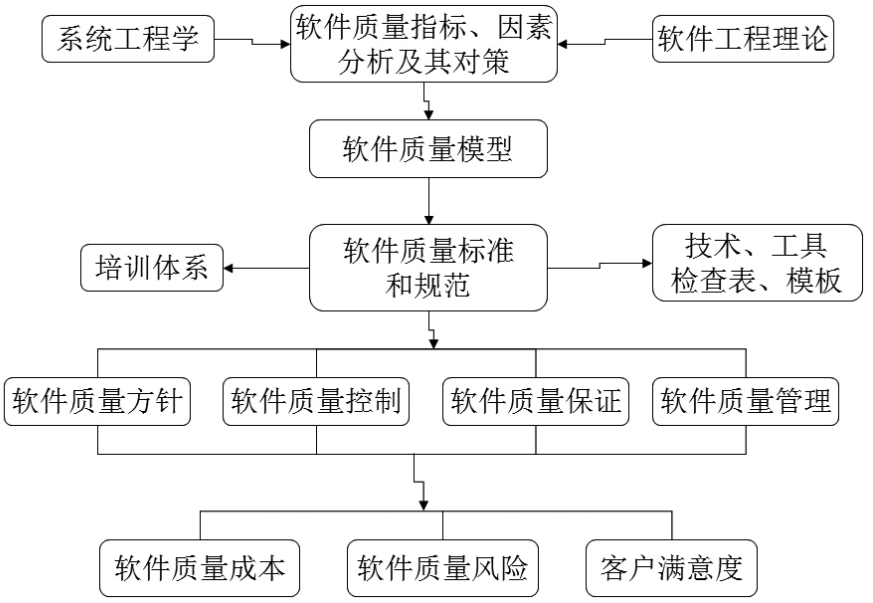


图20–3 软件质量工程体系的构成

20.1.4　软件质量控制

谈到软件质量工作时，人们经常会提及到软件质量控制、质量保证和质量管理，的确，软件质量控制、质量保证和质量管理是代表软件质量工作中的不同层次内容 ：

- 软件质量控制（SQC，Software Quality Control）是科学地测量过程状态的基本的方法。就像汽车表盘上的仪器，可以了解行驶中的转速、速度、油量等。
- 软件质量保证（SQA，Software Quality Assurance）则是过程和程序的参考与指南的集合。ISO 9000 是其中的一种，就像汽车中的用户手册。
- 软件质量管理（SQM，Software Quality Management）才是操作的誓学，教你如何驾车，建立质量文化和管理思想。

为了更容易理解软件质量工作层次时，可以从另一个方面简单地阐述软件质量管理的四个层次为 ：

（1） 检查，通过检验保证产品的质量，符合规格的软件产品为合格品，不符合规格的产品为次品，次品不能出售。这个层次的特点是独立的质量工作，质量是质量部门的事，是检验员的事。检验产品只是判断产品质量，不检验工艺流程、设计、服务等，不能提高产品质量。这个层次是初期阶段，相当于 “ 软件测试——早期的软件质量控制 ”；

（2） 保证，质量目标通过软件开发部门来实现，开始定义软件质量目标、质量计划，保证软件开发流程合理性、流畅性和稳定性。但软件度量工作很少，软件客户服务质量还不明确，设计质量不明确。相当于初期的 “ 软件质量保证 ”；

（3） 预防，软件质量以预防为主，以过程管理为重，把质量的保证工作重点放在过程管理上，从软件产品需求分析、设计开始，就引入预防思想，面向客户特征，大大降低低质量的成本，相当于成熟的 “ 软件质量保证 ”；

（4） 完美，以客户为中心，全员参与，追求卓越，相当于 “ 全面软件质量管理 ”。

在质量控制、质量保证和质量管理基础之上，是质量方针，即在质量方针指导下，质量管理发挥指挥和控制组织的质量活动，协调质量的各项工作，包括质量控制、质量保证和质量改进。

质量控制是质量管理的一部分，致力于满足质量要求。作为质量管理的一部分，质量控制适用于对组织任何质量的控制，不仅仅限于生产领域，还适用于产品的设计、生产原料的采购、服务的提供、市场营销、人力资源的配置，涉及组织内几乎所有活动。

早在 20 世纪 20 年代，美国贝尔电话实验室成立了两个研究质量的课题组，其一为过程控制组，学术领导人是美国统计应用专家休哈特 ；另一为产品控制组，学术领导人为道奇（H. F. Dodge）。通过研究，休哈特提出了统计过程控制的概念与实施方法，最为突出的是提出了过程控制理论以及控制过程的具体工具——控制图，现今统称之为 SPC（Statistical Process Control）。道奇与罗米格（H. G. Romig）则提出了抽样检验理论和抽样检验表。这两个研究组的研究成果影响深远，休哈特与道奇成为统计质量控制的奠基人。

统计过程控制是一项建立在统计学原理基础之上的过程性能及其波动的分析与监控技术，从其诞生至今，经过 80 多年的不断发展与完善，已经从最初的结果检验到今天的过程质量控制，从最初的仅仅应用于军事工业部门，发展到今天被广泛应用于社会经济生活的各个领域。由于统计过程控制技术对于分析和监控过程性能及其波动非常有效，它已成为现代质量管理技术中的重要组成部分。

质量控制的目的是保证质量，满足要求。为此，要解决要求（标准）是什么、如何实现（过程）、需要对哪些进行控制等问题。质量控制是一个设定标准（根据质量要求）、测量结果，判定是否达到了预期要求，对质量问题采取措施进行补救并防止再发生的过程，质量控制已不再仅仅是检验，而更多地倾向于确保生产出来的产品满足要求的过程控制。

20.1.5　软件质量保证

质量保证是质量管理的一部分，是为保护产品和服务充分满足消费者要求的质量而进行的有计划有组织的活动，致力于提供对满足质量要求的信任。组织规定的质量要求，包括产品的、过

程的和体系的要求，必须完全反映顾客的需求，才能给顾客以足够的信任。“ 帮助建立对质量的信任 ” 是质量保证的核心，可分为内部和外部两种 ：

- 内部质量保证是组织向自己的管理者提供信任 ；
- 外部质量保证是组织向外部客户或其他方提供信任。

质量保证定义的关键词是 “ 信任 ”，对达到预期质量要求的能力提供足够的信任。这种信任不是买到不合格产品以后保修、保换、保退，而是在顾客接受产品或服务之前就建立起来的，如果顾客对供方没有这种信任，则不会与之签订协议。质量保证要求，即顾客对供方的质量体系要求往往需要证实，以使顾客具有足够的信任。证实的方法可包括 ：供方的合格声名 ；提供形成文件的基本证据（如质量手册，第三方的检验报告）；提供经国家认证机构出具的认证证据（如质量体系认证证书或名录）等。

从管理功能看，质量保证着重内部复审、评审等，包括监视和改善过程、确保任何经过认可的标准和步骤都被遵循，保证问题能被及时发现和处理。质量保证的工作对象是产品和其开发全过程的行为。从项目一开始，质量保证人员就介入计划、标准、流程的制定 ；通过这种参与，有助于满足产品的实际需求和能对整个产品生命周期的开发过程进行有效地检查、审计，并向最高管理层提供产品及其过程的可视性。

在 CMM（能力成熟度模型，Capability Maturity Model SM）中，软件质量保证是其等级 2 的一个关键过程域 KPA（Key Process Area），软件质量保证被定义为 ：从事复审 / 审查（Review）和内审 / 检查（Audit）软件产品和活动，以验证这些内容是否遵守已适用的过程和标准，并向软件项目和相应的管理人员提交复审和内审的结果。CMM 同样清楚地告诉我们，其复审或内审的对象不只是产品，还包括开发产品的流程，而软件质量保证的活动被分为两类 ：

- 复审（Review）：在软件生命周期每个阶段结束之前，都正式用结束标准对该阶段生产出的软件配置成分进行严格的技术审查，如需求分析人员、设计人员、开发人员和测试人员一起审查 “ 产品设计规格说明书 ”、“ 测试计划 ” 等 ；
- 内审（Audit）：部门内部审查自己的工作，或由一个独立部门审查其他各部门的工作，以检查组织内部是否遵守已有的模板、规则、流程等。

基于软件系统及其用户的需求，包括特定应用环境的需要，确定每一个质量要素的各个特征的定性描述或数量指标，包括功能性、适用性、可靠性、安全性等的具体要求。再根据所采用的软件开发模型和开发阶段的定义，把各个质量要素及其子特征分解到各个阶段的开发活动、阶段产品上去，并给出相应的度量和验证方法。复审或内审就是为了达到事先定义的质量标准，确保所有软件开发活动符合有关的要求、规章和约束。软件质量保证过程的活动形式主要有 ：

- 建立软件质量保证活动的实体。
- 制订软件质量保证计划。
- 坚持各阶段的评审和审计，并跟踪其结果作合适处理。
- 监控软件产品的质量。
- 采集软件质量保证活动的数据。
- 对采集到的数据进行分析、评估。

质量管理体系的建立和运行是质量保证的基础和前提，质量管理体系将所有（包括技术、管理和人员方面的）影响质量的因素，都采取了有效的方法进行控制，因而具有减少、消除、预防不合格的机制。

经过长期的发展和演变，软件质量控制和质量保证的思想和方法越来越融合在一起，都强调活动的过程性和预防的必要性，最终保证产品的质量。

20.1.6　软件质量改进

质量改进是质量管理的一部分，是不断为改进软件开发过程、产品和服务的持续过程。同时，为确保有效性、效率或可追溯性，组织应注意识别需要改进的项目和关键质量要求，考虑改进所需的过程，以增强组织体系、改进过程和产品并提高满足要求的能力。

在质量改进工作中，有许多模型，包括 PDCA 模型、PEIS 模型、6-Sigma 模型的 DMAIC、CMM 模型有关质量改进部分、SPICE 模型等。这里通过介绍 IDEAL 模型，来了解质量改进的过程。IDEAL 模型将质量改进过程分为初始化、诊断、建立、行动和学习等五个阶段。该模型和软件工程研究所（SEI）提出的 IDEAL 模型有类似和借鉴的地方。IDEAL 模型由五个阶段组成，如图 20-4 所示 ：

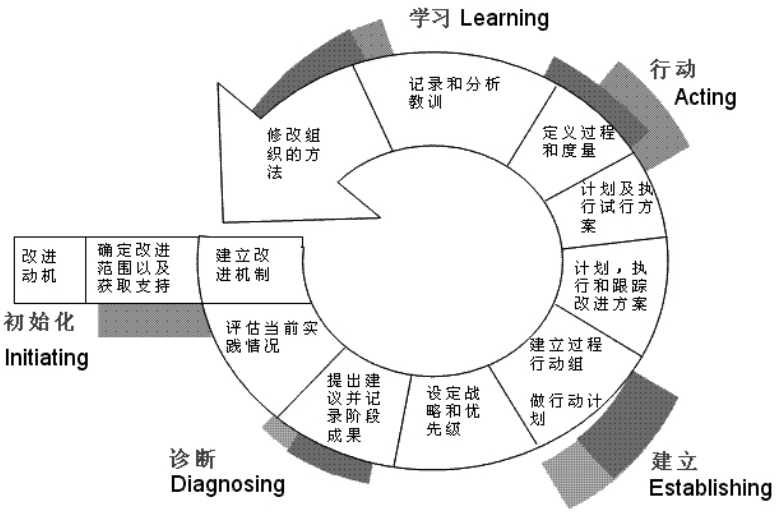


图20-4 IDEAL模型用于质量改进

- I — Initiating（初始化），为成功地改进质量工作奠定基础。
- D— Diagnosing（诊断），确定现状与质量目标之间的差距。
- E— Establishing（建立），计划如何达到质量目标。
- A— Acting（行动），根据计划开展质量工作。
- L— Learning（学习），从经验中学习，以提高在将来采用新技术的能力。

20.2　质量保证活动

质量保证（QA）过程是确保过程或项目中的工作产品以及活动都遵循相应的标准和规范的活动过程，对项目产品和服务的质量加以管理，使之软件产品或服务能真正满足客户的需求，从而获得最大的客户满意度。质量保证和管理在项目以及组织层次上建立对产品和过程质量管理的关注，涵盖产品质量、过程质量和质量体系等，并需要组织上的全力支持。其基本目标有 ：

- 以客户的质量需求为基础，在整个软件项目生命周期内不同的检查点确立相应的质量目

标。

- 定义质量度量标准、方法与工具，并实时检查，以便在项目生命周期中的不同检查点评估有关内容是否达到了相应的质量目标。
- 制定出相应的计划与进度表，确定质量保证和管理活动所需的资源、组织及其组织成员的职责。
- 从软件工程的需要出发，系统地指出良好的实践经验，并能集成到生命周期的过程模型中。
- 实施已经定义的质量活动并监控这些质量活动的执行情况。
- 当未能达到质量目标时，要及时采取相应的纠正措施。

软件的质量是软件开发各个阶段质量的综合反映，因此软件的质量保证贯穿整个软件开发周期（如图 20-5 所示）。为了更好地保证软件产品质量，首先需要制定项目的质量计划 ；然后，在软件开发的过程中，需要进行技术评审和软件测试，并进行缺陷跟踪 ；最后对整个过程进行检查，并进行有效的过程改进，以便在以后的项目中进一步提高软件质量。

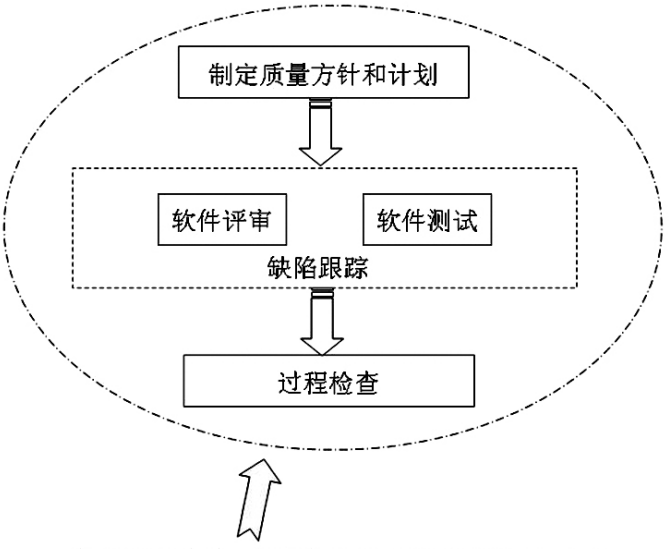


图20-5 软件质量保证过程

20.2.1　验证与确认

验证（verification）过程是依据实现的需求定义和产品规范、确定某项活动的软件产品是否满足所给定或所施加的要求和条件的过程。一般根据软件项目需求，按不同深度确定验证软件产品所需的活动，包括分析、评审和测试，其执行具有不同程度的独立性。为了节约费用和有效进行，验证活动应尽早与采用它的过程（如软件获取、开发、运行或维护）相结合。该过程的成功实施期望带来如下的结果 ：

- 根据工作产品所制定的规范（如产品规格说明书）实施必要的检验活动（软件测试）；
- 有效地发现各类阶段性产品所存在的缺陷，并跟踪和消除缺陷。

确认（validation）过程是确定最终的、已建成的系统或软件产品是否满足特定的用户需求的

过程，集中判断产品中所实现的功能、特性是否满足客户的实际需求。确认过程和验证过程，构成软件测试缺一不可的组成部分，也可以看作是质量保证活动的重要支持手段。确认也应该尽量在早期阶段进行，如阶段性产品的确认活动。确认，与验证相似，也具有不同程度的独立性。该过程的成功实施期望带来如下的结果 ：

- 根据客户实际需求，确认所有工作产品相应的质量准则，并实施必需的确认活动。
- 提供有关证据，以证明开发出的工作产品满足或适合指定的需求。

20.2.2　评审

软件评审（Software Review :Review 常被翻译为评审，也可以翻译为审查、复审、复查等）的重要目的就是通过软件评审尽早地发现产品中的缺陷，因此软件评审可以看作软件测试的有机组成部分，两者之间有着密不可分的关系。通过软件评审，可以更早地发现需求工程、软件设计等各个方面的问题，大大减少大量的后期返工，将质量成本从昂贵的后期返工转化为前期的缺陷发现。通过评审，还可以将问题记录下来，使其具有可追溯性，找出问题产生的根本原因，在将来的项目开发中进一步减少缺陷，有利于软件质量的提高。

那什么是软件评审呢？根据 IEEE STD 1028-1988 的定义 ：评审是对软件元素或者项目状态的一种评估手段，以确定其是否与计划的结果保持一致，并使其得到改进。检验工作产品是否正确地满足了以往工作产品中建立的规范，如需求或设计文档是否符合所定义的模板要求，各项内容是否清楚、一致。软件评审的对象有很多种，主要分为管理评审、技术评审、文档评审和流程评审。对于软件测试，应该包含了技术评审和文档评审，而管理评审和流程评审则属于软件质量保证的组织和过程管理的活动内容。

技术评审是对产品以及各阶段的输出内容进行评估，其主要目的是揭示软件在逻辑、执行以及功能和函数上的错误，以验证软件是否符合需求，确保需求说明、设计说明等符合系统的要求。技术评审后需要以书面的形式对评审结果进行总结。

- 技术评审会分为正式和非正式两种，通常有技术负责人（技术骨干）制定详细的评审计划，包括评审时间、地点以及定义所需的输入文件。
- 在评审过程中，评审小组会按照评审检查单对需要评审的内容进行逐项检查，确定每项的状态，检查项状态可以被标记为合格、不合格、待定、不适用等。
- 评审结束后，评审小组需要列出存在的问题，建议措施，责任人等，并完成最终的《技术评审报告》。

文档评审往往分为格式评审和内容评审。所谓的格式评审，是检查文档格式是否满足标准，而内容评审则是从一致性、可测试性等方面进行检查。文档评审，往往要完成《产品 / 市场需求说明书》、《功能说明书》、《系统架构设计说明书》、《详细设计说明书》、《测试计划》和《测试用例》等的评审。评审过程中要把握正确性、完整性、一致性、有效性、易测性、易理解性等。

评审的方法很多，有正式，非正式的。图 20-6 就是从非正式到正式的各种评审方法。

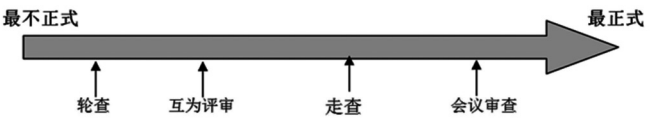


图20-6 各种评审方法

（1）轮查（Passround）

轮查又称为分配审查方法。作者将需要评审的内容发送给各位评审员，并收集他们的反馈意见，但轮查的反馈往往不太及时。

（2）互为评审或同行评审（Peer Review）

互为评审是由软件开发者的同事对软件产品进行系统的检查，来发现错误和检查修改过的区域，是一种常用的评审方式，实施起来比较简单，例如程序员甲写的程序让程序员乙复查或复审，程序员乙写的程序让程序员甲复查。 目的都一样，尽早、有效地清楚软件阶段性产品中的错误。然而，评审的过程还不够完善，特别是评审后期的问题跟踪和分析往往被简化和忽略。

（3）走查（Walkthrough）

走查也属于一种非正式的评审方法，它在软件企业中被广泛使用，一般来说，初级工程师（如新招的毕业生）的工作成果需要进行全面审查，一般由其主管或一位资深工程师来评审。只有当这样的走查进行几次以后，其主管对该员工建立了信心，就可以不做走查，而实行互为评审的活动。走查的目的也是希望参与评审的其他同事可以发现产品中的错误，对产品了解，并对模块的功能和实现等达成一致意见。

然而，由于作者的主导性也使得缺陷发现的效果并不理想。因为评审者事先对产品的了解性不够，导致在走查过程中可能曲解作者提供的信息，并假设作者是正确的。评审员对于作者实现方法的合理性等很容易保持沉默，因为并不确信作者的方法存在问题。

（4）评审会议（Inspection）

评审会议更为严格，是最为系统化的评审方法，其过程包含了 ：制定计划、准备和组织会议、跟踪和分析审查结果等，见下一小节详细描述。

20.2.3　正式评审会议

事实上，大量的实践已经证明了软件评审是使产品达到用户要求的一项十分有意义的活动，的确是软件过程中软件质量保证的一个重要手段。然而，在进入正式的评审流程之前，需要进行是否准入的判断，只有满足评审的入口条件时，才能进入软件评审过程。评审的入口条件包含了如下内容 ：

（1） 创建者为待评审的工作产品选择了评审方法。
（2） 准备好所有必需的支持文档。
（3） 创建者陈述了该次评审的目标。
（4） 评审者接受了同行评审过程的培训。
（5） 为待评审的文档分配了版本号。所有页面都标明了页号和行号。文档经过了拼写错误检查。
（6） 为待评审的源代码分配了版本号。代码清单标明了行号和页码。代码已经使用项目标准编译转换器编译过，并且没有错误和警告信息。使用代码分析器发现的错误已经被改正。
（7） 对于二次评审，前一次评审中发现的所有问题都已经解决。

（8） 满足所有针对特定的工作产品定义的附加入口条件。

进入评审流程后，主要分计划、总体会议、准备会议、审核会议、和返工 / 跟踪 5 个步骤，其流程如图 20-7 所示。

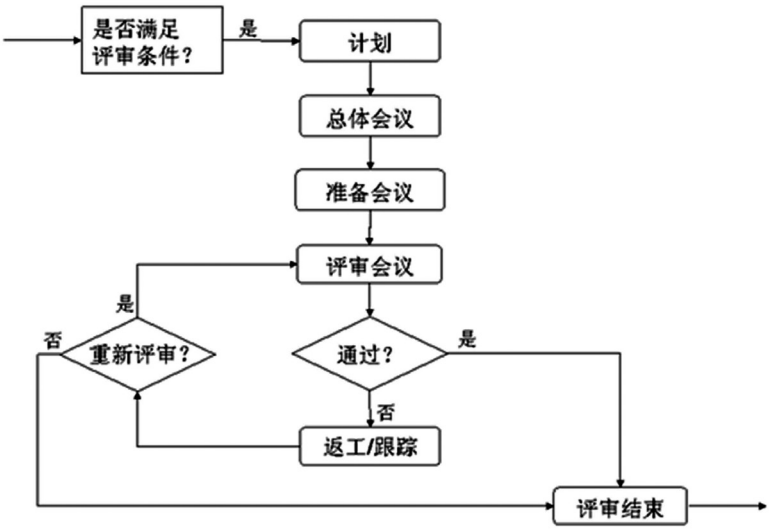


图20-7 软件正式会议评审流程

（1）计划

计划主要是确定即将送审的工作产品，并做好评审时间表，这一阶段有如下几个主要任务 ：

- 将需要评审的工作产品和支持文档，如规范、以往文档和相关测试文档交给评审负责人。
- 确定工作产品是否满足评审入口条件。
- 根据工作产品的规模和复杂度确定需要多少次评审会议。
- 选择检查者，并为其分配角色。确认评审员同意参加评审。
- 确定是否需要一次评审总体会议。
- 安排评审会议或者还有评审说明会议的时间，并发出会议通知。

（2）总体会议

总体会议根据实际情况是可选择的。如果评审员对于即将评审的工作产品已经比较熟悉，那么可以省略总体会议，直接开始准备正式的评审会议。如果之前评审员不是十分熟悉工作产品，可以召开总体会议以便对产品有一个总体介绍。在总体会议上，作者需要向评审小组的其他成员描述工作产品的重要特征，并陈述评审目标。同时，评审人员评估工作产品的前提条件、历史记录及背景。

（3）会议准备

为了保证评审质量，要求在评审会议开始前至少 3 天向所有的评审员发送评审包。同时，评审员需要在评审会议开始前独立、认真地阅读需要评审包的内容，保证在评审会议开始时，已经熟悉评审的内容。在这期间，需要完成主要工作如下 ：

- 要求每个检查者以特定的角度准备评审。例如，检查交叉引用的一致性，检查接口错误，

检查对以往的规范的可追溯性和一致性，检查对标准的符合性。

- 检查工作产品，理解它，发现其缺陷，并提出问题。使用适当的缺陷检查表，集中于发现这类工作产品中普遍存在的缺陷。适当使用其他分析方法查找缺陷。
- 将微小缺陷记录到微错清单上，如排版错误或风格不一致。在评审会议上或之前交给创建者。

（4）评审会议

准备工作完成后，将所有的评审员召集在一起召开正式的评审会议。评审会议的工作主要有如下几点 ：

- 小组组长介绍所有与会人员并说明其角色。陈述评审的目标。指导检查者将精力集中于发现缺陷，而不是解决方法。提醒参与者评论要针对正在评审的工作产品，而不是创建者。
- 确认准备情况 ：小组组长询问每个检查者的准备时间，并记录到评审总结报告上。如果准备不充分，则需要中止评审会议并重新安排会议时间。
- 展示工作产品 ：作者或读者向评审小组描述工作产品的各部分。
- 提出缺陷和问题 ：每当阅读人展示完工作产品的一部分，评审员应该指出关心的，潜在的缺陷，疑问或改进建议。
- 记录问题 ：对每个提出的问题，记录者都需要记录到问题日志上，确保所有的问题都被正确地记录。
- 解答问题 ：当有评审员提出问题时，作者需要简短回答提出的问题，使检查者进一步了解工作产品，从而帮助发现缺陷。

在会议最后，评审小组应该就评审内容进行最后讨论，形成评审结果。可能的评审结果有如下 4 种 ：

- 接受，评审内容不存在大的缺陷，可以通过。
- 有条件接受，评审内容不存在大的缺陷，修订其中的一些小缺陷后，可以通过。
- 不能接受，评审内容中有较多的缺陷，作者需要进行返工，并在返工之后重新进行评审。
- 评审未完成，由于某种原因，评审未能完成，还需要后续会议。

会议结束之后，评审小组还需要一系列的评审结果。评审结果包含了如下几项 ：

- 问题列表。
- 评审总结报告（会议记录）。
- 评审决议。
- 签名表。
- 问题列表说明了项目或产品中存在的问题，并需要后期跟踪。评审总结报告包含了评审的内容，评审人，会议总结等基本信息。

（5）返工和跟踪

评审会议结束时并不意味着评审已经结束了。评审会议的一个主要输出就是问题列表，发现的大部分缺陷是需要作者进行修订和返工的。因此需要对作者的修订情况进行跟踪，其目的就是验证作者是否恰当地解决了评审会上所列出的问题，并使所有的问题都得到妥善解决。

在前面我们提到了评审的最后决议有接受、有条件接受、不接受、未完成 4 种情况。其中接受和未完成基本不存在返工和缺陷的跟踪，因此返工和跟踪主要针对有条件接受和不接受。

① 有条件接受的缺陷跟踪

对于有条件接受的情况，被评审产品的作者在评审会后需要对产品进行修改，修改期限一般为 3 ～ 5 个工作日。修改完成后，被评审产品作者将修改后的被评审产品提交给所有的评审组成员。

评审组对修改后的被评审产品进行确认，在 1-2 个工作日内提出反馈意见。如有反馈意见，被评审产品作者应立即修改并重新发给评审组。

评审组长做好评审会后的问题跟踪工作，确定评审决议中的问题是否最终被全部解决。如全部解决，则认为可以结束此次评审过程 ；如仍有未解决的问题，则评审组长应督促被评审产品的作者尽快处理。

在满足结束此次评审过程的条件后，评审组长要将评审报告发给所有的评审组成员、被评审产品作者、SQA 人员。评审报告可以看作是评审会结束的标志。

② 不接受的缺陷跟踪

对于不接受的评审结果，被评审产品的作者在评审会后需要根据问题列表对产品进行返工，并将返工后的结果提交给所有的评审组成员。

评审组长检查修改后的产品，如果认为返工后的结果满足评审入口条件，则重新组织和召开评审会议对产品进行审查。

20.2.4　单元测试与集成测试

工厂在组装一台电视机之前，会对每个电子元件进行检验，或者要求电视机组装厂的每个供应商提供符合质量标准的电子元件，这种对电子元件的检验，就相当于软件中的单元测试。

单元测试的对象是构成软件产品或系统的最小的独立单元，如封装的类或对象、独立的函数、进程、子过程、组件或模块等。在单元测试活动中，软件的独立单元将与程序的其他部分被隔离开来。单元测试目的是检验每个软件单元能否正确地实现其功能，满足其性能和接口要求，验证程序和详细设计说明的一致性。

单元测试包括静态测试和动态测试，静态测试就是程序代码的审查活动，互为代码审查、集体开会审查代码，可以使用测试工具来完成对代码的检查，从编程语言的语法、规则和代码风格等各个方面检查是否满足实现定义的要求。而动态测试是通过单元测试用例、测试工具来执行程序以更好地实现测试的目标。单元测试可以采用白盒方法，也可以采用黑盒方法，但一般以白盒方法为主。

- 规格说明测试或 “ 黑盒测试 ”，验证单元外观上的可观察的行为 ；
- 结构测试或 “ 白盒测试 ”，验证单元的内部实现。

集成测试是将已分别通过测试的单元按设计要求组合起来再进行的测试，以检查这些单元之间的接口是否存在问题。集成测试是一个逐渐加入单元进行测试的持续过程，即采用渐增式测试模式，直至所有单元被组合在一起，成功地构成完整的软件系统，从而完成集成测试的使命。

集成测试一般，即把下一个要测试的模块同已经测试好的模块结合起来进行测试，测试完以后再逐步增加新的模块，不断继续下去，完成集成测试。目前业界普遍实行每日构建和每日集成，这就是人们经常所说的“持续集成”，最终使集成测试成为日常编码中的一部分内容，和单元测试并行进行，即编码、单元测试、集成测试在软件这个层次是并行进行的，没有明确的阶段划分。

20.2.5　功能测试

功能测试比较容易理解的，主要是根据产品规格说明书，来检验被测试的系统是否满足各方面功能的使用要求，功能测试也可以分为单元的功能测试和系统的功能测试，但其任务主要有：

- 每项功能符合实际要求。
- 功能逻辑清楚，符合使用者习惯。
- 菜单、按钮等各项操作是否正常、灵活。
- 系统的界面清晰、美观。
- 系统的各种状态按照业务流程而变化，并保持稳定。
- 能接受正确的数据输入，对异常数据的输入可以进行提示、容错处理等。
- 数据的输出结果准确，格式清晰，可以保存和读取。
- 软件升级后，能继续支持旧版本的数据。
- 与外部应用系统的接口有效。

对于功能测试，这里列出的是一般情况，针对不同的应用系统，其测试内容的差异很大，但都可以归为输入、输出、操作逻辑和数据处理过程等方面的测试。在功能测试中，常用的黑盒测试方法有等价类划分法、边界值划分法、错误推测法、因果图法和组合分析法等。

20.2.6　回归测试

无论在进行系统测试还是功能测试时，当发现一些严重的缺陷而需要修正时，会构造一个新的软件包（Full Build）或新的软件补丁包（Patch），然后进行测试。这时的测试不仅要验证被修复的软件缺陷是否真正被解决了，而且要保证以前所有运行正常的功能依旧保持正常，而不要受到这次修改的影响。这就要求进行回归测试。回归测试的目的是在程序有修改的情况下保证原有功能正常的一种测试策略和方法，因为这时的测试不需要进行全面测试，从头到尾测一遍，而是根据代码修改所影响的情况进行有效测试。

在软件生命周期中的任何一个阶段，只要软件发生了改变，就可能给该软件带来新的问题。软件的改变可能是源于发现了缺陷并做了修改，也有可能是因为在集成或维护阶段加入了新的功能或增强原有的功能。当软件中所含错误被发现时，如果错误跟踪与管理系统不够完善，就可能会遗漏对这些错误的修改；而开发者对错误理解的不够透彻，也可能导致所做的修改只修正了错误的外在表现，而没有修复错误本身，从而造成修改失败；修改还有可能产生副作用从而导致软件未被修改的部分产生新的问题，使本来工作正常的功能产生错误。同样，在有新代码加入软件的时候，除了新加入的代码中有可能含有错误外，新代码还有可能对原有的代码带来影响。因此，每当软件发生变化时，就必须重新测试现有的功能，以便确定修改是否达到了预期的目的，检查修改是否损害了原有的正常功能。同时，还需要补充新的测试用例来测试新的或被修改了的功能。为了验证修改的正确性及其影响就需要进行回归测试。

回归测试作为软件生命周期的一个组成部分，在整个软件测试过程中占有很大的工作量比重，软件开发的各个阶段都可能需要进行多次回归测试。在渐进和快速迭代开发中，新版本的连续发布使回归测试进行的更加频繁，而在极限编程（XP，eXtreme Programming）方法中，更是要求每天都进行若干次回归测试。因此，通过选择正确的回归测试策略来改进回归测试的效率和有效性是非常有意义的。

在软件生命周期中，即使一个得到良好维护的测试用例库也可能变得相当大，使得每次回归测试都重新运行完整的测试包变得不切实际，时间和成本约束也不允许进行一个完全的测试，需要从测试用例库中选择有效的测试用例，构造一个缩减的测试用例集合（test suite）来完成回归测试。选择回归测试策略应该兼顾效率和有效性两个方面，下面有几种方法，在效率和有效性的侧重点是不同的。

（1）再测试全部用例，选择测试用例库中的全部测试用例构成回归测试包，这是一种比较安全的方法，具有最低的遗漏回归错误的风险，但测试成本最高。再测试全部用例几乎可以应用到任何情况下，基本上不需要进行用例分析和设计，但是随着开发工作的进展，测试用例不断增多而带来相当大的工作量，受预算和进度的限制。

（2）基于风险选择测试，基于一定的风险标准来从测试用例库中选择回归测试包。首先运行最重要的、关键的和可疑的测试，而跳过那些次要的、例外的测试用例或那些功能稳定的模块。运行那些次要用例即便发现缺陷，这些缺陷的严重性也较低。

（3）基于操作剖面选择测试，如果测试用例是基于软件操作剖面开发的，测试用例的分布情况反映了系统的实际使用情况。回归测试所使用的测试用例个数可以由测试预算确定，回归测试可以优先选择那些针对最重要或最频繁使用功能的测试用例，释放和缓解最高级别的风险，有助于尽早发现那些对可靠性有最大影响的故障。

（4）再测试修改的部分，当测试者对修改的局部化有足够的信心时，可以通过相依性分析识别软件的修改情况并分析修改的影响，将回归测试局限于被改变的模块和它的接口上。通常，一个回归错误一定涉及被修改的或新加的代码。在允许的条件下，回归测试尽可能覆盖受到影响的部分。这种方法可以在一个给定的预算下最有效地提高系统可靠性，但需要良好的经验和深入地代码分析。

综合运用多种测试技术是常见的，在回归测试中也不例外，测试者也可能希望采用多于一种回归测试策略来增强对测试结果的信心。不同的测试者可能会依据自己的经验和判断选择不同的回归测试技术和方法。

20.2.7　系统的非功能性测试

系统测试除了前面所说的功能测试之外，还要对系统的非功能特性（如可用性、安全性和可扩充性等）进行验证。例如，性能测试在运行软件系统时模拟各种实际环境（包括高负载），对系统自身表现、所占用的资源和其他指标进行监测，以验证系统是否满足产品的性能需求。

（1）负载测试（Stress test），也称为强度测试、压力测试。负载测试是模拟实际应用的软硬件环境及用户使用过程的系统负载（如并发访问用户量），长时间或超大负载地运行被测系统，以发现系统可靠性、性能瓶颈等方面的问题。

（2）容量测试（Capacity test），通过不断增加负载，以确定反映软件系统应用特征的某项

指标的极限值，如某个 Web 站点可以支持的多少个并发用户的访问量、网络在线会议系统的与会者人数。

(3) 性能测试（Performance test），通过测试以评估系统运行时的性能表现，如获取系统运行速度、响应时间、占有系统资源（如内存和 CPU）等方面的实际值，然后和预先设定的目标值进行比较，通过测试结果分析，发现性能瓶颈，并评估软件系统是否满足性能要求。

(4) 安全性测试（Security test），检查系统对非法侵入的防范能力。在安全性测试中，测试人员模拟非法入侵者，采用各种办法攻击软件系统，例如突破用户登录保护、功能使用权限、数据存取权限等防线。系统安全设计的准则是，使非法侵入的代价超过被保护信息的价值。

(5) 容错测试（Recovery test），主要检查系统的容错能力。当系统出错时，能否在指定时间间隔内修正错误并重新启动系统。容错测试首先要通过各种手段，让软件强制性地发生故障，然后验证系统是否能尽快恢复。对于自动恢复需验证重新初始化、检查点、数据恢复和重新启动等机制的正确性；对于人工干预的恢复系统，还需评估平均修复时间，确定其是否在可接受的范围内。

20.2.8　验收测试

验收测试是在软件产品完成了单元测试、集成测试和系统测试之后、产品发布之前所进行的软件测试活动。它是技术测试的最后一个阶段，也称为交付测试，一般用户或用户代表要参与其测试，这是其一个重要特征。验收测试一般会根据产品规格说明书严格地检查产品，逐字逐句地对照说明书以检查事先定义的软件产品的各项具体要求，确保所开发的软件产品符合用户期望和需求。验收测试就是检验产品和产品规格说明书（包括软件开发的技术合同）的一致性，同时验收测试是在用户的实际使用环境中进行，包括用户的硬件环境、用户数据和实际操作习惯等。

云服务可能没有验收测试，因为不需要将产品交付给用户，而是通过在公司内部云平台上部署软件产品，从而让用户使用服务功能，完成交付。所以对于云服务平台，一般进行 BETA 测试，即在数据中心部署研发部门交付的版本，先让内部员工或外部少数用户试用，试用期间发现严重问题时得到及时修正，再发布补丁包，继续试用，等产品质量相对稳定之后，再全面部署到产品线上，让所有用户使用，这时，才算正式交付产品。对非关键性产品，特别是一些免费产品，甚至在产品线上直接提供 BETA 版本，由用户完成在线测试。在云服务上，可以采用 SaaS 服务的一些概念，例如 ：

LA ：Limited Available，即云服务只提供给非常有限的用户，相当于 BETA 测试。

GA ：Limited Available，即云服务提供给所有用户，这时才算产品正式上线。

验收测试还出现敏捷方法中，包括 Scrum 框架。在敏捷方法中，鼓励开发人员进行足够的单元测试，但单元测试不能发现整个系统所存在的某些问题。这就需要单独的阶段或活动来对整个系统的行为进行验证，以确定系统是否能够全面满足客户的需求，从而确定系统整体实现的质量状态。在敏捷开发方法中，验收测试包括对每个新实现的用户故事进行验证，更多的测试是针对用户的业务流程完成端到端的测试。理想情况下，敏捷中的验收测试是根据开发前定义的验收标准来进行测试。也就是说，验收测试用例是在写代码之前事先写好的。验收测试内容，主要包括：

- 用户交互测试，验证系统行为是否正确、是否符合用户的期望。
- 可用性测试，验证系统是否易用，包括用户界面是否良好、操作是否合理。
- 性能测试，测试应用程序在各种负荷下的工作状态。
- 压力测试，使应用程序在用户和事物的极限值情况或其他任何让应用程序处在压力下的运行情况运行。

验收测试可以采用手工方法进行，也可以采用自动化手段进行，借助一些验收测试框架（如FitNesse）实现这一目标。

20.2.9　技术运营阶段的QA活动

在技术运营阶段，出现的问题比较多，一方面可能是由于设计存在问题（这将在后面的 20.4 节进行详细讨论）而导致出错、性能降低甚至崩溃等质量事故，另一方面由于数据中心管理混乱、不规范操作、缺少服务监控与报警机制等造成的。所以正如前面提到的，技术运营阶段的质量保证（Quality Assurance，QA）在云服务平台是至关重要的，包括环境、网络、设备、软件、存储、防病毒、日常操作、权限等各种涉及技术运营质量因素的管理。对这些对象的日常管理工作有明确的责任说明和制度建设，责任到人、有章可循、奖惩分明，实现整个系统的全生命周期的质量管理，包括研发阶段、技术运营阶段的质量管理，以及这两个阶段的 QA 工作的相互衔接。

（1） 人是决定的因素，每个人都应该建立问题预防的意识，并加强培训，因为不少系统故障（甚至宕机）都是由人为错误造成的。人员分工明确，包括各种设备、各种层次的管理人员、操作人员。例如有专门的网络管理人员、虚拟机的管理人员、数据库管理人员（DBA）和业务的技术支持人员等。

（2） 云服务平台的可用性、安全性（下一节 20.3 将详细讨论 ）等是 QA 活动关注的主要对象，例如，为了保证服务的可用性，始终要做好系统的监控、系统和数据的备份、故障转移、系统定期维护、日志分析等工作。故障恢复。如果出现故障，确保在尽可能短的时间内完成恢复，包括系统恢复和数据恢复 ；更重要的是建立紧急情况操作流程、应急响应机制等。如果系统用户增加很快，系统的性能在逐渐降低的时候，这时就需要在线扩容，系统应具有良好的可扩充性。而云服务的数据安全、隐私保护已成为用户对云服务最为担忧的问题，要解决这类安全的问题（如系统安全漏洞、身份验证失效、访问控制不利、DoS 攻击等），需要强化技术运营管理和加强安全技术保障，如分级分权管理、用户数据隔离、数据加密、身份认证等。

（3） 灵活性、个性化是云服务的显著特点之一，服务面对有着不同需求的用户，为了让客户更满意使用服务，系统应具有满足用户个性化定制需求的能力，提供相应的个性化配置功能，动态、快速地调整云平台的资源，满足用户需求。例如，亚马逊（Amazon）云服务平台最大用户之一 Pinterest，每到周末，就需要很多的资源，亚马逊平台能提供这些资源，但到了周一，需要的资源又少得多，这些资源能够释放。

（4） 全方位监控，技术运营 QA 工作从日常监控入手，依靠完整的、全方位实时有效的监控系统，做好事件管理和变更管理，事先建立好应急预案，做好监控数据的记录和技术分析，提前发现问题、消除隐患。多层监控，如基础设施（电力、温度和湿度控制、空调、网络设备、服务器硬件等）、通讯层（网络连通性、路由、DNS、虚拟网络）、操作系统

层（包括虚拟机、虚拟存储、数据库、内存和 CPU 使用情况等）、应用服务层（包括业务流量、业务数据、日志等）进行全面监控。而且这种监控需要通过工具来实现 24 小时的自动化监控，所以会开发或引进一系列工具来完成，如基础设施监控工具、系统监控工具、应用服务器监控工具、业务后台管理平台、短信平台等。

(5) 加强系统技术运营测试。技术运营测试主要指在系统运行阶段实施的测试活动，在整体上完成对实际系统的可用性和服务水平的验证。技术运营测试完全基于产品运行环境（生产环境）下实施产品在线测试（production online testing），主要包括对上线后系统运行状态进行监测和异常报告、对上线系统实施动态测试等，以寻找系统缺陷和风险，并对相关缺陷进行根本原因分析（Root Cause Analysis），预防问题，指导系统调优 ；对系统风险进行评估，制定预防策略，确保系统上线后稳定运行。

(6) 虚拟化的统一管理，如虚拟机 VMWare vCenter，包括高效、灵活的动态分配资源，一键自动部署各种版本或一个完整的应用服务器等。可以实施更高级的技术运营管理，例如预测性维护和故障模型分析，进一步降低质量风险。

(7) 客户关系管理。云服务平台面对众多不同的用户，为了留住客户，在技术运营过程中要维护好和客户的关系，做好服务评审（这可以基于 SLA 来进行，见 20.5 节的详细描述）、定期进行客户满意度调查，与客户进行定期或不定期的针对服务提供情况的沟通，处理好客户的抱怨。所有这些工作，也都可以归为技术运营阶段的 QA 活动。

20.3　云服务平台的特有质量诉求

传统的软件系统往往由特定的用户所使用，业务领域集中，需求比较明确，应用环境相对单一。但在云服务下，借助互联网技术和力量，软件服务面向全国用户或全球用户，为许多不同企业的一类业务应用提供统一的支撑环境，即不同用户在相同环境下共享软件系统的主要功能。而且，这种云服务允许用户能够自我定制其用户界面（UI）和功能、能够选择自己喜欢的模板、对功能可以进行灵活的组合，这就要求灵活的系统架构来适应客户定制的需求。另外，软件服务一旦上线，系统就应一直不间断地运行下去。即使为了满足新的用户需求，对系统进行升级、更新版本的时候，软件服务也不能中断。所以，基于云服务的软件系统，不应该受到时间和地域的限制，分布在各地的用户随时都应能享受相同的优质服务。相比传统的软件系统，云服务对系统的安全性、可用性、兼容性、可扩充性、可维护性等各方面都有着更高的要求。

同时，在云服务模式中，由于软件部署由软件服务商自己控制，且不会像渠道销售软件套装产品其很长时间和制造成本，所以云服务模式下的软件发布周期可以大大缩短，力求在软件开发过程中做到最简单和最有效，最优先要做的是通过尽早的、持续的交付有价值的软件来使客户满意。例如，在云服务模式下的软件开发，可以将敏捷方法和 RUP 过程方法结合起来，敏捷过程能够保持快速、稳定的开发速度，RUP 过程可以保证系统的灵活架构、良好的扩充性和移植性，促进开发过程在质量和速度上达到平衡，确保为用户提供持续、稳定的高质量服务。

不论是传统软件系统还是云服务系统，在功能性、易用性、性能、安全性、可扩充性等方面有相同的质量诉求，只是云服务模式在某些方面（如稳定性、可用性等）会有更高的要求，而且在质量保证过程中其具体的活动、实践也具有自身的特点。

20.3.1　可用性

云服务平台一个显著的质量诉求就是高可用性，也就是经常说的，每周 7 天每天 24 个小时（7x24）不间断地运行。可用性和可靠性是一致的，高可用性需要系统是高可靠的，一般都是通过并行系统、冗余组件设计、系统故障转移等实现。产品或服务对于客户的是否能保持有效，可以用“ 系统平均无故障时间（MTTF，Mean Time To Failure）除以总的运行时间（MTTF 与故障修复时间之和）” 来计算系统的可用性。即在预定的启动时间中，系统真正可用并且完全运行时间所占的百分比，例如，网上银行系统需要大于 99.999% 的可用性才能满足客户的质量要求。大于 99.999% 的可用性又意味着什么呢？例如，以一年来考虑，一年时间为 525600 分钟，系统失效时间为 525600 x 0.001% = 5.256 分钟，即系统宕机或不能提供服务时间要少于 5.256 分钟，即小于 315 秒。

一个系统也可以根据不同的时段定义不同的可用性，用户经常使用系统的时间不同，在正常的时间用户使用频率要高得多，系统就要提供更可靠的服务。例如可以这样说明 :“ 在工作日期间，在当地时间早上 6 点到午夜 11 点，系统的有效性至少达到 99.995%，而在其他时间（午夜 11 点到第二天 6 点），系统的有效性至少要达到 99.95% ”。所以系统迁移、备份、部署新的版本等活动一般都安排在半夜里进行。如果是面向全球的客户，必须部署多个服务器集群，每个服务器集群服务某个区域，如分为欧洲、东亚、西亚、美洲等，这样就可以分别处理。否则，只有一个服务集群面向全球用户，就没有时间区分，每个时间很重要，因为各国时区不同，任何一个时间都有比较多的用户在使用这个系统。

20.3.2　安全性

云服务多数都是通过互联网来提供的，而在互联网上用户的行为难以规范和控制，系统的安全性就显得特别重要。安全性是指系统和数据的安全程度，包括功能使用范围、数据存取权限等受保护和受控制的能力。除了数据加密、隔离、存储、备份和恢复等要求之外，安全性还需要设定系统合理的、可靠的系统和数据的访问权限，防止一些不速之客的闯入和黑客的攻击，以避免数据泄密和系统瘫痪。

软件系统的安全性和可靠性也往往是一致的，安全性高的软件，其可靠性也要求相对高，因为任何一个失效，可能造成数据的不安全。一个安全相关的关键组件，需要保证其可靠，即使出现错误或故障，也要保证代码、数据被储存在安全的地方，而不能被不适当的使用和分析。

但软件的安全性和其性能、适用性会有些冲突，如加密算法越复杂，其性能可能会越低 ；或者对数据的访问设置种种保护措施，包括用户登录、口令保护、身份验证、所有操作全程跟踪记录等等，必然在一定程度上降低了系统的适用性。

20.3.3　可扩充性

软件必须能够在用户的使用率、用户的数目增加很快的情况下，保持合理的性能。只有这样，才能适应用户的市场扩展得可能性。系统的可扩充性是指将来功能增加、系统扩充的难易程度或能力。例如简单的模块结构、模块间低耦合性、多层分布体系架构等特性可以改善系统的可扩充性。

可扩充性和可伸缩性、可维护性有关系，如果系统具有良好的可伸缩性或可维护性，那么

系统就具有良好的可扩充性。高可伸缩性代表一种弹性，在系统扩展成长过程中，软件能够保证旺盛的生命力，通过很少的改动甚至只是硬件设备的添置，就能实现整个系统处理能力的线性增长，实现高吞吐量和低延迟高性能。可伸缩性讲究平滑线性的性能提升，更侧重于系统的水平伸缩，通过廉价的服务器实现分布式计算。

可维护性是指在系统在运行过程中，当环境改变或软件发生错误时，进行相应修改所付出的工作量或难易程度。可维护性取决于理解软件、更改软件和测试软件的难易程度，可维护性与灵活性密切相关。高可维护性对于那些经历周期性更改的产品或快速开发的产品很重要。

20.4　需求评审和设计评审

基于软件服务的质量要求，需要审查软件系统能否实现预定的质量特征指标，以及为实现这些指标的策略或技术方案是否合理。软件服务的质量目标会受到各种各样约束和因素的影响，通过因果图分析方法确定影响因素和软件服务质量目标之间的关系，找出对质量目标影响最关键的因素，然后针对系统设计进行审查，以确定这些关键因素是否得到充分考虑和获得最有效的解决方案。该解决方案描述软件体系结构所达到的服务质量（Quality of Service，QoS）技术指标，如性能指标、安全性等级、可靠性要求等，并在用户使用模式分析的基础上，绘制用户业务操作的完整流程，从而审核系统设计是否符合用户的各种使用模式，以及系统架构设计是否遵守相应的技术规范。

20.4.1　需求评审

软件缺陷并不只是在编程阶段才产生的，需求和设计阶段同样会产生问题，各种阶段性成果中都有可能存在缺陷，包括需求定义文档、设计文档、程序代码等。如果按阶段性工作成果——市场需求文档、规格说明书、系统设计文档、程序代码等存在的缺陷所占的比例来看（即进行缺陷分布的对比分析），权威机构调查的结果显示，软件缺陷出现最多的地方是软件需求规格说明书（即软件需求定义），而不是程序代码。其次，如果错误发现得越迟，那么修正这个错误的成本就越大。例如需求定义问题，如果在需求评审阶段被发现，只要修改需求文档就可以了，但如果在系统测试阶段被发现，为了修正这个错误，就需要修改需求文档，然后修改设计、代码等，所返工带来的工作量要大得多，也就是造成大得多的成本。所以，修改后期的错误所做的工作要比修改前期的错误大得多。如果错误不能及早发现，那只可能带来越来越严重的后果。

所以需求评审是软件质量保证过程中最重要的活动之一。通过需求评审来保证系统需求在市场需求文档（MRD）或产品需求文档（PRD）及相关的文档中无歧义的描述。不一致、遗漏和错误将会被审查出来并得到改正，而且需求描述文档应该符合已定义的软件过程和软件产品的标准。需求评审也是做好软件测试计划和设计等工作的基础。概括起来，需求评审对软件测试和质量的作用表现在以下几个方面 ：

（1） 对软件需求进行正确性的检查，以发现需求定义中的问题，尽早地将缺陷发现出来，降低成本，并使后续过程的变更减少，降低风险。

（2） 保证软件需求的可测试性，即确认任何客户需求或产品质量需求都是明确的、可预见的并被描述在文档中，将来可以用某种方法来判断、验证这种需求或特性是否已得到实现。

（3） 通过产品需求文档的评审，与市场、产品、开发等各部门相关人员沟通，使得大家认识一致（on the same page），避免在后期产生不同的理解，引起争吵。

（4） 通过产品需求文档的评审，更好地理解产品的功能性和非功能性需求，为制定测试计划和测试方法打下基础，特别是为测试范围、工作量等方面的分析、评估工作积累充足的信息。

（5） 在需求文档评审通过后，测试的目标和范围就确定了。虽然此后会有需求的变更，但可以得到有效的控制，这样可降低测试的风险。

需求评审，归为静态测试的范畴，包含了文档评审和技术评审双重内容，通常通过正式的评审会议来进行。为了保证评审的质量和效率，需要精心挑选评审员。首先要保证不同类型的人员都要参与进来，否则很可能会漏掉很重要的需求 ；其次在不同类型的人员中要选择那些真正与系统相关的、对系统有足够了解的人员参与进来，否则很可能使评审的效率降低或者最终不切实际地修改了系统的范围。在软件需求评审过程中，一般来说，作者事先主动地向相关利益者介绍需求分析、定义的背景和说明相关的细节，以征集大家的意见。在介绍之前，应先将文档发给相关利益者。而评审员应仔细地阅读需求文档，将发现的问题、不明白的地方一一记下来，通过邮件发给文档的作者，或通过其他形式（面对面会谈、电话、远程互联网会议等）进行交流。通过交流，大家达成一致的认识和理解，并修改不正确、不清楚的地方。在各种沟通形式中，自然、面对面的沟通形式效率最好，但是在口头交流达成统一意见后，最后须通过文档、邮件或工作流系统等记录下来，作为备忘录。

如果通过一些非正式的形式（临时评审、走查等）不能很好地完成需求分析的评审，就必须通过正式形式（集体 / 小组评审会议）来完成这一工作。对于比较大型、需求改动较大等的项目来说，一次评审会议也许还不够，还要通过 2 ~ 3 次甚至更多次的评审会议才能最后达成一致。评审是否完成是以需求文档获得多发签发（sign - off）或签字通过为标志的。这不应该只体现在“ 签字 ”形式上，更重要的是达到下面的结果 ：

- 所有参与方达成一致。
- 已发现的问题被阐述清楚、被修正。

对系统需求的评审着重于审查对用户需求描述的解释是否完整、准确。根据 IEEE 建议的需求说明标准，对于系统需求进行审查的质量因素有如下内容。

（1） 正确性 ：检查在任意条件下软件系统需求定义及其说明的正确性。如需求定义是否符合软件标准、规范的要求？是否所有的功能都有明确的目的，是否存在对用户无意义的功能？每个需求定义是否都合理，经得起推敲？所采用的算法和规则是否科学、成熟和可靠？有哪些证据说明用户提供的规则是正确的？是否正确地定义了各种故障模式和错误类型的处理方式？对设计和实现的限制是否都有论证？

（2） 完备性 ：涵盖系统需求的功能、性能、输入 / 输出、条件限制、应用范围等方面，覆盖率越高，完备性越好。通过增强创造力的方法避免思维的局限性，能全面地考虑各种各样的应用场景或操作模式，以提高完备性。如是否有漏掉的功能？是否有漏掉的输入、输出或条件？是否考虑了不同需求的人机界面？需求定义是否包含了有关文件（如质量手册、配置计划等）中所规定的特定需求？功能性需求是否覆盖了所有非正常情况的处理，出现异常情况时系统如何响应？是否识别出了所有与时间因素有关的功能？是否识别并

定义了在未来潜在变化的需求？是否定义和说明了系统输入和输出的来源、类型、值 域、精度、单位和格式等？

(3) 易理解性 ：需求文档的描述性被理解的难易程度，包括清晰性。如需求描述是否足够清楚和明确，使其已能作为开发设计说明书和功能性测试数据的基础？是否将系统的实际需求内容和所附带的背景信息分离开来？每一个需求是否只有一种解释，语言是否有歧义性？功能性需求的描述结构化、流程化是否良好？是否使用了形式化或半形式化的语言？需求定义是否包含了实现的细节，是否过分细致了？

(4) 一致性 ：包含了兼容性，如所定义的需求之间是否一致，是否有冲突和矛盾？是否使用了标准术语和统一形式？使用的术语是否是唯一的？同义词、缩略语等的使用在全文中是否一致并事先已予以说明？所规定的操作模式、算法和数据格式等是否相互兼容？是否说明了系统中软件、硬件和其他环境之间的相互影响？

(5) 可行性 ：需求中定义的功能是否具有可执行性、可操作性等。如需求定义的功能是否能通过现有的技术实现？所规定的模式、数值方法是否能解决需求中存在的问题？所有的功能是否都能够在某些非常规条件下实现？是否能够达到特定的性能要求？

(6) 可维护性 ：系统的运行状态能实时监控、日志格式及其记录，而且系统和数据的备份与恢复、系统迁移、系统扩充等需求，在云服务模式下要得到足够的关注。

(7) 易修改性 ：对需求定义的描述易于修改的程度，如是否有统一的索引、交叉引用表？是否采用了良好的文档结构？是否有冗余的信息？

(8) 易测试性（可验证性）：所定义的功能正确性是否能被判断？系统的非功能需求（如性能、可用性等）是否有验证的标准和方法？输入、数出的数据是否有清楚地定义从而容易验证其精确性？

(9) 易追溯性 ：每一项需求定义是否可以确定其来源？是否可以根据上下文找到所需要的依据或支持数据？后续的功能变更是否都能找到其最初定义的功能？功能的限制条件是否可以找到其存在的理由？

对用户的需求进行评审时，上述标准就是评审的依据，逐字逐句地审查需求规格说明书的各项描述。需求规格的描述，不仅包括功能性需求，而且包括非功能性需求。例如，系统的性能指标描述应该清楚、明确，而不是给一个简单的描述——“ 每一个页面访问的响应时间不超过 3 秒 ”，业务要求通常用指定响应时间的非技术术语表示性能，如下所示 ：

“ 系统能够每秒接受 50 个安全登录，在正常情况下或平均情况下（如按一定的时间间隔采样）Web 页面刷新的响应时间不超过 3 秒。在定义的高峰期间，响应时间也不得超过 12 秒。年平均或每百万事务的错误数须少于 3.4 个。”

业务要求通常用指定响应时间的非技术术语表示性能。有了更专业、更明确的性能指标，就可以对一些关键的使用案例进行研究，以确定在系统层次如何保证该要求得到实现的结构、技术或方式。在多数情况下，将容量测试的结果作为用户负载的条件，即研究在用户负载较大或最不利情况下保证系统的性能。如在这种情况下，系统的性能有保证，在其他情况下就不会有问题。

20.4.2　系统架构设计评审

在云服务系统架构设计中，要考虑到网站系统的性能、可靠性和可扩充性之外，还要考虑数

据的迁移、不同的升级方式、多版本共存的运行环境等需求，对数据兼容性、系统的备份和恢复等进行充分的讨论和分析，保证用户升级过程中，所获得服务没有受到影响，数据受到保护，一切使用正常。系统架构设计要完成下列一系列的任务 ：

- 构造软件系统的逻辑体系结构，确定实现解决方案所需的软件体系结构、组件及其之间的层次、依赖关系。
- 将逻辑体系结构中指定的组件映射到物理环境，从而生成一个可实施的体系结构。
- 创建一个实现规范，该规范提供关于如何构建体系结构的所需信息。
- 创建一系列详细说明软件系统部署的计划，包括迁移计划、安装计划、用户管理计划、测试和验证计划等。

体系结构逻辑设计时，不仅需要确定提供各种软件服务的组件，还要确定必要中间件和平台服务的其他组件。例如，J2EE 体系结构正是由下列三层构成的 ：

（1） 系统服务质量，如性能、可用性、可伸缩性及其他要素。

（2） 逻辑层，表示可被客户层访问的对象、业务和数据服务，即要基于软件服务的特性，表示软件组件组成的逻辑层次关系及其业务关系。

（3） 基础结构服务，一系列允许分布式组件间相互通信和交互操作的基础结构服务。

无论是对 J2EE 体系结构还是 .Net 体系结构，都非常适合设计为多层体系结构。在多层体系结构中，服务根据其提供的功能放在不同层次中。每个服务都是逻辑独立的，并且可由同层或不同层的服务访问，如图 20-8 显示了企业应用程序的一个由 “ 客户层、表示层、业务服务层和数据层 ” 构成的多层体系结构模型。根据多层体系结构中的功能或服务层次有助于确定在网络中分配服务的方式，有助于确定体系结构中的组件之间所存在的访问或支持服务。多层体系结构的直观性有助于满足系统的可用性、可伸缩性、安全性和其他质量特性。

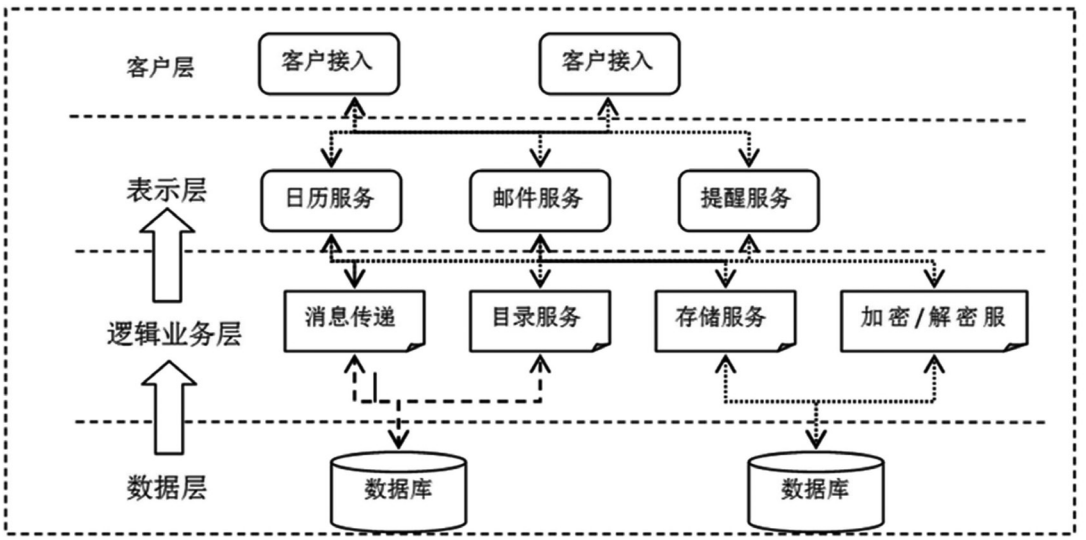


图20-8　多层体系结构模型示意图

20.4.3　系统部署物理设计评审

云服务通过互联网向用户提供服务，而这基础是软件系统的部署。软件部署的最终目标是实

现业务目标，这就要求在软件需求分析、设计和验证时，要充分考虑系统部署的各方面需求，包括服务器集群、分布式网络、故障转移、系统在线扩充、数据备份和恢复等。逻辑体系结构是确定分配服务最佳方式的关键，物理设计是在逻辑体系结构框架下展开并受其约束。

在物理设计审查时，还是要从服务级别协议（SLA）、QoS 出发，在合理的成本控制下，确保软件服务可以满足用户的需求，例如 ：

- SLA 指定了最低性能要求以及未能满足此要求时必须提供的客户支持级别和程度，相当于物理设计的底线。
- QoS 要求。物理体系结构和逻辑体系结构有清晰的映射关系，从而达到性能、可用性、可伸缩性、可维护性等 QoS 目标，例如系统从意外故障中恢复的过程。
- 用量分析。 有助于通过系统负载的使用模式来隔离性能瓶颈，开发出满足 QoS 要求的策略，以便用于物理设计中。尽管使用案例已包含在用量分析中，但评估部署设计时，应参考使用案例，确保任何案例中所揭示的问题在物理设计中得到处理或解决。
- 成本。 在物理设计中，满足 QoS 要求的同时，尽量降低成本。所以，有必要设计 2~3 个软件部署的物理方案，通过分析、比较，对资源优化，采用平衡策略，能够在业务约束范围内达到业务要求，并获得成本最优化。

软件部署的物理设计是一个反复进行的过程，通常要复查 QoS 要求和初步设计，考虑不同 QoS 要求之间的相互关系，对 QoS 和成本的平衡以获得最佳的部署方案，如表 20 - 1 所示。

表20–1软件部署物理设计的审查列表

系统性质	说明
性能	对于将 CPU 集中分布在个别几台服务器上的性能解决方案，服务能否对计算能力加以高效利用？（例如，某些服务对可高效利用的 CPU 数量有上限。）
潜在容量	设计策略是否处理超出性能估计的负载？ 对于过载，是进行垂直扩展的方式，或以负载平衡到其他服务器的方式，还是以这两种方式兼用的方式进行处理？ 在达到下一部署扩展重大事件点前，潜在容量是否足以处理出现的异常峰值负载？
安全	是否对处理安全事务所需的性能开销给予了充分考虑？
可用性	对于水平冗余解决方案，是否对长期维护资源给予了充分估计？ 是否已将系统维护所需的计划停机考虑在内？ 是否在高端服务器和低端服务器间求得了成本平衡？
可伸缩性	是否对部署扩展的重大事件点进行了估计？ 是否制定了可在达到部署扩展重大事件点前提供足够的潜在容量来处理预测的负载增长的策略？
可维护性	是否在可用性设计中考虑了管理、监视和维护成本？ 是否考虑了采用委派管理解决方案来降低管理成本？

20.5　云服务的验证

在云服务质量诉求中，包括功能正确性、易用性、性能以及 4.3 节中特别提到了系统可用性、可伸缩性、安全性等等，以保证云服务平台能提供符合规范的、高效的功能服务，全面地满足用户的基本需求。在功能和标准符合性测试、性能测试方面，与传统的软件测试没有很大不同。只是在功能测试上要关注用户的可定制性，虽然众多用户共享软件中许多功能，但每个用户

能够定制其所需的功能特性，也可以按照自己的文化、习惯和喜好等定制其 UI 界面。而且云服务平台多数通过 Web 方式提供服务，并大量融入了面向服务的体系结构，所以要求云服务平台能够很好地支持 UDDI、XML、SOAP、WSDL 等技术规范，这样 Web 服务测试（包括功能测试和性能测试等）所使用的大多数方法和技术可以被云服务验证所借鉴和引用。

20.5.1　可用性验证

软件系统的可用性主要通过下列三种方法——负载平衡、故障转移和备份机制来实现，而针对可用性进行验证时，首先在设计评审时尽可能发现和消除单一故障点（SPOF，Single-point of Failure），确保关键组件都有冗余部分或备份机制，任何组件失效时，但整个系统不会失效，能够继续向用户提供服务。

（1）负载平衡和故障转移

故障转移机制，实现对冗余硬件和软件的管理，在任何组件发生故障时能将服务自动转换到正常工作的组件。多数情况下，负载平衡和故障转移可以一起考虑，例如，平行冗余服务器可提供负载平衡和故障转移两种功能来提高可用性。最简单的一种情形是双服务器系统，一台服务器即可满足性能要求，另一台服务器作为备份服务器。其中一台服务器发生故障时，另一台服务器立刻接受请求，继续提供 100% 的服务，但这样会浪费资源、成本比较高。

为了降低成本，可以通过在两台服务器间分配性能负载来实现负载平衡和故障转移。如果一台服务器发生故障，所有服务仍然可用，但是性能只能达到完全性能的某个百分比（如 50%～80%）。例如，为满足性能要求的单个服务器需要配置 10 个 CPU, 这时每个服务器配置为 6 个 CPU，两个服务器为 12 个 CPU，正常运行（它们同时运转）时能保证 100%～120% 的性能。当其中一台服务器发生故障时，另一台服务器提供 6 CPU 计算能力，即满足 60% 的性能要求。

如果用 5 台双 CPU 服务器（5×2=10）来提供同样性能要求的软件服务，这时如果一台服务器发生故障，其余服务器可继续提供总计 8 个 CPU 的计算能力，达到 10 个 CPU 性能要求的 80%。如果在设计中增加一个具有 2 个 CPU 计算能力的服务器，实际得到的便是 N+1 设计。如果一台服务器发生故障，其余服务器可满足 100% 的性能要求。N+1 设计具有下列优点 ：

- 单台服务器发生故障时能更好满足性能要求。
- 即使不止一台服务器停机，仍然具有可用性。
- 可轮换将服务器停机，以进行维护和升级。
- 多台低端服务器的价格通常低于单台高端服务器。

如图 20-9 所示，主应用域通过多台 Web 服务器、2 台应用服务器、2 台数据库服务器等构成内部负载平衡和故障转移机制，通过主应用域（domain）和备份应用域构成系统级别的异地故障转移机制。负载平衡器（如 NetScaler LoadBalance）能把对某个服务的任意请求引导至该服务器集群（cluster）中当前负载最小的某个服务器上。如果任一实例发生故障，其他实例可以承担更大的负载。

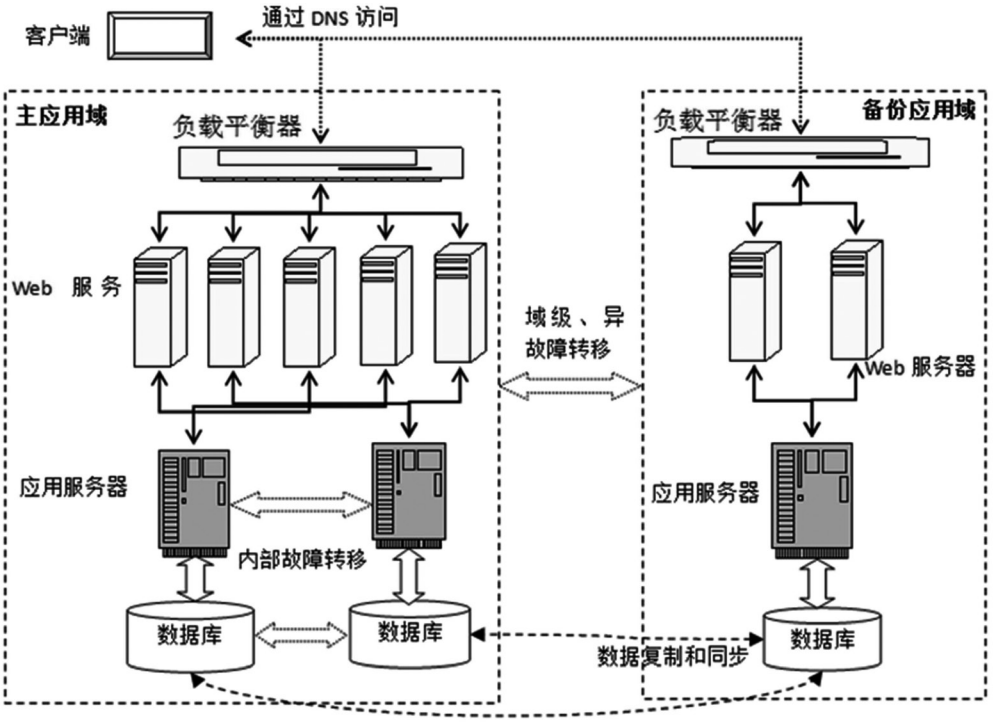


图20-9　可用性设计（故障转移、负载平衡）的示意图

（2）复制机制

复制机制主要应用于数据的可用性设计，包括文件复制、目录复制和数据库复制等。复制机制也有多种策略，例如有单主复制和多主复制。采用单主复制，为主数据库提供一个中心源，然后将该中心源分配到使用者副本中，而采用多主复制，在多个服务器间分配主数据库，然后为每个主数据库配置副本，如图 20-9 所示。

在多主复制中，一个或多个目录服务器实例管理主目录数据库。每个主数据库都有一个指定同步主数据库的过程的复制协议，可被复制为任意数量的使用者数据库并定期更新，而使用者的实例都按读取和搜索访问进行了优化，使用者接收的任何写操作都被引回到主数据库。多主复制策略提供了一个在更新主数据库时提供负载平衡、对目录操作提供本地控制的可用性策略。

（3）可用性设计验证所需信息

为验证可用性策略和设计，测试人员需要收集有关可用性的信息，主要有如下内容。

- 指定的可用性中有多少个“ 9 ”，99.9%、99.99% 或 99.999% ？
- 故障转移情况下的性能要求是什么？如 1 分钟完成故障转移，或故障转移的性能为原来的 60%。
- 用量分析是否区分高峰和非高峰使用时间？
- 地域考虑因素有哪些？
- 考虑可维护性、可伸缩性对可用性的要求？
- 是否存在单一故障点（Single-point of Failure）？

在用户看来，可用性更多牵涉到单个服务的可用性，并不总是整个系统的可用性。例如，即时消息传送服务不可用，通常情况下对其他服务的可用性几乎没有影响。但是，许多其他服务（如目录服务器）所依存的服务的不可用性则会受到较大影响。较高的可用性规范应该明确引用要求关键任务的特定使用案例和用量分析。根据一组有序的优先级列出可用性需求，对软件部署的可用性验证也是有帮助的。在进行可用性验证时，还应研究组件交互和用量分析，对组件进行逐个验证分析，以确定各个组件是否满足事先设计的要求。

不仅要验证是否满足 QoS 要求所需的资源，而且对所有可用选项进行分析，通过分析每个设计决策中的平衡点，验证资源是否能被平衡利用，即确认是否为最佳解决方案——成本最低而又能满足 QoS 要求。例如，针对可用性进行水平扩展可能会提升总体可用性，但代价是需要增加维护和维修成本 ；针对性能进行垂直扩展可能会以经济的方式提高附加的计算能力，但所提供的软件服务对这些附加能力的使用效率不高。

20.5.2　安全性验证

软件服务，多数都依赖于互联网。基于互联网的企业应用（尤其电子商务），其安全性设计和审查是至关重要的，包括安全政策和安全目标。由于云服务采用多租户的模式，多个用户访问的是同一数据库实例和同一套应用程序，所以其安全性更具有挑战性。软件系统的安全性涉及物理安全、网络安全、应用程序及其数据安全、个人安全惯例等不同方面，但从云服务平台安全性来看，主要对网络安全、应用程序及其数据安全等进行审查和验证。

（1） 网络安全的审查，主要针对防火墙、安全访问区、访问控制列表和端口访问的设置进行，以确认是否已建立有效的策略，能防止未授权访问、篡改和拒绝服务（DoS）攻击。

（2） 云服务软件系统的安全验证，涉及 Web 服务器、应用程序、数据库和数据传输等，如是否采用用户密码加密、访问权限控制、应用程序会话时效控制、数据传输加密和数据库访问会话加密等安全技术，以及确认针对口令、加密、认证、访问权限和控制等策略是否有效，是否还存在其他安全漏洞。

（3） 数据安全主要指客户信息是否以安全的方式被保存在安全的地方，比如是否加密、是否在不同地理位置有备份的数据中心以及机房本身是否安全（如是否有 24 小时录像监控，是否有可靠的预防火灾、水灾的措施）等。

针对云服务的安全性，需要验证和管理好安全证书，使用适当协议保护数据，避免引入安全漏洞，在各个环节保证云服务数据的安全。例如 ：

（1） 密码失效是攻击者获得信息的首选方法，而容易被猜到的密码则是主要目标，采取的对策是创建一个高强度密码，密码必须由大小写字母、数字、特殊字符等构成，长度不低于 9 位、每个月需要更换一次密码、不能使用旧密码等。

（2） 使用不安全协议的应用往往会容易导致数据被截取、暴露给他人，例如远程访问的 Telnet、文件传输协议（FTP）、超文本传输协议（HTTP）、邮件接收的邮局协议（POP）与互联网消息访问协议（IMAP）等。而应该选择 SSH 代替 Telnet、HTTPS 代替 HTTP、FTPS 代替 FTP 等。

（3） 当客户有能力将适用范围扩大时，也可能引入应用缺陷和安全风险。此类威胁会随具体应用而变化，但也不容忽视。

云服务模式供应商可以采用的保证数据安全的方法有很多，例如通常采用数据备份、数据镜像等手段，也可以采用磁盘阵列并对数据加密等。下面是一些具体的实例 ：

- Iron Mountain 采用数据传输加密、用户访问控制以及把数据保存在位于地下 200 英尺的数据中心等多种方法来保证数据的安全。
- Elephant Drive 通过把数据保存到多个基于硬盘的存储池来保证数据的安全，对数据制的保护已经内嵌到其产品之中，所有的数据至少被分别保存到位于不同地理位置的两个数据中心。
- AmeriVault 把客户数据保存在三个不同的地方。其中一个地点保存有两份数据，分别存放到两个不同的磁盘系统中，而第三份数据保存在上千公里之外的业务连续性站点上。
- DS3 DataVaulting 采用 EMC 的 Clariion 作为主存储设备，另外一份备份的数据保存在一个完全不同的高端磁盘系统，以保证数据恢复简单易行。它的三个数据中心中有一个专门用于保存客户数据。

对于特定的应用系统的安全要求，只有明确了安全要求、目标的情况下，才能建立适当的、可靠的安全政策。例如，为 WEB 环境建立安全目标时，应该考虑 ：

- 保护公司知识产权。
- 保护个人信息。
- 客户、合作伙伴、供应商等各方信息的分离。
- 财务交易的安全性。
- 建立对 Web 环境的信任。
- 致力于提供一种愉快而且安全的 Web 体验。

20.5.3　可伸缩性验证

可伸缩性是指增加系统容量的能力，而且要求在增加系统资源时不改变部署的体系结构。在系统需求分析、设计阶段，系统容量的预测往往只是估计值，可能与部署系统的实际情况存在较大差异，所以可伸缩性验证也是非常重要的。

软件系统的可伸缩性设计，有不同的策略，如高性能设计策略和渐增式部署策略。高性能设计策略在性能要求的确定阶段加入潜在容量，可使系统具有一定的缓冲时间来应付增长的负载。而渐增式部署策略，基于负载的要求以及评估，事先明确系统扩展的条件以及条件可能达到的时间，对每一个重大的系统扩展特定日期 / 时间有一个估计和安排，从而建立部署的整个日程表。相对来说，高性能设计策略可使系统达到高可用性、从容地制订系统扩展的方案，但其一次性预算较高。

对用户设计模式的分析，可以预估系统的负载情况。系统的可伸缩性设计和用户、用量分析有直接的关系，更确切地说，系统的可伸缩性是建立在用户、用量分析的基础之上。如表 20-2 所示 ：

表20-2 用户、用量分析项目列表

主 题	说　明
用户数量及类型	确定解决方案必须支持的用户数量，并在必要时对用户进行分类。例如： “ 企业对企业（B2B） ” 解决方案的访问用户数比较小，但每个用户访问时间长，对性能、安全性要求高 “ 企业对消费者（B2C） ” 或 “ 消费者对消费者（C2C） ” 解决方案一般会有大量访问者，操作量大、数据大，而且区域分布也比较明显

主 题	说　明
活动和非活动用户	确定活动和非活动用户的使用模式和使用比率。活动用户是指登录系统并与系统的服务进行交互的用户，系统的运行或操作性能主要关注这类用户。而非活动用户则对数据库、数据查询、存储需求影响较大
管理用户	确定用户对系统访问的权限和范围，从而对软件部署进行监控、更新和支持的用户，包括安全性技术要求和特定的用户管理模式（例如，从防火墙外部管理部署）
使用模式	确定各类用户如何访问系统，并提供预期用量目标。例如： 是否存在因用量高涨而产生的高峰期？持续时间是多少？ 正常业务时段和非正常业务时段的分布和区别 或7x24不间断服务？ 用户是否呈明显的区域分布？
用户增长	确定用户群体规模是否固定，如果用户数量具有不断增长趋势，要进行预测并增加预测值
用户事务	确定必须支持的用户事务类型，可将这些用户事务转化为使用案例。例如： 用户登录后是否保持登录状态？是否频繁登录、注销？ 用户登录后，会执行哪些任务？有什么关键业务？ 用户间的重要协作是否通过公共电子日历、Web页面或会议等来实现？
用户/历史数据	利用现有用户研究和其他资源来确定用户行为模式。应用程序的过去记录下来的日志文件可能会包含一些有用的统计数据，对估量会有较大帮助

20.5.4　通过SLA来保证质量水平

SLA 是服务品质保障协议（Service Level Agreement）的英文缩写，用来陈述服务质量、优先级和权责，也是服务提供者和客户之间的一个正式合同，包含出现故障时服务提供者和客户应采取的步骤，以及保证所提供的服务在一定百分比的时间内（如前面提到的 99.9%）是可用的，用来保证可计量的网络性能达到所定义的品质（Quality of Service，QoS）。SLA 一般都明确定义了 QoS，如服务器宕机的最长和平均响应时间，并如何利用基于因特网的工作流自动化、分发和报告技术。如果经过指定的一段时期后服务提供者还无法达到所定义的 QoS，客户就可以获得一些权利和赔偿。不同 SLA 的权利、赔偿和例外情况也是不同的，有时还包括特定说明的一些例外情况。SLA 还应该为客户包含一个退出条款 ；当服务提供方不能圆满解决经常发生的可用性、可靠性和安全性问题时，客户希望有终止协议的权利。

SLA 从客户的角度出发，把承诺的服务品质进行量化（服务质量的指标），包含下列内容 ：

（1） 目标 ，这份 SLA 所针对的产品或服务内容。
（2） 参与者，SLA 中涉及的参与者，比如服务提供商和用户。
（3） 有效期，SLA 覆盖的时间段。
（4） 限制条件，为了获得所要求的服务级别而必须执行的步骤。
（5） 服务级别目标，服务提供者和用户都认可的服务级别。服务级别中的每一个方面，如可用性、可靠性、延迟等等，都要达到一定的级别。
（6） 服务描述，如何用服务操作的术语来描述和度量一个服务的性能。
（7） 义务，当服务不能满足 SLA 中规定的级别时适用的意外处理条款（也就是罚款、补救措施、奖励终止条件等等）。
（8） 异常情况，SLA 中不能包括的内容。
（9） 管理机构，对每一个 SLA 过程负责的个人或组织。

SLA 内容重要，更重要的是其过程，即 SLA 的生命周期，经历从 “ 产品 / 业务开发、协商和销售、实现、执行到评估 ” 这样一个完整过程。QoS 是服务提供商不断追求的目标，以便最大限度地提升客户满意度。

参考文献：

[1] 朱少民 . 全程软件测试 . 北京 ：电子工业出版社，2007.9

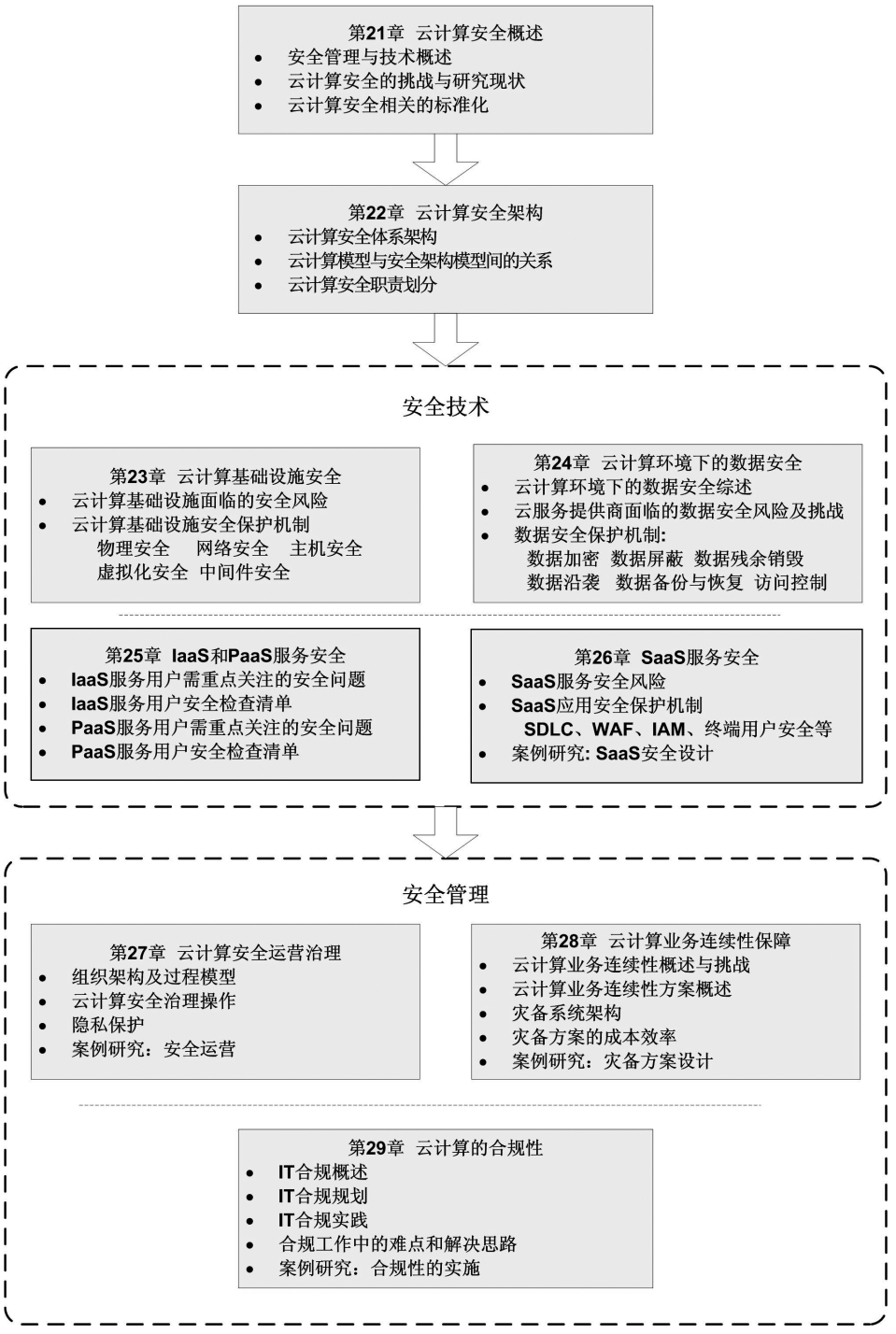
[2] 朱少民等 . 软件测试方法和技术 . 第 2 版，北京 ：清华大学出版社，2010.9

[3] 朱少民等 . 软件质量保证与管理 . 北京 ：清华大学出版社，2007.1

第四部分　安全技术与管理
Security Technology and Management

第四部分 安全技术与管理

Security Technology and Management



总图 5 安全技术与管理部分的章节逻辑结构

第21章　云计算安全概述

21.1　概述

21.1.1　什么是云计算安全

维基百科给出云计算安全（Cloud Computing Security）的定义[1]：云计算安全是计算机安全、网络安全以及广义上的信息安全的一个子域，是指用于保护云计算数据、应用和相关基础设施而部署的一系列安全策略、技术与控制措施。

从维基百科给出的云计算安全定义中可以看出，云计算安全并没有完全突破传统安全技术的范畴，但由于多用户、联合应用等服务模式的引入以及虚拟化技术的广泛采用等，又使得云计算安全在局部处理策略、技术和实施上又与传统安全范畴有所不同。

云服务的安全问题大致分为两个方面。第一个方面是云服务提供商提供的网络、系统和应用必须是安全的，必须能够防止他人盗用账号非法进入系统，造成用户数据泄密。同时，云服务提供商内部管理和流程必须规范以减少用户隐私泄露。这些都是云服务提供商需要解决的问题，并需要向用户或客户提供承诺。此外，正如自来水公司需要按照国家相关法规提供水资源服务一样，云服务提供商的行为和技术，也需要国家或行业相应的法规政策及服务标准加以约束和规范；第二方面，客户在使用云计算提供的服务时也需要增加自身安全意识，保管好自己的账户，防止他人盗取账号使用云中的服务而让你买单。另一方面，需要对自己数据的安全性和重要性进行评估，明确哪些数据是非常重要或敏感的数据不能放到云里，或需要将其加密后再放到云中。

云计算安全与传统 IT 解决方案安全不一样的地方是，传统的 IT 解决方案通常部署于用户可自行控制的安全域范围内，用户自行承担安全风险 ；而在云服务模式下由于服务模式（资源共享和多用户）的转变，并随着云服务模型、部署模式以及使用的云服务技术的不同，用户承担安全风险级别及安全控制力度都有很大的不同。而在更多时候，云计算安全的责任主体由用户转移到了云服务提供商那里。

从总体上来，云服务并不必然比传统的 IT 服务环境更为安全或更不安全，正如任何新技术一样，它会创造新的风险和新的机遇。在某些情况下，应用迁移到云中的过程提供了机构一个新的机会，可以重新审视并设计更为安全的应用程序和基础设施，以达到甚至超过当前的安全要求。而在另外一种情况下，将极其敏感或重要的数据和应用迁移到一个不够安全的云中，其风险可能会超出您的想象。

21.1.2　广义的云计算安全

人们往往将云计算的可靠性（Reliability）、可用性（Availability）和安全性（Security）混在一起谈论。但从严格意义来讲，安全性、可用性和可靠性是有区别的。

可靠性指的是指提供服务的系统能够按预期工作。可靠性强调的是提供服务的系统连续工作的能力，其评价指标可以使用概率指标或时间指标，如可靠度、失效率、平均无故障工作时间、平均失效前时间、有效度等。

可用性指的是在需要的外部资源得到保证的前提下，系统在规定的条件下和规定的时刻或时间区间内处于可执行规定功能状态的能力。换言之，可用性通常是指系统在遇到故障或问题时仍可保持提供服务的能力。在互联网环境下可用性至关重要，例如当用户访问一个网站的时候，即使服务器繁忙，也要给用户一个合理的反馈，如“系统繁忙，请稍等”，不能没有任何反应。可用性强调的是系统的容错能力。

而安全性指的是系统抵御威胁和风险、免受打探和攻击的能力。在计算机领域，安全性是保证存储在计算机上的数据不被没有权限的人盗取和访问，绝大多数安全措施都涉及到数据加密和口令。

不难看出，安全性问题可能会导致系统瘫痪或服务不可用，但可靠性或可用性问题未必均由安全性问题引发。如微软云计算平台 WindowsAzure 运行的中断和亚马逊的“简单存储服务”（S3）的两次中断等都属于服务的可靠性问题。当然这类问题的背后，有可能是微软、亚马逊的安全措施没有到位，遭受了黑客的攻击所致；也可能是系统自身的可靠性没有得到充分保证所致。但其表现出来的问题不是传统意义上的安全问题，而是服务的可靠性或可用性问题。因此，从用户服务感知的角度考虑，把包括云计算安全性、可靠性和可用性的概念统一纳入到广义的云计算安全范畴。

在这里还要说明一下，云计算安全与云安全的区别，这是两个业界容易混淆或产生歧义的概念。云计算安全即指云计算服务安全（the Security of Cloud Computing Service），云安全（Cloud Security）是指云安全服务（the Security Service based on the Cloud Computing），是“云计算”技术在信息安全领域的应用，如一些传统的反病毒安全厂商，他们利用云计算平台向用户提供的各类病毒防护产品或服务。

为便于后续描述，这里给出美国国家标准及技术研究所（NIST）在其云计算参考架构[2]里给出五个核心云服务相关角色定义，它们分别是：

- 云服务消费者（Cloud Consumer）：使用云产品或服务的个人或机构，文中简称为 CS；
- 云服务提供商（Cloud Provider）：提供云服务（包括软件、平台或基础设施）的公司或个人，文中简称 CP；
- 云服务运营商（Cloud Carrier）：负责实现数据及其他信息在分布式式网上进行传输的机构。文中简称为 CC；
- 云服务代理商（Cloud Broker）：是云服务消费者与云服务提供商之间的中间人，简化云服务提供，并创造增值云服务；
- 云服务审计者（Cloud Auditor）：指可以对云服务、信息系统运维以及云服务实现的性能与安全性进行独立评估的第三方。

需要说明的是，在上述角色中，在不同的服务模型和部署模型下，云服务消费者、云服务提供者与云服务运营商等角色可能是同一实体，而云服务代理商则不一定每一种模式里都有。如对于自建、自营的大型私有云而言，云服务消费者也是云服务运营者，也并不需要云服务代理商；而对于电信运营商自建自营对外提供的公有 IaaS 服务而言，服务提供商与服务运营商都是电信运营商同一实体。

21.2　云计算安全的挑战和研究现状

21.2.1　云计算安全研究焦点域

云服务改变了服务提供模式，但并没有颠覆传统的安全模式。所不同的是，在云服务时代，安全设备和安全措施的部署位置有所不同；而随着云服务模型的不同，安全责任的主体也会随之发生较大变化。原来，用户自己要保证服务的安全性，现在很多情况下是由云服务提供商来保证服务提供的安全性。同时，对于基于互联网的云服务，服务的可靠性和可用性也显得更为重要。从总体上来讲，解决云计算安全问题的办法和解决传统的信息服务的安全问题一样，也是策略、技术和管理（人）三个要素的组合。

云计算安全研究域从关注的对象来说可分成两大类：

一类是云服务提供商（CP，如 SaaS、PaaS、IaaS 服务提供商）面临的安全风险及相关技术实施、处理策略和控制措施等。通常情况下，CP 要确保云服务基础设施是安全的，同时也要确保用户的数据以及应用也是受到保护的。即服务运营商必须确保为用户提供应用服务的基础设施，如网络、主机和各类基础服务中间件是安全可靠的（不易因各种攻击行为而瘫痪或提供非期望的服务），必须确保用户存储在服务运营商侧的数据是安全可用的（信息不易泄露且数据总是可用），必须确保提供给用户的应用系统是安全健壮的（只允许合适的用户访问合适的服务或操作合适的功能，且不易受到攻击而瘫痪或提供非期望的服务等）。

另一类是云服务消费者（CS）面临的安全风险及相应的技术实施。CS 必须明确知道哪些安全职责应由云服务提供商来提供，哪些安全职责应由自己来承担。同时还必须具备较强的安全防范意识，并应通过客户端侧安全技术实施来保障其敏感信息不被泄露。对于云服务这种用户信息高度集中于服务提供商侧的服务模式而言，还有一项更重要的是，服务提供商和用户自身还应确保用户隐私信息不因双方自身原因而遭到泄密。

云安全联盟 CSA 则把云计算安全研究域分成十三个焦点域[3]，分别是：云计算架构框架、治理和企业风险管理、法律与电子证据发现、合规与审计、信息生命周期管理、可移植性和互操作性、传统安全业务连续性和灾难恢复、数据中心运行、应急响应通告和补救、应用安全、加密和密钥管理、身份和访问管理、虚拟化。

21.2.2　国内外云计算安全技术研究现状

在 IT 产业界，各类云计算安全产品与方案不断涌现。例如，SUN 公司发布开源的云计算安全工具可为 Amazon 的 EC2、S3 以及虚拟私有云平台提供安全保护。SUN 公司提供的云计算安全工具包括 OpenSolaris VPC 网关软件，能够帮助客户迅速便捷地创建一个通向 Amazon 虚拟私有云的多条安全的通信通道；Sun 公司为 Amazon EC2 设计的安全增强的 VMIs，包括非可执行堆栈、加密交换和默认情况下启用审核等安全功能；云安全盒（cloud safety box），使用类 Amazon S3 接口，自动对内容进行压缩、加密和拆分，简化云中加密内容的管理等。

微软为云计算平台 Azure 筹备代号为 Sydney 的安全计划，旨在帮助企业用户在服务器和 Azure 云之间交换数据，以解决虚拟化、多租户环境中的安全性。EMC、Intel、VMware 等公司联合宣布了一个“可信云体系架构”的合作项目，并提出了一个概念证明系统。该项目采用 Intel

的可信执行技术（trusted execution technology）、VMware 的虚拟隔离技术、RSA 的 enVision 安全信息与事件管理平台等技术相结合，构建从下至上值得信赖的多租户服务器集群。开源云计算平台 Hadoop 也推出安全版本，引入 Kerberos 安全认证技术，对共享商业敏感数据的用户加以认证与访问控制，阻止非法用户对 Hadoop clusters 的非授权访问。

21.2.3　云计算模式下信息安全技术演进趋势

云服务使得 IT 资源 、数据资产和应用系统高度集中，这对以赢利为目标的黑客们将是一个极大的诱惑，同时也给他们实施攻击创造了便利条件。由于信息和数据的高度集中，黑客们只要攻下一“ 点 ”，便可牟取暴利。

“ 魔高一尺，道高一丈 ”，信息安全技术的发展本身就是信息安全防护技术和信息安全攻击技术不断博弈的过程。在云服务模式下，随着攻击技术的发展以及 IT 服务模式的转变、虚拟化等技术的广泛应用，信息安全技术也由边界防护、病毒防护为主构建的纵深防御体系向以“ 防（边界防护和主机加固及数据加解密）、测（基于网络流量和应用内容的检测监控）、控（身份认证和访问控制技术）、管（安全审计与日志管理）、评（漏洞扫描与风险评估）” 相融合的主动防护体系转变，因此，信息安全技术将由传统的病毒防护、入侵检测、防火墙、UTM 等技术向深度包检测技术、可信技术、虚拟化安全技术、基于云模式的病毒防护技术、同态加密技术、终端安全接入技术以及 Web 应用安全技术等多种新型信息安全技术演进。

21.3　云计算安全相关的标准化组织

近两年，云计算已成为国际上标准化工作热点之一。当前国际上已有 70 余家组织和团体在进行“ 云计算 ” 相关的标准化研究工作，其中超过 40 多个宣称有包含云计算安全相关方面的议题，如欧盟的 ENISA（Europe Network and Information Security Agency）也发布了云安全风险评估和云安全框架等相关研究成果，云安全联盟（Cloud Security Alliance，CSA）则因其《云安全联盟指南 2.1》的正式发布及其在云计算安全方面研究的广度和深度获得业界广泛关注。下面主要介绍这一标准化组织的一些研究概况。

21.3.1　CSA

云安全联盟（Cloud Security Alliance，https://cloudsecurityalliance.org/）是一个非赢利性组织，成立于 2009 年 4 月 21 日旧金山的 RSA 大会上。自成立后，CSA 迅速获得了业界的广泛认可。现在，CSA 和 ISACA、OWASP 等业界组织建立了合作关系，很多国际领袖公司成为其企业成员。

截止到 2011 年 7 月，CSA 企业成员达到 52 个，名单中涵盖了国际领先的电信运营商、IT 和网络设备厂商、网络安全厂商、云计算提供商等。值得注意的是，中国领先的专业安全公司绿盟科技成为中国、乃至亚太地区第一个企业成员。云安全联盟成立的目的是为了在云计算环境下提供最佳的安全方案。

2009 年 12 月 17 日，云安全联盟发布了新版的《云计算关键领域安全指南 V2.1》，代表云计算和安全界对云计算及其安全保护认识的重要提升。新版安全指南聚焦于十三个焦点安全域，较为真实地反映了最新的业界最佳实践和观点，提供了大量非常详细、务实的建议，很容易转化为

项目的检查列表等形式，方便项目实施参考。同时在技术层面上明确阐述了法律、电子证据发现（D3）以及虚拟化（D13）方面的建议指南。新版 CSA 安全指南关于数据的可移植性和互操作性（D6）方面也提供独特的建议。

除《云计算关键领域安全指南 V2.1》外，云计算安全联盟还发布了《云计算面临的严重威胁》、《云控制矩阵》等研究报告，并发布了云计算安全定义。这些报告从技术、操作、数据等多方面来强调云计算安全的重要性、保证安全性应当考虑的问题以及相应的解决方案，对形成云计算安全行业规范具有重要影响。

21.3.2　其他相关标准组织

2009 年底，国际标准化组织 / 国际电工委员会、第一联合技术委员会（ISO/IEC，JTC1）正式通过成立分布应用平台服务分技术委员会（SC38）的决议，并明确规定 SC38 下设云计算研究组。

国际电信联盟 ITU-TSG17 研究组会议于 2010 年 5 月在瑞士日内瓦召开，确定成立云计算专项工作组，旨在达成一个“ 全球性生态系统 ”，确保各个系统之间安全地交换信息。工作组将评估当前的各项标准，以推出新的标准。云计算安全是其重要的研究课题，计划推出的标准包括《电信领域云计算安全指南》。

近期，分布式管理任务组（Distributed Management Task Force，简称 DMTF）也已启动了云标准孵化器项目。参与成员主要关注通过开发云资源管理协议、数据包格式以及安全机制来促进云计算平台间标准化的交互，致力于开发一个云资源管理的信息规范集合。该组织的核心任务是扩展开放虚拟化格式（OVF）标准，使云计算环境中工作负载的部署、管理更为便捷。

最后，需要说明的一点是，IT 技术界总是“ 道魔 ”相克相生，当这些安全标准化组织正在积极研究云计算安全相关技术标准时，黑客界的“ 精英们 ”也在“ 磨拳擦掌 ”，他们在 Black Hat USA 2009 和 2010 黑客大会上，发表了云服务环境下的安全漏洞、利用云端服务作为僵尸网络控制主机的方法以及虚拟环境中密码机制的脆弱性等研究成果。从全球应用情况来看，目前云服务尚处于小规模应用阶段，因此，不管是“ 红道 ”还是“ 黑道 ”，短期内尚难发布统一的业界标准。

21.4　写作重点和章节结构

本部分的写作重点体现在以下三个方面 ：

（1）在读者侧重上，强调从服务提供商和云服务运营商的角度描述云计算服务安全相关技术实施及运营相关内容，因为云服务提供商和云服务运营商对云计算的安全负有重要职责。

（2）在章节组织上，强调系统性和全面性，本部分给出了云服务系统在生命周期各阶段应重点关注的安全实施内容 ：在规划建设阶段，云服务系统应遵循的合规法则 ；在设计实现阶段，各系统层次应关注的安全技术实施 ；在运行阶段，应组建的运营组织构架、应实施的治理操作及业务连续性保障等。

（3）在内容编排上，注重理论阐述与实践建议或案例相结合，增强可操作性与可借鉴性。

本部分章节内容安排如下（见总图 5）：

云计算安全概述（第 21 章）：描述云计算安全的概念、挑战、研究现状和标准化组织等 ；

云计算安全架构（第 22 章）：阐述安全架构模型，从总体视图上说明云计算安全层次划分及研究内容 ；

云计算基础安全（第 23 章、第 24 章）：

- 基础设施安全 ：从云服务提供商的角度，描述云服务基础设施安全技术实施 ；
- 数据安全 ：从云服务提供商的角度，描述云服务环境下数据安全技术实施 ；

云计算服务安全（第 25 章、第 26 章）：

- IaaS 和 PaaS 服务安全 ：从云服务消费者的角度，描述 IaaS 和 PaaS 存在的服务风险及应对措施 ；
- SaaS 服务安全 ：从云服务提供商的角度，描述 SaaS 服务安全技术实施 ；

云计算安全管理（第 27 章、第 28 章、第 29 章）：

- 安全运营治理 ：从云服务运营商的角度，描述云计算安全运营所需的组织架构、治理操作及隐私保护 ；
- 业务连续性保障 ：从云服务运营商的角度，描述云计算安全业务连续性挑战及灾备系统方案设计 ；
- 合规性 ：从云服务运营商和云服务提供商的角度，描述云服务的合规性概念、规划及实践。

21.5　本章总结

本章主要介绍了 云计算安全的基本概念、广义的云计算安全涵义，以及业界经常与之混为一谈的云安全服务的概念及两者间的区别。在此基础上，本章结合业界不同的云计算安全研究域划分方式，对云计算安全研究领域进行了简要说明，为后续各章深入展开云计算安全技术探讨奠定基础。

为便于读者对云服务模式下的信息安全技术有一个前瞻性了解，本章也扼要介绍了信息安全技术的演进趋势。由于云服务模式是一个新的服务模式，加之信息安全技术的快速发展，这里列举的技术只是作者自己的观点和认识，目的是希望能够起到抛砖引玉的作用，引发读者更多的思考与关注。作为补充性介绍，本章还介绍了与云计算安全关系比较密切的标准化组织 CSA、ITU-TSG17 和 DMTF。关注云计算安全标准化的读者可以按图索骥，在这些组织提供的网站信息里找到自己感兴趣的内容。

阅读完本章，如果读者对云计算安全已经在脑海中形成了一个概念并知道其主要研究内容，那就达到了本章的目的。如果还有一些问题产生，后续的章节或许会更具体地帮你解决这些问题。

参考文献：

[1] Cloud Security, Wikipedia，http://en.wikipedia.org/wiki/Cloud_security

[2] Cloud Computing Reference Architecture，Special Publication 500-292，NIST

[3] CSA ：Security Guidance for Critical Areas of Focus in Cloud Computing V2.1，2009.12

第22章　云计算安全架构

22.1　云计算安全体系架构

传统的信息安全是人、管理和技术的统一体。从总体上讲，云计算安全也是如此，也是由组织机构、运营管理、技术实施、安全策略等构成的一整套安全体系，该体系并能基于业务风险从规划、执行、控制到持续改进保持动态循环。

图 22-1 给出云计算安全体系架构模型。该体系架构旨在从整体上描述云计算安全实施涉及的要素及需要关注的层面，以便于后续章节展开描述之前构建一个对云计算安全的整体概貌。

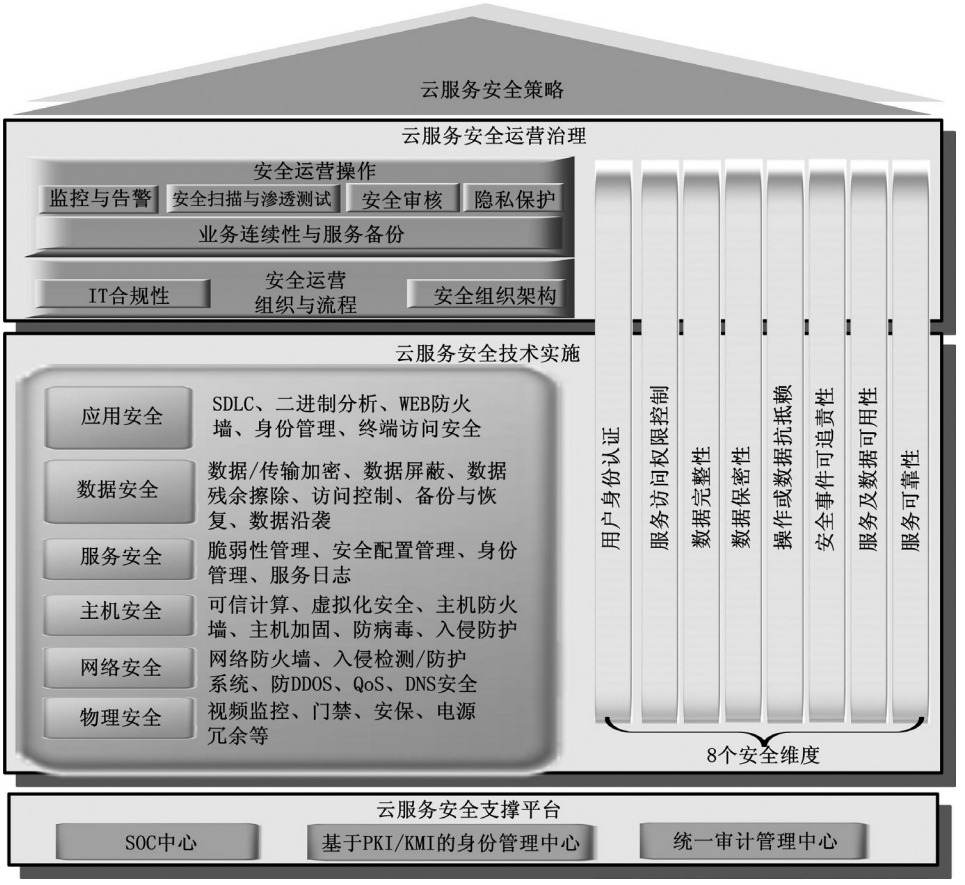


图22-1　云计算安全体系架构

云计算安全体系架构从底至上包括三个层面，分别是云计算安全支撑平台层面、云计算安全技术实施层面、云计算安全运营治理层面，其中云计算安全运营层面包括云计算安全治理、云计算业务连续性保障及云计算 IT 合规性三个层次。云计算安全架构的各个层面 / 层次是相辅相成、相互作用的，表现在特定的云计算安全组织机构需要按照一定的安全机制（安全策略），使用相应的技术工具（安全技术实施与安全支撑平台）进行安全操作（安全治理），并通过技术实施、管理架构、策略与操作的持续调整最终实现云计算全过程安全的战略目标。

1. 云计算安全支撑平台

云计算安全支撑平台层是云计算安全技术实施的基础。云计算作为一种规模化提供的服务模式，为更好地保障云计算全流程、整个生命周期过程中的安全，通常 CP 会建立一套相对独立而完善的云计算安全支撑平台，该平台可能包括安全监控中心、基于 PKI/PMI 技术的身份管理中心及审计管理中心，提供集中的事前认证、事中监控、事后审计。

上述的云计算安全支撑平台也可以利用 CP 已有的基础设施体系，但需要根据云计算系统安全接口要求进行相应的改造开发，如统一审计管理中心需要支持云计算系统相关审计信息接口要求。这部分内容已有相对成熟的技术和应用实践，本书不做详细讨论，感兴趣的读者请参考相关资料[1-8]。

2. 云计算安全技术实施

云计算安全技术实施是整个云计算安全架构的核心内容，也是本部分后续章节重点讨论的内容。在这个层面上对云计算安全在纵向上分了三个层次 ：

第一个层次是基础设施安全层，包括物理与机房安全、网络安全、主机安全及中间件安全等，该层面的安全技术部分仍可延用传统信息服务安全技术，但需要在重点关注虚拟化技术引入给网络、主机系统带来的安全问题。该层的具体技术实施将在本部分的第 23 章 “ 云计算基础设施安全 ” 一章里进行详细描述。

第二个层次是数据安全，这个层面主要讨论数据存储、数据传输、数据隔离及数据应急恢复等方面的技术实施，如数据加密技术、虚拟专用通道技术、数据访问控制、残留数据覆盖擦除及数据备份等技术。本部分的第 24 章 “ 云计算环境下的数据安全 ” 将对数据安全技术实施进行具体说明。

第三个层次是应用服务层。由于云计算模式下用户身份的多样性及复杂性，以及攻击导致的破坏性影响程度，已经使得云计算应用层次的安全防护任务变得更加艰巨，也更具有挑战性，需要从应用设计和应用运行两方面考虑安全技术实施，如应用程序安全开发生命周期管理、二进制代码分析、身份认证、访问控制、WEB 应用防火墙、应用扫描器等技术，具体技术实施将在本部分的第 26 章 “ SaaS 服务安全 ” 进行说明。作为补充，第 25 章 “ IaaS 和 PaaS 服务安全 ” 则从 IaaS 和 PaaS 服务用户的角度描述了其应该关注的安全技术实施。

3. 云计算安全运营

云计算安全运营包括云计算安全运营治理、云计算业务连续性及云计算安全合规性。

云计算安全运营治理包括云计算安全组织架构、云计算安全管理策略、云计算安全治理操作、云计算用户隐私保护等。云计算安全组织架构是提供高效而快速的云计算安全保障的基础 ；云计算安全管理策略则提供了规范 CP 相关人员行为、保护云计算信息安全的操作指南，是云计算安全体系架构中重要组成部分，这些策略在监查并防范社会工程学攻击时尤其有效。云计算安全治理操作包括安全监控、威胁管理、安全审计等具体安全运营操作。隐私保护成为公有云计算模式下影响用户使用云计算的最大隐忧，主要原因是用户对哪些隐私需要保护、如何保护等内容缺乏清晰地了解。上述这些内容将一并在第 27 章 “ 云计算安全运营治理 ” 中进行讨论。

云计算的可靠性与可用性是影响用户体验的重要因素，也是服务提供商品质服务能力的重要体现。本部分内部将在第 28 章“ 云计算业务连续性保障 ”一章中进行描述。

任何一种 IT 服务的安全实施都是一个持续改进、不断完善的过程，云计算的安全管理体系也不例外。安全管理合规性标准提供云计算系统建设与运维管理与技术实施的符合性对照标准，为云计算安全管理过程规范化、流程合理化提供了标准参考模型（如 ISO 27001:2005 PDCA 过程模型 [9]），这部分内容将在第 29 章“ 云计算的合规性 ”一章里进行说明。

云计算纵向上三个层面的技术实施和三个层面的运营治理最终是满足横向上 8 个安全维度的需求，即用户身份认证、服务访问控制、数据完整性、数据保密性、操作或数据抗抵赖、安全事件的可溯源性、服务及数据的可用性以及服务的可靠性。当然，具体到某个层面，并不一定都会包含这 8 个维度的安全需求，如在应用服务层，则主要涉及用户身份认证和服务访问控制，对数据的完整性、保密性、抗抵赖性等安全需求则主要由数据层面的安全技术实施来完成。同时，某一纵向层面的安全需求可能需要多个层面的技术共同实施方能满足，如服务的连续性（可靠性）将会涉及到多个层面的技术实施。

特别说明的是 ：上述安全技术实施将涵盖 CSA D7-D13 运行域内容，包括传统安全、业务连续性和灾难恢复、数据中心运行、事件响应、通告和补救、应用安全、加密和密钥管理、身份和访问管理、虚拟化。而安全运营治理内容涵盖 CSA 治理域的 D2（IT 治理和企业风险）、D3（法律与电子证据发现、D4（合规性和审计）、D5（信息生命周期管理）研究焦点域。CSA 的 D6 焦点域（可携带性与可交互性）涉及多个云计算商间的业务兼容与可移植性，本书不做深入讨论。

22.2　云计算模型与安全架构模型间的映射关系

为更清楚地说明云计算模型与云计算安全架构模型间的对应关系，便于在云计算架构设计及运营管理时更好地实施安全技术与管理，在参考 CSA 云计算安全指南 [10] 基础上给出了云计算模型与云计算安全架构模型间映射关系，如图 22-2。

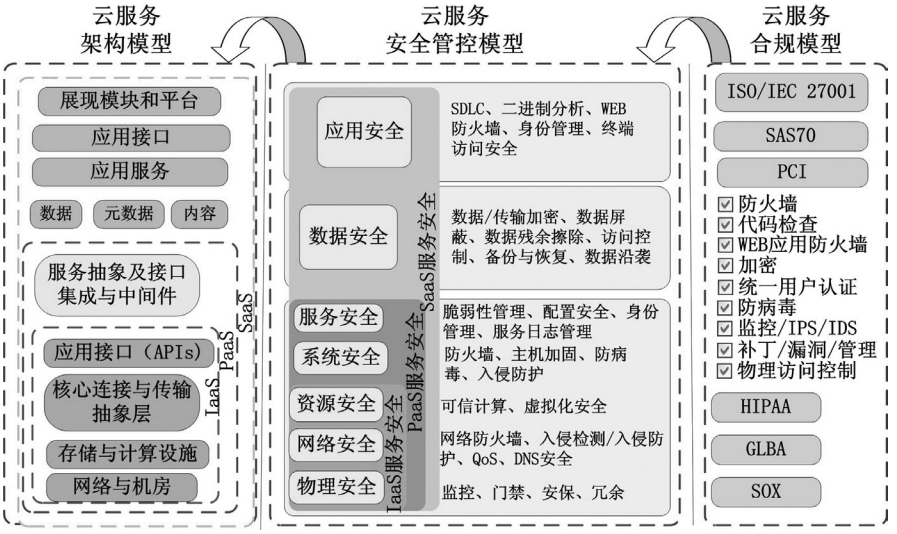


图22-2　云计算架构模型与云计算安全模型间的映射关系

图 22-2 描述了服务架构模型与云计算安全管控模型间的对应关系，而右侧的合规模型或监管要求则为这些管理技术实施提供参考标准、指南或要求，以指导或强制性要求 CP 应该采取哪些安全管控措施，并确定如何对待存在的安全差距或残余风险，即是接受、转移还是降低等。这些安全服务规范将在本部分第 29 章“ 云计算安全合规性 ”一章里加以说明。

云计算分为三种类型，为便于后续章节阐述，从安全技术实施层面上讲，将云计算安全也分为三个层面。但需要说明的是，图 22-2 中的三个技术层面的安全实施与服务模型并不一一对应。

22.3　云计算安全职责划分

根据云计算服务交付模式（SPI）和服务级别协议（SLA），云计算的安全责任范围将落在 CS 或 / 和 CP 身上，关键是要明确哪些安全责任由 CS 来承担，哪些安全职责由 CP 来承担。不同的云计算模型，安全职责的责任主体有很大不同，如图 22-3，这些责任主体可能是 CS、CP（或第三方）：

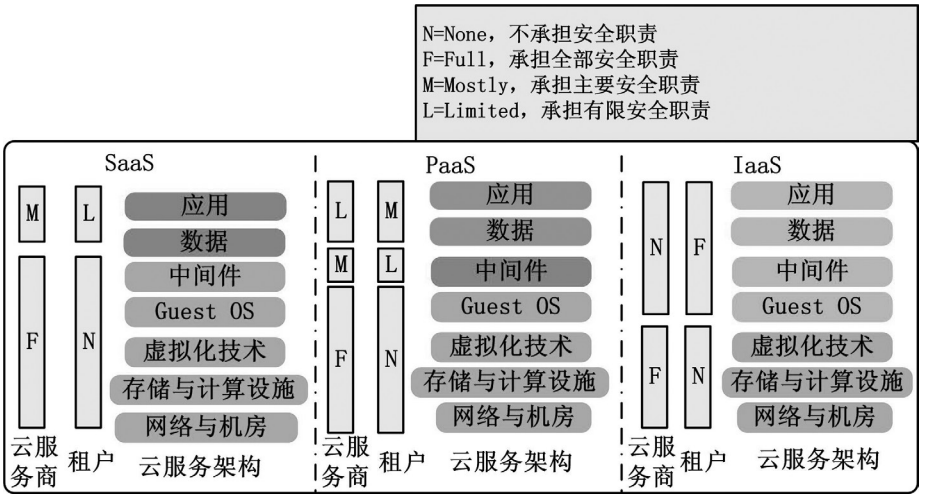


图22-3　云计算安全职责主体划分

（1）IaaS 服务的硬件和网络层通常是不暴露给用户的，IaaS 提供商需保证这些基础设施的安全，包括物理安全、环境安全和虚拟化实施层的安全，如 Amazon 的 AWS EC2 服务，服务供应商需要负责直到 hypervisor 层面的安全职责。通常情况下，IaaS 提供商不知道客户在其提供基础设施上部署的应用及其运行环境，客户需要对云主机之上的应用及其运行环境的安全负责，这包括操作系统、应用和数据等。当然，用户也可将安全职责委托给 CP 或第三方来承担。此外，IaaS 服务提供商也应对 CS 的应用数据进行安全检查，避免一些安全事件发生，如 CS 可能执行病毒程序破坏硬件基础设施等。

（2） PaaS 服务最大限度地抽象了底层基础设施，并将中间件封装成一种容器提供给用户，这些容器在简化应用开发的同时也限定了用户对底层资源的交互行为，它提供了用户使用平台支持的语言进行设计、开发、测试、部署和支持定制应用程序运行的集成环境。用户可在云基础设施之上部署用户创建或采购的应用。因此，PaaS 服务安全包含三个层次 ：

① 第一个层次是 PaaS 平台基础设施安全，该层完全由 PaaS 服务提供商负责，包括机房环境、硬件、网络、主机安全。

② 第二个层次是平台中间件服务安全，该层主要由 PaaS 服务提供商负责，但用户要配合完成有限的安全职责，理解并配合 CP 采取的安全措施的实施，如针对 PaaS 服务平台外部网络主 机或基础设施的监控行为，以及针对服务平台所做的安全配置管理，包括运行时引擎的升级，以及变更、发布和补丁管理等，以及采用满足平台访问控制要求的终端接入技术来访问平台服务等。

③ 第三个层次是客户基于 PaaS 平台部署的应用安全，该层主要由用户负责，用户应理解其应用对 PaaS 服务环境安全性的依赖，要求 CP 通过 API 提供一系列的安全功能来确保其开发应用的整体安全性，包括用户认证、SSO、授权、以及 SSL 或 TLS 支持等，PaaS 服务提供商则予以配合实现。

（3） SaaS 服务抽象层次最高，通常只向用户暴露基于软件的应用服务，如电子邮件、在线办公等，因此 SaaS 服务提供商需要负责管理和维护整套应用，相应地，SaaS 服务体系化的安全包括服务安全性、合规性、可用性等都主要由 CP 来承担，客户通常只需负责操作层面的安全功能，如访问控制和身份验证等。

从上述分析可以看出，从 IaaS 模式到 SaaS 模式，资源或服务的抽象层次越来越高，CP 在获得更大的对资源或服务控制力度的同时，对云计算安全承担的职责范围也越来越大，面临的安全风险也相应增加，因此，一些 CP 通常倾向于通过修订服务协议（如 SLA）来规避这方面的风险，所以对安全敏感型的用户而言，在签订客户服务协议时需要审慎考量 CP 的安全服务承诺能否达到自己业务安全保障要求。相应地，从 SaaS 模式到 IaaS 模式，用户在获得更大的资源掌控权和服务延展性的同时，其承担的安全职责也增大，面临的安全风险也相应增加。

最后需要说明一下部署模式与安全职责的关系。相比公有云模式，私有云用户对资源或服务拥有最大的控制权限，相应地，私有云用户需要承担的安全职责也大得多。同时，私有云模式下云计算安全职责划分还与基础设施放在哪里有关，如果放在用户处，则云计算主要由用户自己或外包方承担 ；如果是在 CP 的数据中心，则 CP 需要承担相应环境及管理上的安全职责。而在混合云和社区云计算模式下，云计算安全的职责划分将会更加复杂，但总体原则是，云计算安全职责划分与用户 / 提供商对云计算资源的掌控力度相适应，掌控力度越大，承担安全职责也越大。

理解不同服务模型对安全实施带来的影响以及安全职责划分对于 CS 或 CP 理解并处理相应的安全风险是至关重要的。后续章节将仅从安全实施的角度进行说明，不再强调哪部分安全职责应该由 CS 还是 CP 承担。

22.4　本章总结

云计算作为信息服务的一种新模式，仍然要遵循信息安全管理体系架构基本规律，即从体系架构上它也包含了技术、管理和人三个安全要素，并是在相应管理策略和管理流程约束下持续改进的循环过程。在云计算安全体系架构模型里重点介绍了云计算安全技术实施层面三个层次，包括服务设施层、数据层和应用层安全。

云计算安全体系是云计算架构视图中的纵向业务支撑层中的服务安全保障层，它与云计算架构横向上四个层次（展现、应用、数据、中间件、基础设施）存在相应的映射关系。从这种映射关系可以看出不同服务模式下应重点关注的安全技术实施。

不同云计算部署模型下，CS 和 CP 的安全职责也不同。总体来讲，对资源 / 服务的掌控力度越大，承担的安全职责越大 ；同一部署模式下，从 IaaS 到 SaaS，服务提供商拥有管理控制权越大，相应地，承担的安全职责也越多。

参考文献：

[1] ISO/IEC9594-8:2000 The Directory: Public-key and Attribute Certificate Frameworks

[2] United States General Accounting Office, Status of Federal Public Key Infrastructure Activities at Major Federal Departments and Agencies. Washington, D.C., GAO, 2003, 7

[3] 荆继武，林璟锵，冯登国 :《PKI 技术（信息安全国家重点实验室信息安全丛书）》，科学出版社，2008 年 5 月

[4] 李明柱，PKI 技术及应用开发指南，2002.06，http://www.doc88.com/p-805812984452.html

[5] ISO 9594-8/ITU-T Rec. X.509 :2005 The Directory: Public-key and Attribute Certificate Frameworks

[6] D.W. Chadwick，A. Otenko，“ The PERMIS X.509 Role Based Privilege Management Infrastructure ”， Future Generation Computer Systems，936（2002）1-13，Elsevier Science BV. , December 2002.

[7] Building a Successful Security Operations Center http://wenku.baidu.com/view/ff0a55e9856a561252d36f2b.html，

[8] 安全审计 , http://wenku.baidu.com/view/817b8707e87101f69e31953b.html，

[9] ISO 27001:2005 Information Technology-Security Techniques-Information Security Management System-Requirement

[10] CSA :Security Guidance for Critical Areas of Focus in Cloud Computing V2.1，2009.12

第23章　云计算基础设施安全

23.1　云计算基础设施面临的安全风险

本章讨论的服务设施是指基于云计算 SPI 模型，用以构成 IaaS、PaaS 和 SaaS 服务的基础资源及平台服务软件，包括数据中心物理环境、网络环境、计算和存储资源、主机系统以及数据库、应用服务器、集成运行环境等中间件。

云计算将资源及数据通过集中共享的方式提供，这一方面给 CP 带来了规模化效益，也降低了 CS 的 IT 投入成本和进入门槛，但正是因为这种大集中，也让黑色产业链上的每一个实体更加兴奋，他们似乎看到了更诱人的 “ 前景 ”，因为他们一旦成功攻入某一台服务主机，将会获得可观的经济利益，或制造更大的轰动效应。因此，在这种服务设施高度集中（无论是逻辑上还是物理实体上）的环境下，云计算设施的安全风险无论是对 CP 还是 CS 而言都不可小觑。总结起来，云计算设施存在的安全风险有以下几点 ：

1. 高度集中的云计算设施更易成为攻击者们攻击的目标

随着企业传统应用向公有云的迁移及私有云向公有云的融合与过渡，云上将会承载大量的企业重要数据（如客户资料等），这对攻击者或非法竞争者来说具有极大的诱惑力。而在这种全球皆可访问的云计算架构里，更为分布式攻击者创造了一个有利的条件。攻击者只要找到虚拟化管理核心单元，对于一个安全设施不够严密的云计算数据中心（包括用户侧的安全接入与安全意识），攻击者可以混杂于众多 CS 之间轻而易举地拿走用户的宝贵资料。而这种攻击将对 CP 来说具有更大的破坏性，因为这种深入到服务设施层的攻击与控制，影响的层次更深（从 IaaS 到 SaaS），破坏的范围更广（从 IaaS 用户到 SaaS 用户），而攻击者的获利空间也更大。

2. 虚拟化技术的引入及网络结构的变化使得防护实施更加困难

云计算设施的虚拟化使得多个组织的应用可能位于同一台物理设备上，这使得基于硬件的物理隔离的传统安全措施（如防火墙）已不再能防范同一台物理主机上的虚拟机间的攻击。同时，识别系统的安全状态和非安全虚拟机的位置也变得更具有挑战性，因为在虚拟化服务环境里，不管虚拟机的位置在哪，入侵检测和保护系统（IDS/IPS）都必须能检测到大量的虚拟机层面的恶意行为，但对传统的以网络真实流量入侵防范为主的网络安全设备而言，将是一个新的挑战。此外，虚拟化动态迁移使得安全维护工作的连续性及记录的可追溯性变得更加困难，因为在云计算环境下，不同物理机间的虚机克隆和镜像分发已成为经常性的工作。

同时，对服务的管理行为也不再局限于传统企业网络的可控连接，而是通过开放的互联网方式，这些都给服务设施带来极大的风险，需要对系统进行严密监控和访问权限控制，稍有疏漏，就会把服务设施暴露于黑客的攻击之下。

3. 安全防护职责的割离使得云计算设施安全风险增加

虚拟化服务环境下操作系统和应用数据位于共享的物理设施上，企业用户可以要求服务提供商提供可信和可审计的系统、文件和行为监控记录。但对于公有云模式的 IaaS 服务，云计算和存储资源安全职责如系统的补丁管理和病毒防护是由用户自己来承担，而不是 CP。而在基于虚拟

化技术部署的云计算环境中，位于同一物理主机上的不同虚拟机操作系统往往各异，部署应用也各异，用户更是千差万别，只要有任何一个用户疏于管理，导致其系统存在漏洞或脆弱性，这个系统将很可能成为攻击整台物理主机甚至整个虚拟环境的跳板，这样很可能使得 CP 的安全实施功亏一“机”。

23.2　云计算基础设施安全保护机制

23.2.1　物理安全

IaaS 服务可以看成是传统 IDC 服务的细粒度化。因此对于提供 IaaS 服务的基础设施，可以从两个层面来分析其安全：一方面是 IDC 物理环境安全，这个层面与传统的 IDC 服务对安全的需求没有太大区别；另一方面，是实现传统 IDC 服务细粒度化的使能层，包括网络层和主机层。由于服务模式、运营模式和云计算技术的引入，除需要考虑传统的 IT 服务安全，还要考虑这些新变化引发的安全风险。

对于传统数据中心，当人们谈到其安全时，第一想到的往往是物理安全。对于面向规模用户的云计算数据中心，物理安全是云计算安全的基础。那么什么是物理安全呢？美国国家安全局（NSA）下属部门国家计算机安全中心（NCS）在计算机安全术语词汇表 [1] 里给出物理安全的定义是：“应用物理屏障和控制程序作为对抗资源和敏感信息威胁的预防措施或对抗措施。”

参照上述定义，可以看出，在物理环境上，云计算与传统的 IT 服务对安全的要求并没有什么特殊，都需要实施物理防损坏、防盗走、防非法进入等防护措施。从提供云计算的 IDC 环境来看，物理安全主要包括三层：物理环境、机架 / 机笼和硬件设备、离站磁带 /DVD。这三个层面的安全实施更多的是管理制度与安全策略的制定与实施，以下对这三层安全策略的简要说明：

对于物理环境，需要核查以下事项：

- 数据中心所处的外部环境，包括附近的环境和物理危害；
- 要考虑站点检查时的内部检查点，如清洁能源、安全区域和制冷设备；
- 建筑的物理属性，如设备装卸区、供水线路和电网；
- 访问控制，以防止入侵者进入大楼。

对于机架 / 机笼安全，可以参照以下实施策略进行检查：

- 机架 / 机笼应可以加锁；
- 使用开锁的钥匙需要授权；
- 使用钥匙应记录在案；
- 相机不允许带入到机柜区。

对于硬件设备安全，应该遵循以下准则：

- 得到授权后方可移动硬件设备；
- 移动所有硬件设备应该用资产标示号来跟踪记录；
- 生产用硬件设备应隔离开，并在预先规定的安全时间内移走；
- 存有敏感信息的服务器应物理加锁。

对于用于异地备份用的离站磁带 /DVD/CD 媒介安全，应遵循如下准则 ：

- 所有的媒介应该放在一个安全的箱子里 ；
- 移动媒介应该记录在案 ；
- 所有的生产数据要备份，以防丢失。

判断一个云数据中心物理安全性是否达标，最好的方式是看数据中心是否有 SAS 70 合格性认证。关于 SAS70 的合格性认证具体请参考第 29 章 “ 云计算的合规性 ” 一章中的相关介绍。

23.2.2　网络安全

1. 传统网络安全防护技术与产品

云计算环境下网络包括三个层面，即数据中心网络、跨数据中心的互联网络以及泛在的云用户接入网络。

- 数据中心网络是指连接云计算数据中心云计算器设备、存储设备及各层网络设备的内部局域网，以及用于连接各虚拟化主机的边缘虚拟交换网络，如分布式虚拟交换机、虚拟桥接及 I/O 虚拟化等。
- 跨数据中心的互联网络是指各云数据中心之间的互联网络，以实现数据中心间的异地灾备、数据迁移、多数据中心间的资源优化及多数据中心的混合业务提供等。
- 泛在的云用户接入网络用于数据中心与云终端用户之间的互联，为公众用户或企业用户提供云计算。

在这三个层面的网络里，数据中心的网络安全是云计算安全的基础。跨数据中心的互联网络安全和泛在的云用户接入网络将在此基础上通过增加数据传输安全通道得到保证，关于安全通道将在第 24 章 “ 云计算环境下的数据安全 ” 介绍。在这里仅针对数据中心网络安全技术实施进行分析。

对于云数据中心而言，传统的网络安全设备仍然是实现云数据中心网络安全的重要基础。现在市场上的主流网络安全产品分为以下三大类 ：基于包过滤策略的基础防火墙类、入侵检测（IDS）和入侵防御（IPS）类，以及针对特殊协议的主动安全类，如针对 HTTP 协议的 Web 应用防火墙 WAF 和专门负责数据库 SQL 查询类的数据库应用防火墙 DAF，关于 WAF 本部分第 26 章 “ SaaS 服务安全 ” 一章里将进行介绍。

2. 云计算环境下网络安全新变化

在云计算环境下，虚拟化的计算和存储资源以及其他云应用最终都是通过网络向用户提供。因此，云数据中心网络上通常需要承载各种不同的复杂应用和服务，而不同的应用和服务通常又部署于虚拟化服务器之上，如何在有限的网络资源范围内安全、有效、可扩展地承载多样化的虚拟化应用是云计算对数据中心网络架构提出新的要求 ；另一方面，如何对这些应用的不同访问权限的用户进行控制、不同业务间的网络传输进行隔离，让用户尽可能安全透明地使用网络，也是云计算下对云数据中心网络架构提出另一个新的要求。

网络虚拟化是近些年来各大标准组织和网络设备厂商应对上述要求而提出的新的解决方案。采用这种技术的网络允许不同需求的用户组访问同一个物理网络，但从逻辑上是隔离的，并可保

持网络较高的安全性、可扩展性、可管理性及可用性，满足各种服务器虚拟化后对基础网络架构适配性的要求，并能节省电源，释放机柜空间等。

网络虚拟化分为纵向网络分割和横向网络整合两种场景。纵向网络分割即 1:N 的网络虚拟化，如采用 VLAN、MPLS VPN、Multi-VRF 等技术进行的网络虚拟化，主要用于隔离用户流量、提高安全性，以及用户通过自定义的控制策略实现个性化的控制，便于增值业务出租 ；横向网络虚拟化整合即 N:1 的网络虚拟化。在云计算环境下，多个网络节点承载上层应用，基于冗余的网络设计给网络带来复杂性。而通过路由器集群技术和交换机堆叠技术将多台物理网络设备整合成一台虚拟的网络设备，不仅可实现跨设备链路聚合，简化网络拓扑，便于管理维护和配置，消除“ 网络环路 ”，还可增强网络可靠性、提高链路利用率。

无论是纵向分割还是横向整合的网络虚拟化技术，在满足云计算环境下对网络架构适应性要求的同时，一方面也提升了网络服务的安全性，但同时也给独立部署于这些虚拟化网络设备之外的传统网络安全设备提出了新的挑战。因为这些网络安全设备对发生在虚拟网络设备内部的流量是无法进行监控与审计的，同样对于那些部署于虚拟机之上的各类应用服务器之间的流量也是无法进行监控与审计，任何一台虚拟交换机或虚拟服务受到攻击，极易扩展到其它虚拟交换机或虚拟服务器上。因此对于基于虚拟化技术部署的云计算环境，必须在传统的网络安全防护措施基础上叠加针对对虚拟化流量的安全监控技术实施，如虚拟化交换机和虚拟化防火墙。

23.2.3　主机安全

相对于小型机的封闭系统而言，PC 服务器在软、硬件架构上开放性更大，标准化也更高，这一方面推动了 PC 服务器的广泛应用，另一方面上也给黑客攻击创造了便利条件，使得服务器底层代码容易被修改从而被植入病毒或其他一些恶意程序 ；而系统漏洞的存在也给黑客窃取权限植入攻击程序造成了可乘之机。另外，无论是自建的 IT 服务还是云计算最难以防范的还是来自内部的越权访问。

可信计算技术就是针对上述情况从芯片、硬件结构和操作系统等方面综合采取措施来从根本上解决 IT 系统安全性的尝试。可信计算的目的是在计算和通信系统中广泛使用基于硬件安全模块支持下的可信计算平台，从根本上提高 IT 系统及网络设备的安全性。图 23-1 是带有可信平台模块的服务器体系架构示意图。

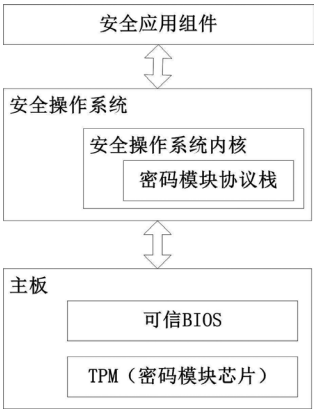


图23-1　可信计算平台架构

对于 SLA 要求较高的 CS，采用基于可信平台模块（TPM）来构建云计算数据中心的云资源或云应用服务器，可以实现 ：

- 用户唯一身份、权限、工作空间的完整性 / 可用性 ；
- 存储、处理、传输的机密性 / 完整性 ；
- 硬件环境配置、操作系统内核、服务及应用程序的完整性 ；
- 密钥操作和存储的安全 ；
- 系统具有免疫能力，从根本上阻止病毒和黑客等软件的攻击。

23.2.4　虚拟化安全

虚拟化技术是实现云计算的核心技术，是服务器端资源共享和应用服务传播的基础。同时也正是由于虚拟化技术的引入，使得在云计算环境下仅仅依靠传统基于主机的病毒防护等安全措施不再凑效。虚拟化技术可能引发的安全问题主要存在于以下三个方面 ：虚拟化软件、虚拟化服务器及虚拟化存储设备，下面分别阐述。

1. 虚拟化软件安全

实现主机虚拟化的方法不止一种，根据实现技术不同可以分为 ：基于 CPU 指令集的虚拟化、并行虚拟化（半虚拟化）、硬件虚拟化（全虚拟化）和操作系统的虚拟化。

对于规模化商用的云计算，多以基于 VWware vSphere 的全虚拟化模式实现，因此虚拟化软件通常都直接部署于裸机之上，它提供了创建、运行、销毁和管理虚拟服务器的能力，也是保证客户的虚拟机在多用户环境下相互隔离的重要一层，因此虚拟化层的完整性和可用性对于保证云计算的完整性和可用性是至关重要的。但很不幸的一个事实是，虚拟化软件如同其它任何一种软件一样，也存在可以为黑客所利用的漏洞。如在 2009 年 7 月下旬的美国黑帽大会上，一些研究机构即对虚拟机存在的漏洞给出了最为清楚的阐释。

但在目前情况下，由于绝大多数商用的公有云虚拟化层使用的是非开源的商用软件，因此安全厂家一般无法获得云计算服务提供商使用的软件源代码。因此虚拟化软件自身的安全技术实施主要依赖于虚拟化厂商的安全技术。

虚拟化主流厂家之一的 VMware 公司目前已发布虚拟化软件安全应用框架 VMware VMsafe，它不仅为虚拟部件（如 CPU、内存、网络和存储）提供安全框架，并能为与虚拟机和 hypervisor 交互作用的虚拟安全服务提供必要的 API，目前已与 VMware vSphere 4 版本进行了集成。

图 23-2 给出了基于集成的 VMsafe 的 vSphere 技术实现 CPU 和内存的安全保护机制。图中 VMM 扩展基于 VMM 对 HyperVisor 层进行的扩展，基于 VMsafe 构建的安全应用可以直接访问这些扩展模块，从而可实现对 CPU、内存、网卡和硬盘等资源的监视，并通过通用寄存器值来检测受保护的虚拟机 CPU 的当前使用状态，以及通过控制寄存器值来监视系统配置状态。

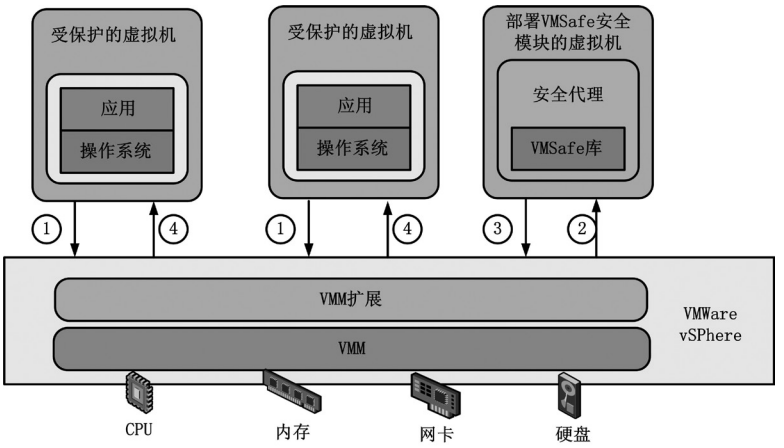


图23-2　基于集成VMsafe的VM vSphere技术对主机资源安全保护机制

基于集成 VMsafe 的 VM vSphere 技术对主机资源的安全保护机制简述如下 ：

- 虚拟机企图执行受限代码
- VMsafe 以事件方式通知 VBE security agent
- 经检测验证是合法请求
- 虚拟机执行该操作，得到结果

由于 VMsafe 虚拟程序具备直接访问处于虚拟化管理程序（hypervisor）中数据的能力，包括读写内存、存储和访问网络设备。在一些情况下，VMsafe 虚拟程序甚至可以改变从内存、存储设备或网络读取的数据。因此，如果部署 VMsafe 应用程序的虚拟设备处于危险的环境中，如隔离区（DMZ）或可以直接访问 Internet 的环境，那么这个虚拟环境就非常容易被攻击。基于此，在基于 VMware VMsafe 进行虚拟化软件安全部署时建议 ：

- 不要把 VMsafe 设备安装在隔离区（DMZ）；
- 不要赋予它们直接访问 Internet 的能力，设置成通过代理服务访问 ；
- 不要安装在虚拟机网络层 ；
- 不要安装在服务管理层和 IP 存储层 ；
- 不要安装在 VMware VMotion 或 Fault Tolerance Logging 网络层。

2. 虚拟化服务器安全

这里的虚拟化服务器包括采用虚拟化技术部署的虚拟主机及其上运行的操作系统。在公共 IDC 基础设施云计算中，虚拟化服务器可能面临的安全威胁包括 ：

- 窃取主机与虚拟主机间通信密钥（如 SSH 的私钥），用于访问和管理主机 ；
- 攻击未及时更新补丁存在漏洞的主机或虚拟机上的标准端口服务（如 FTP、NetBIOS、SSH）；
- 劫持弱口令或没有密码保护的虚拟机账户 ；
- 攻击安全策略设置不够严密的主机防火墙 ；
- 在虚拟机软件组件或在虚拟机镜像（操作系统）本身嵌入木马等。

对于虚拟化服务器安全的技术实施既要参考传统服务器的安全原理与实践，如系统漏洞修补

措施等，还要兼顾虚拟化技术引入后带来的安全技术实施的不同点，如系统病毒防护技术实施。下面将从虚拟化服务器的安全部署及日常管理两方面对虚拟化服务器的安全进行阐述。

（1）虚拟化服务器安全部署

消除单点故障：配置服务器集群，当其中的一台机器发生故障，要能够及时将这台机器上的应用动态迁移到别的机器上，VMware 和微软两大虚拟化平台提供商都提供了类似故障迁移的功能，从而保证应用服务不会被中断，提供了高可靠性。

安全防护：虚拟化服务器系统还应安装基于主机的防火墙、病毒防护软件、基于主机的 IPS（IDS）以及日志记录和恢复软件，以便将它们相互安全隔离，并与其他安全防范措施一起构成多层次防范体系。

目前已有部分技术领先的安全厂家针对虚拟化架构发布了相应的安全防护产品，如趋势科技公司发布的服务器深度安全防护系统 Deep Security 7.5，产品通过与虚拟化软件如 VMware vCenter 的 VMware 和 ESX 服务器的集成，能够将安全防护功能应用到 VMware 基础架构上；此外通过与 VMware VMsafe APIs 的紧密集成，使得它能在 ESX 服务器上作为一个虚拟应用快速部署，实现透明化地无代理保护 vSphere 虚拟机。

网络隔离：网络架构是服务器虚拟化的过程中变动最大的一环，也是最有可能产生安全问题的关键。服务器虚拟化之后，所有的虚拟机很可能就集中连接到同一台虚拟平台（如 VMware vSphere 或微软的 Hyper-V），与外部网络进行通讯。在这种架构之下，应通过 VLAN 和不同 IP 网段的方式进行逻辑隔离，对需要相互通信的虚拟服务器之间的网络连接应通过 VPN 方式来进行，以保护它们之间网络传输的安全。除此之外还要建立起一套行之有效的监控手段，以便了解各个虚拟机之间通信记录。

（2）虚拟化服务器安全运维管理

从运维管理的角度来看，对于虚拟服务器系统，应当像对一台物理服务器一样地对它进行系统安全加固和虚拟化软件脆弱性检查，包括虚拟机操作系统补丁、系统最少化服务开放、虚拟化软件补丁等。同时严格控制物理主机上运行虚拟服务的数量，禁止在物理主机上运行其他网络服务。如果虚拟服务器需要与主机进行连接或共享文件，应当使用 VPN 等安全连接方式，以防止由于某台虚拟服务器被攻破后影响物理主机。文件共享也应当使用加密的网络文件系统方式进行。此外，由于虚拟机的镜像文件包含了虚拟化软件、虚拟化服务器配置信息、运行状态、访问帐号等重要信息，建议对此实施相应的安全备份策略，并控制镜像文件的访问权限。

最后，还要对虚拟服务器的运行状态进行严密的监控，实时监控各虚拟机当中的系统日志和防火墙日志，以此来发现存在的安全隐患。对不需要运行的虚拟机应当立即关闭。此外，还需要定期对虚拟镜像文件的加密存储和完整性检查以及访问控制。

3. 虚拟化存储设备安全

随着大量有价值的应用数据集中于云端，这将极大地激发黑客或非法竞争者窃取云上存储数据或对云存储设备进行攻击的兴趣。一些入侵行为将会突破基于网络的入侵检测系统（NIDS）和基于主机的入侵检测系统（HIDS）的防护，进入到主机系统，进而对存储的数据进行攻击。

另一方面，研究[2]也发现，在存储层次上，有些入侵行为能很容易被检测到，这些可疑操作包括：植入后门或者密码、窜改日志、改变文件属性（隐藏修改）等。所以，在云计算环境下，

存储层将成为 IDS 攻防双方又一个兴趣所在。但值得欣慰的是，随着处理器速度的提高、内存容量的增长，也使得 IDS 技术可以嵌入运行在不同的网络存储设备中，这就使得虚拟化存储设备的安全防护有了新的手段，即基于存储的 IDS（SIDS）防护产品。

图 23-3 给出 NIDS、HIDS 和 SIDS 在虚拟化存储网络中的部署位置示意图。

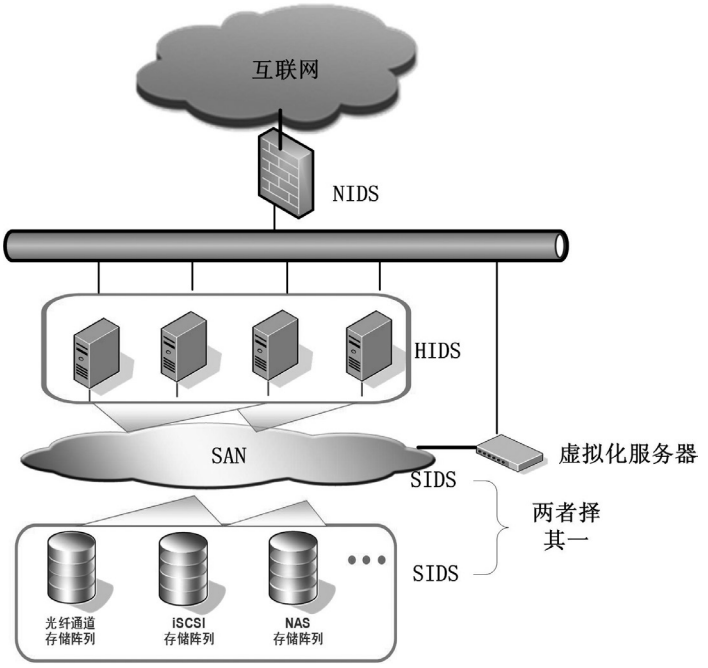


图23-3　不同入侵检测技术在虚拟化存储网络中的部署位置

SIDS 能对存储设备上的所有读写操作进行抓取、统计和分析，对可疑行为进行报警。由于基于存储的入侵检测系统是运行在存储系统之上，拥有独立的硬件和独立的操作系统，与主机独立，能够在主机被入侵后继续对存储介质上的信息提供保护。

23.2.5　中间件安全

1. 中间件安全概述

按照 PaaS 服务定义，PaaS 服务提供了使用平台支持的语言进行设计、开发、测试、部署和支持定制应用程序的集成环境。PaaS 服务中间件即指用于设计、开发、测试、部署和交付应用所需的所有基础服务软件的统称，如数据库服务、开发环境、运行环境和应用服务器等。对于大型 PaaS 服务提供商，这里的集成环境不仅包括服务中间件，还包括前面所述的基础设施，即是 PaaS+IaaS 服务的融合。基础设施安全已在 23.2.1 - 23.2.4 章节里进行了说明，下面主要讨论 PaaS 服务中间件安全。

在讨论 PaaS 服务之前，先来简单了解一下业界两个 PaaS 服务的典范，一个是 Windows Azure，它是微软推出的云计算版本操作系统，提供微软各种软件的网络版本应用，目前包括 SQL、.NET、Live、SharePoint、Dynamics CRM 等服务；另一个是 Google App Engine

（GAE），是谷歌基于分布式文件系统（GFS）、并行计算（MapReduce）、分布式数据库（BigTable）等技术提供的应用开发服务，Google 基于其服务已开发了 Google Docs、Google Email、Google Search 和 Google Earth 等大量应用。

但就是这两个业界典范的 PaaS 服务，相继都发生了服务故障 ：

2009 年 3 月，Windows Azure 服务出现故障。微软对外宣称 Azure 服务中断是由于操作系统升级时，由于网络问题，Windows Azure 的部署服务减缓，导致大量服务器出现超时和中断。因为升级过程中一旦这些服务器掉线，监控系统就会通知微软工作人员，与此同时，光纤控制器自动启动，将受影响应用程序移至其他服务器，由于光纤控制器会采取全面的修复步骤，因此，这一系列措施的执行需要很长时间。

2011 年 9 月，Google 文档（Google Docs）办公套装遭遇服务中断故障，免费个人用户和企业用户均受到此次故障的影响。故障期间当用户使用这项服务时，约 10 分钟内无任何反应，此后才会出现错误页面。后 Google 对外解释是由于内存管理的 BUG 导致。因为每次 Google Docs 服务器需要更新的时候，都有一台机器去寻找那些需要升级的服务器。但由于内存管理的 BUG，这台查询的机器在查询完毕后没有正确地清空它们的内存，于是导致这些服务器最终耗尽内存，不得不重启。然而在重新之后，它们再次被查询的机器捕获，使得它们更快地再次耗尽内存，结果这些服务器在周三就无法正常的处理文档列表、文档、绘图和脚本。

当然，由于 Google PaaS 服务与 SaaS 应用的紧密结合性，难以了解到这次看似是 Google SaaS 服务的故障到底与底层的 PaaS 服务多大关系，但无论如何这些服务的中断让我们认识到无论是 PaaS 服务还是基于 PaaS 服务开发的应用在安全方面都还有许多需要关注的地方。

虽然 PaaS 服务可提供业务快速生成、测试、部署及运行的环境，但相对 IaaS 和 SaaS 服务而言，仍是一种新兴的服务模式。从总体上讲，商用模式尚未形成，实现技术也尚不成熟，在实现技术上基本上是各家自成体系。此外，由于 PaaS 服务的安全技术实施往往是与平台的体系架构设计密切结合（这也是 PaaS 服务与 IaaS 服务安全技术实施最大的区别，IaaS 服务的安全技术实施大多可单独部署），因此，在讨论 PaaS 服务平台安全之前，最好先借助具体的 PaaS 服务平台模型来分析。图 23-4 借助 Google 的云计算平台模型为例说明 PaaS 服务组件一般都可能包含哪些服务元素 ：

- 文件管理服务 ：Google 为云计算提供海量数据存储自行开发了一套分布式式文件管理系统 GFS ；
- 数据库服务 ：Google 针对其高达 PB 级的数据存储开发了并行结构化数据库管理服务 BigTable ；
- 应用编程模型（开发框架）：Google 针对其大量需要处理的数据开发了并行数据处理模型 MapReduce，同时，为了解决并行处理过程中的冲突，并开发了分布式锁机制 Chubby ；
- 应用开发工具 ：Google 除了使用传统的 C++ 和 Java，还开始大量使用 Python。

基于这些 PaaS 服务基础软件之上，是 Google 自行开发的大量 SaaS 应用，如 Google Earth、Google Email 和 Google Search 等。此外，PaaS 服务平台组件通常还为用户提供应用程序的运行环境（如 JAVA 虚拟机）和应用服务器（如 WebSphere、JBOSS、WebLogic 等）。

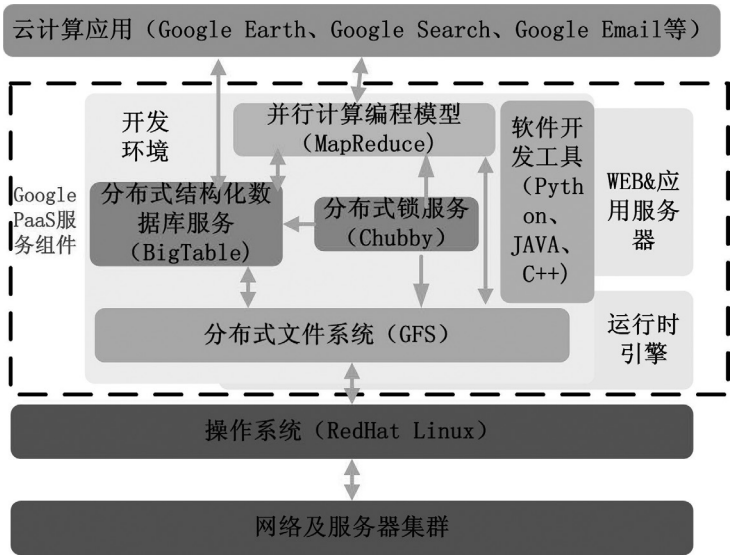


图23-4 Google PaaS服务平台模型

一般来说，PaaS 服务提供商（例如 Google、Microsoft 和 Force.com）负责保障平台整个软件栈的安全，包括运行用户应用程序的运行时引擎及应用服务器等组件的安全。

从本质上讲，PaaS 服务也是一种资源共享的服务模式，因此在多租户模式下安全和隔离是其主要考虑内容，包括用户隔离、用户认证、权限管理、用户安全接入及服务组件补丁管理等。下面主要对 Google PaaS 服务平台中的安全技术实施进行简要分析。

2. 中间件安全关键技术

（1）安全沙盒技术

在多租户 PaaS 服务交付模式中，PaaS 服务中间件安全设计的核心原则是多租户应用程序间的控制和隔离。在这个模式下，对租户的数据访问应当限定在租户自身所拥有和管理的应用程序内。

沙盒技术很好地实现了将应用程序在隔离于其它租户的安全环境中运行，该环境与网络服务器的硬件、操作系统和物理位置无关，并确保用户仅能执行不影响其他应用程序的性能和可伸缩性的操作。另一方面，由于安全沙盒环境仅提供对基础操作系统的有限访问权限，这些限制让应用引擎可以在多个服务器之间分发应用程序的网络请求，并通过启动和停止服务器来满足流量控制需求。

以下是安全沙盒环境的隔离或限制作用几个示例 ：

- 应用程序只能通过提供的网址抓取以及电子邮件服务访问互联网中的其他计算机。其他计算机只能通过在标准端口上的 HTTP（或 HTTPS）请求来连接至该应用程序。
- 应用程序无法写入文件系统。应用程序只能读取通过应用程序代码上传的文件。该应用程序必须使用应用引擎数据存储区、缓存或其他服务存储所有在请求之间持续存在的数据。

· 应用程序代码仅在响应网络请求或调度任务时运行，且任何情况下必须在 30 秒钟内返回响应数据。请求处理程序不能在响应发送后生成子进程或执行代码。

提供商维护 PaaS 平台运行引擎的安全，在多用户模式下必须提供“沙盒”架构。PaaS 平台沙箱体系在维护部署于 PaaS 中的应用程序的保密性和完整性方面起到了重要作用，PaaS 服务提供商有责任检测那些运行在 PaaS 平台并可能打破沙箱体系架构的用户程序中的错误和漏洞。

（2）用户认证

用户认证是确保合法的访问主体（用户、进程、服务等）能正确地被识别的过程。双因子认证（智能卡和生物特征）机制在安全性上虽然较强，但复杂性太高、实施周期也长、代价也大，目前在 PaaS 服务中采用的不多。大多 PaaS 服务提供商仍采用用户名和密码这种简单而古老的方式进行认证，当然你也可以通过一些技术来增强这种认证过程的安全性，如通过回答一个安全问题或识别一个预置的图片等。当然 PaaS 服务还可以使用一些基于外部组织声明为安全的标识来做认证，如 SAML（安全断言标记语言）等。

基于 Google 应用引擎开发的应用程序使用 Google 帐户进行用户认证。应用程序可使用 Google 帐户登录、获取当前登录用户的电子邮件地址，并生成与帐号相关的登录和退出网址。应用程序根据 Google 帐户指定网址路径及部署描述符文件预先定义的安全约束限制用户的访问行为。如果用户访问了有安全约束的网址路径但用户没有登录，应用引擎会自动将用户重定向到 Google 帐户登录页面。在成功登录或注册新的帐户后 Google 帐户会将用户重定向回应用程序网址。应用程序无需做任何其他事即可确保只有登录用户可以访问网址。

（3）安全网址

Google 对于使用 *.appspot.com 域网址访问的用户应用，Google 应用引擎支持通过 HTTPS 的安全连接。当请求使用 HTTPS 访问网址时，请求数据和响应数据都在传输前由发送器加密，然后在接收后由接收器解密。使用安全连接有利于保护用户数据，如联系人信息、密码和私人消息。

应用程序使用安全网址之前，用户必须使用 <ssl-enabeld>true</ssl-enabled> 元素在应用程序的 appengine-web.xml 文件中启用安全网址。有关此文件的详细信息及安全网址配置说明，请参阅 Google 相关文档说明 [3]。

（4）访问控制

访问控制是针对越权使用资源的防御措施。基本目标是限制访问主体（用户、进程、服务等）对访问客体（文件、数据、服务等）的访问权限，从而使客体资源被合法使用。目前权限控制主要是基于角色访问控制框架实现，即帐号赋给角色，角色与应用访问或资源操作权限相关联。

PaaS 服务基于资源共享的理念设计，因此，PaaS 服务提供商尽可能不要让用户访问到他们更底层的基础设施或其他不相关的资源和服务，尽管这种做法可能使得付出的成本更大，但这是 PaaS 服务与企业应用最大的区别，因为我们无法预知哪个用户可能会带有恶意攻击行为，因此，PaaS 服务中权限访问控制最好的建议是遵循“最小化”权限赋予原则。

Google 对基于 Java 应用引擎开发的应用程序使用 JRE 类白名单方式控制应用程序的访问权

限，使得应用程序对 Java 标准库（Java 运行时环境或 JRE）中类的访问权限仅限于 App Engine JRE 白名单中的类。

除上述安全技术实施外，Google 对其 PaaS 服务平台的服务组件还要进行相应安全管理工作，如对运行时引擎的升级，以及变更、发布和补丁管理等，并需要对 PaaS 平台外部的网络和主机进行安全监控，例如对于共享网络和运行用户应用程序的系统基础设施的监控。

最后需要强调一点的是，PaaS 服务中间件的安全还必须有 PaaS 用户的积极参与配合。CP 应尽可能地让用户了解自己的安全体系架构设计，以便他们在应用实践中能更好地遵守安全约定，充分利用体系架构中的安全设计功能合理地进行应用安全配置。同时还需要让用户明确了解一些安全相关的法律问题，如数据不允许随意散布等。

23.3　本章总结

本章从云服务供应商 CP 的角度说明了云计算基础设施层面的安全技术实施及应对措施。

作为 CP，可以提供在云计算设施多个层面提供完整而有效的安全保护机制。在服务设施层面的安全保护机制主要包括物理环境、网络环境、物理主机、虚拟化环境以及 PaaS 服务中间件等。前四个层面主要由 IaaS 服务提供商负责实施 ；PaaS 服务提供商则要负责 PaaS 服务中间件部分的安全技术实施。但对于大型 PaaS 服务提供商，IaaS 部分往往是其 PaaS 服务基础设施的一部分，因此，这种 PaaS 提供商将负责从物理环境到 PaaS 服务中间件整个层面的安全机制。这部分还重点说明了云计算下由于虚拟化技术的引入对网络安全和主机安全带来的不同点。

参考文献：

[1] NCS-TG-004-88，Glossary of Computer Security Terms，NCS

[2] Joel Scambray，Stuart McClure, George Kurtz, Hacking Exposed: Network Security Secrets and Solutions, McGraw Hill，2001

[3] Phil Cox，Top Threats in a PaaS Cloud Service and How to Avoid Them，http://searchcloudcomputing.techtarget.com/tip/Top-threats-in-a-PaaS-cloud-service-and-how-to-avoid-them

第24章　云计算环境下的数据安全

24.1　云计算环境下数据安全综述

24.1.1　数据安全保护的意义

云计算模式的引入，使得数据安全成为人们关注的焦点。这是因为相对于传统的 IT 服务或私有云计算模式，公有云、社区云或混合云（尤其是 SaaS 服务）中数据安全风险的最终承担者与安全职责主体并不完全一致，CS 承担了数据安全风险的绝大部分，但云中数据安全的技术实施主体往往却要依赖于 CP，CS 在这个层面通常只起到一个辅助配合的作用。因此，CP 在数据安全层面的作用是至关重点的，对于一个 SaaS 服务提供商而言，CP 的安全技术实施在很大程度上决定了用户数据是否安全。当然，一个 SaaS 服务提供商的信誉好坏与其提供数据服务的安全性也密切相关，相信没有用户会愿意把数据放在一个经常出现数据泄露事件的 CP 那里。除安全技术实施外，CP 还应该详细收集、监控、记录一些安全行为的数据，以在必要时提供给用户或相关组织作为事件审计追踪使用。

但无论 CP 怎么承诺，CS 必须要牢记保护云中有价值的数据安全的责任永远都是数据拥有者本身，而不是 CP。即使 CP 可能按服务合同采取了某些安全机制，但数据拥有者也就是 CS 自己一定要对其数据是否真正安全负起责任，包括评估、监督、检测 CP 的安全机制是否满足你的要求，并要清醒意识到用户侧的安全对于云侧的安全同样重要，任何用户侧的不安全行为将会使用云计算侧的安全保护实施失去意义。而一旦那些有价值的数据丢失和损毁，导致破产的不会是 CP 而是 CS 自己。

24.1.2　数据生命周期

云计算环境下的数据如同其任何事物一样，也具有完整的生命周期。在云计算环境下，数据从生成、传输、存储、使用到销毁构成了一个完整的生命周期过程。在此生命周期过程中，数据在每个阶段都应有相应的保护侧重点。图 24-1 给出了云计算环境下数据生命周期与每一阶段侧重点的数据保护机制对应关系图，图中的每行文字的左边部分代表数据所处生命周期阶段，右侧文字代表该阶段重点的数据保护机制。

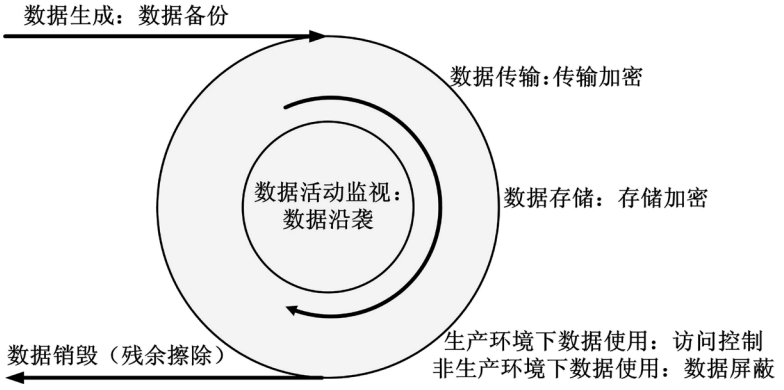


图 24-1　云环境下数据生命周期与数据保护机制对应图

从图中我们可以看出，除多租户环境下需要考虑数据隔离机制外，保护云中的数据并没有用到更多新的技术，身份认证和访问控制、传输加密、数据加密、数据屏蔽、安全删除（残余销毁）、数据沿袭、数据备份等用于传统数据中心的数据保护机制仍然可用于云计算环境下的数据保护。

本章后续章节将说明在云计算环境中用户和服务提供商面临的风险，并针对这些风险阐述在技术上可能实施的数据安全保护机制。

24.2　服务提供商面临的数据安全风险及挑战

24.2.1　数据加密

不够安全的加密数据或无法加密的数据是 CP 进行数据安全管理面临的最大风险。

数据加密的重要性 IT 专业人员可能很清楚，但放在云中，尤其是公有云环境下，不管是 IaaS、PaaS 还是 SaaS 服务，不是每一个用户都会很清楚意识到它的重要性。云计算环境下数据最大的风险来自于数据不加密或加密算法不够强。

即便进行了数据加密，但数据通过互联网传输时，如没有使用安全传输协议（直接使用 FTP 或 HTTP 协议），虽然可以保证数据的机密性，但在某些情况下并不能保证数据的完整性，如在对称加密情况下。

更为重要的是，即便是数据在传输时也选择了加密传输，但数据通过互联网传输时还需要用户和服务提供商双方的共同配合。因为采用何种安全传输协议（FTPS、HTTPS 等等）是 CP 可以控制的事情，但安全帐号管理和安全协议具体配置却是 CS 自己掌控或选择的事情，一个弱口令（或管理不严的证书密钥）或重复使用的口令甚至不足够安全的加密算法都可能使这种努力付诸东流。

抛开用户的安全意识或安全管理等问题，即使 CS 和 CP 都愿意使用可能获得的最安全的协议来保护其应用数据的安全，但在具体的实际应用中实施起来也困难重重。如果你仅仅使用的是 IaaS 服务（如亚马逊的 S3），加密数据很容易做到，但是对于 PaaS（如 Google APIs）或 SaaS（Salesforce.com 的 CRM）中的这些基于应用的数据进行加密实施起来却不大容易，这是因为对于这类数据（姑且称其为处理中的数据）加密导致大多数数据分析方法（如数据索引或查询等）失效，即便可以实现，性能或效率也是一个很大的问题。因此，对于云中的数据，除了极少部分仅仅用作静态存储的数据外，大量基于应用的数据往往都是没有加密的数据。

对于上述这些基于应用没有加密的数据，它的风险很大程度上取决于 CP 或 CS 的应用设计是否足够安全。但很多时候云计算的经济性是以共享基础设施为代价的，这也意味着在 PaaS 或 SaaS 服务中应用的数据通常是共存于一个大型数据仓库中（如 Google 的 BigTable）。在应用的设计上虽然考虑多租户而采用了数据标签来标识数据的用户属性来防止非授权的用户访问，但由于应用或系统总存在一定的漏洞或脆弱性（如 2009 年 Google Docs 的一个漏洞使得用户文档和表单出现非授权共享），使得一些攻击者仍然可以通过这些漏洞访问到这些数据。

24.2.2　钓鱼行为

云中数据的高度集中性和可获利性使得攻击者对其垂涎三尺。目前绝大多数云计算仍使用基

于简单的用户名和密码的认证方式，对于一个防范意识不强的用户而言，钓鱼攻击者很容易获取用户的认证信息，如果一个钓鱼攻击者成功获取了用户的认证信息，那么他可以轻而易举地访问到用户的其他数据 ；另一方面，即便云计算使用了公钥设施下（PKI）认证体系，如果云端的用户没有能将其认证信息严密管理好，攻击者仍然可以利用其从用户那里获取的认证密钥等信息来访问用户数据。对于安全防护技术实施不够严密的 CP 而言，此种访问行为通常是难以检测的。

对于这种行为，CP 只能增强自身的数据访问的认证和访问控制措施，但很多时候，CP 提供的数据访问增强保护机制的意义却往往依赖于用户的安全设置与操作。最好的防护措施是教育员工或用户增强对信息泄密及钓鱼行为的防范意识和识别可能的帐户登录欺骗行为的能力。下面是一些 CP 增强型的数据访问保护机制的例子 ：

Salesforce.com 登录过滤服务 ：Salesforce.com 提供一种服务用于限制对某些 CRM 应用实例的访问，用户可以对这种服务进行设置，这样即使访问者提供了合法的认证信息，如果登录者不是来自某个事先用户设置好的白名单 IP 地址，也不允许其登录应用。这种防护限制对制止钓鱼行为非常有效。

Google 的登录会话重验服务 ：Google 在 Apps/Docs 等云计算里提供了该服务，使用此服务的系统随机性地要求用户在使用服务期间重新输入密码，尤其当系统观测到某一可疑事件时，如同一帐号刚从美国登录又从中国登录，服务还会将这一事件自动通知给用户。

亚马逊 Web 服务认证 ：亚马逊的云资源服务认证很严格。当用户申请使用一个基于虚机的 EC2 时，亚马逊将默认为其创建强 PKI 密钥并要求使用这些密钥作为资源访问认证信息。如果你申请一个新 LINUX 虚拟机，并想通过 SSH 远程连接它，你就必须用基于密钥认证的 SSH 而不是一个静态口令。但这种方法也不总是能防范钓鱼攻击行为。

24.2.3　数据审计与监控

云中的数据来源、数据完整性、数据活动、数据的真实性等不仅是用户关注的主要问题，也是 CP 关注的主要问题。云中的数据来源在很多场合很有用，如在申请专利或用于证明有价值的数据拥有者身份等 ；而同样证明某些敏感数据不在云上对于 CP 一样重要，尤其在涉及跨国界的敏感数据隐私保护问题时。而数据在云中的处理过程对 CP 具有更重要的意义，因为无论是从合规性要求还是监管部门的审计要求，CP 都必须清楚知道什么数据何时放到云上，经过了什么处理，处理完后数据又放到了哪里等。但在云计算环境下，由于对物理资源的高度抽象、虚机的动态性以及多租户下用户数据的庞杂性，使得 CP 要对云中数据做全生命周期的详尽跟踪记录是一件极具挑战性的事情，它需要耗费大量的资源和时间，尤其对于中小型服务提供商，这更是一笔巨大的开销。

最后强调一下，上述是从服务提供商角度分析给出的数据安全风险。如从用户角度来讲，数据安全风险则与之完全不同，市场分析公司 Gartner 已在其研究报告中指出用户在使用云计算时存在七大数据安全风险 ：

（1）特权访问风险
（2）管理权限风险
（3）数据处所风险

（4）数据隔离风险
（5）数据恢复风险
（6）调查支持风险
（7）持续服务风险

24.3　数据安全保护机制

24.3.1　数据加密

1. 存储加密

云计算资源由多个“租户”共享，且 CP 对于云环境中的数据拥有特殊访问权限，因此，对存放于云中的敏感数据进行加密很重要，这样可以防止恶意的 CP、恶意的邻居“租户”的滥用及黑客攻击者的数据盗取。

目前，对于用户存放在云中的敏感数据进行安全保护通过采用加密、访问控制、隐私保护等措施。其中，加密是最常见也是最基础的静态数据安全防护手段，它是数据保护的最后一道防线。对存储的静态数据加密能实现数据的机密性、完整性和可用性，防止数据在存储介质意外丢失或者不可控的情况下信息的泄密，并减少对 CP 的安全实施的依赖性。

在 IaaS 服务中，用户可以使用 CP 或第三方工具很方便地实现对长期存储在云中静态数据的加密保护，这也是敏感数据放到云中之前用户必须做的；在 PaaS 服务中加密静态数据相对较复杂，需要使用 CP 提供的或支持的专用设备。在 SaaS 服务中加密静止数据难度更大，通常情况下云用户无法直接控制，通常要依赖于应用提供商的应用加密机制。而对于 PaaS 或者 SaaS 应用中需要动态处理的数据，是无法加密的，因为加密过的数据会妨碍索引和搜索，到目前为止还没有可商用的算法可实现对应用数据的完全加密。

上面简单分析了云中数据加密的重要性及不同服务环境下如何实施，下面着重对 IaaS 服务环境下的加密方法、加密实施方案进行说明。

（1）加密方法

有很多静态数据加密的方法，下面是几种可以用来保护云中静态数据的加密方法：

- 磁盘级数据加密：这是一种强力（brute-force）加密方法，通过对磁盘加密，将磁盘上的所有数据一同加密，包括操作系统、应用以及应用的数据。这种加密方法也会带来性能和可靠性的问题。如果加密不是在硬件驱动层进行的，那么这种方法将会耗费系统的性能。另外一个问题就是即使轻微的磁盘损坏，对操作系统、应用及数据将带来致命性的破坏。
- 目录级（文件系统级）加密：这种加密方法是将整个数据目录作为一个容器来加密和解密。访问文件则要求使用加密密钥。这种方法也可以用作对相同敏感性或同级别的数据进行整合，然后对不同的目录用不同的加密密钥加密。
- 文件级加密：仅对特定文件加密，加密效率更高。通常由第三方加密软件进行。
- 应用级加密：对应用管理的数据进行加解密，通常由应用系统本身带有的功能实现。

（2）加密方式

对存放于云中的数据进行保护有三种方式：

- 主机端加密：利用应用级加密方法由用户侧应用系统先对数据进行加密，然后再传输到云中。由于在进入云之前数据已经加密，这种加密方式相对安全性高。但由于不同用户端应用采用的加密算法的多样性导致加密强度的不一致，不利于云中数据存储安全的统一防护。为解决这个问题，IEEE 安全数据存储协会提出了 P1619 安全标准体系，这个体系制定了对存储介质上的数据进行加密的通用标准，采用这种标准格式可使得各厂家生产的存储设备具有很好的兼容性。
- 云中专用硬件加密：对存放于云中的数据进行安全保护的另一种方式是在云存储设备之前串接一个硬件加密装置，对所有流入存储网络的数据进行加密后，将密文提交给存储设备；对所有流出存储设备的数据进行解密后将明文提交给服务器。这种加密方式与上面提到的采用 P1619 的解决方案类似，但这里的加密是由外部加密装置完成，而不是集成在存储网络中。这种方式与上层应用和存储无关，但在数据量大的情况下，对硬件加密装置的加解密性能和处理能力要求较高。此外，这种加密方式还需要与传输加密配合使用才能有效保护用户数据在流经云中的安全性。
- 存储设备自加密：对数据加密保护的第三种方式是依靠存储设备自身的加密功能，如基于磁带机的数据加密技术，通过在磁带机上对数据进行加密，使数据得到保护。目前可信计算机组织（TCG，Trusted Computing Group）也已提出了针对磁盘的自加密标准，自加密磁盘提供用户认证密钥，由认证密钥保护加密密钥，通过加密密钥保护磁盘数据。认证密钥是用户访问存储磁盘上数据的惟一凭证，只有通过认证后才能解锁磁盘并解密加密密钥，最终访问存储磁盘上的数据。

2. 传输加密

在公有云计算这种开放的共享环境下，对网络传输中的敏感数据比如关键业务数据、帐户信息数据进行保护是极其必要的。保护传输中的数据最常用方法也是使用加密，即通过使用与认证技术相结合的加密来构建一个安全通道，来确保数据在云和用户侧的安全传输。

对传输中的数据加密有两个最主要的目的：一是防止数据被篡改（完整性），第二个是确保数据在传输过程中即使被窃取也不能获取信息，即除了收发双方，任何第三方都不可对它进行修改，或即使获取数据也难以解析数据内容。此外，通过与签名技术结合，传输加密还可确保发生在通信双方的任何操作都具有不可否认性。

（1）传输加密方法

对传输中的数据进行加密有以下几种方法：

①在非安全的网络中传输加密数据

在传输之前加密数据然后传输，如多用途网际邮件扩充协议（Secure Multipurpose Internet Mail Extensions. RFC 2311，S/MIME），但这种机制不能保证数据的完整性。

②传输的数据由底层协议加密（SSL/IPSec）

此种方式不必改变应用层协议，也不必改变传输层协议，它是在应用层与传输层之间加一层安全加密协议，达到安全传输的目的。

1995 年，Netscape 公司在其浏览器 Netscape1.1 中加入了安全套接层协议（Secure Socket Layer），以保护浏览器和 Web 服务器之间重要数据的传输。SSL 很好地封装了应用层数据，做到了数据加密与应用层协议的无关性，各种应用层协议都可以通过 SSL 获得安全特性。

由于 SSL 用较小的成本就可以获得安全加密保障，因此，它很快就广泛地应用于 Web 领域。

而 IPSEC（Internet 协议安全性）是通过对 IP 协议（互联网协议）的分组进行加密和认证来保护 IP 协议的网络传输协议族（一些相互关联的协议的集合）[1]。IPSEC 引入了完整的安全机制，包括加密、认证和数据防篡改功能 ；并通过包封装技术，能够利用 Internet 可路由的地址，封装内部网络的 IP 地址，实现异地网络的互通。

需要强调的是，虽然 SSL 3.0 通过数字签名和数字证书可实现浏览器和 Web 服务器双方的身份验证，但不能支持多方身份认证，而且只能保证数据在传输过程中的安全，对数据在到达本地的安全没有规定。所以，SSL 在在线交易领域的应用受到了一些限制。

③安全电子交易（SET）

SET（Secure Electronic Transaction，安全电子交易）是一种基于消息流的协议，其核心技术主要有公开密钥加密、电子数字签名、电子信封、电子安全证书等。SET 主要是为了解决用户、商家和银行之间通过信用卡支付的交易而设计的，以保证支付信息的机密、支付过程的完整、商户及持卡人的合法身份以及可操作性。由于它得到了 IBM、HP、Microsoft 、Netscape、VeriFone、GTE、VeriSign 等很多大公司的支持，它已成为事实上的工业标准，目前它已获得 IETF 标准机构的认可。

但是 SET 系统本身十分庞大且复杂，牵涉面广，已非 IT 行业本身所能完成，并且其实施费用较高，在一次交易中需要十几次的加解密操作，较为繁忙的 Server 通常需要加密硬件的辅助。

在上述三类传输方式中，SSL/IPSec 因其在部署实施简便性和安全性间的很好的折衷，使其已成为众多 CP 的一致选择。IPSec 由于其较高的安全性将成为 CP 在跨数据中心间数据传输时的最佳选择。而在 CP 与用户间的 Web 服务传递时，SSL 协议由于内嵌于 HTTP 协议而无需部署客户端，相比 IPSec 而言获得更广泛的应用。下面对 SSL/TLS 协议做简要介绍 ：

传输层安全（Transport Layer Security，TLS）是 SSL 的升级版协议，最终将取代 SSL 协议，同时保留 SSL 的向后兼容实现。1999 年 1 月，TLS1.0 第一次定义在 RFC 2246 中定义为 SSL 3.0 的升级版本。TLS1.0 也被称为 SSL3.1。

SSL/TLS 协议为实现客户机 / 服务器间通过网络进行安全通信而设计，以防止数据窃听和篡改。TLS 使用互联网加密来提供端点身份验证通信的保密性。SSL/TLS 协议可分为两层 ：SSL 记录协议（SSL Record Protocol），它建立在可靠的传输协议（如 TCP）之上，为高层协议提供数据封装、压缩、加密等基本功能的支持 ；SSL 握手协议（SSL Handshake Protocol），它建立在 SSL 记录协议之上，用于在实际的数据传输开始前，通讯双方进行身份认证、协商加密算法、交换加密密钥等。

通过 SSL 客户端与服务器间数字证书的传递，互验身份的合法性，特别是验证服务器身份的合法性，在云计算环境下更具有意义，这可以有效地防止网络环境下虚拟网站的钓鱼事件。

（2）密钥管理方法

安全算法和安全协议在理论上解决了基础设施层面的安全问题，但在大多数时候，安全专业人员每天面临的是如何安全地管理密钥和证书，硬件安全模块（HSM）、智能卡、加密狗等都是常用的解决方案。在服务器端，它们是密钥服务器，其中硬件安全模块（HSM）是一种硬件密钥服务器。在客户端，它们是智能卡和加密狗。

硬件安全模块（HSM）是带有严格访问控制和物理安全保护机制的硬件密钥服务器。使用 HSM 的目的是 ：

- 生成安全密钥 ；
- 安全存储密钥，它可以避免任何 HSM 之外的明文密钥处理 ；
- 控制对加密和敏感数据资料的使用 ；
- 彻底卸载应用服务器上处理非对称和对称加密的负荷。

HSM 对上述内容提供了从非授权使用到潜在恶意攻击的逻辑和物理上的保护，通常用于保护高价值的加密密钥。

许多硬件安全模块系统自带安全备份或者复制密钥的功能，它们要么通过计算机操作系统打包工具进行处理，要么是使用一种外部智能卡或其他一些安全令牌环进行处理。硬件安全模块决不允许机密信息以明文形式暴露，即使是在两个硬件安全模块之间的迁移或执行备份操作等也不允许。

硬件安全模块主要用于金融机构、发卡机构或政府机构。这些机构需要安全相关部署，例如 CA 签名机制、EMV 数据准备和卡个性化、PIN 认证、数据加密和数据签名等。

为了保护和简化客户端的密钥 / 证书管理，智能卡或者加密狗提供了一种高成本效益的解决方案。供应商通常将软件保护加密狗（和加密狗控制的软件）称作 “ 硬件钥匙 ”、“ 硬件令牌 ”、“ 保安装置 ”，而不是 “ 加密狗 ”，不过加密狗的叫法在日常生活中更常见。加密狗大多用于大批量交易或高敏感数据中。有些加密狗厂商在加密狗产品中采用智能卡产品，这些智能卡产品广泛使用于军事和银行等对安全要求极其严格的环境中。

通常，在一些专门设计的软件中，智能卡和加密狗是一起使用的，这给终端用户带来很多问题。所以要确保使用智能卡 / 加密狗级安全保护的 SaaS 应用成功推广，还需要保证客户端软件必须要设计成完全傻瓜型（dummy proof）且要足够健壮的。

最后需要指出的是，在云计算环境下，CP 不仅需要关注发生在用户和云计算数据中心间的数据传输安全，还要关注发生在跨多个云计算数据中心之间的数据传输以及云计算中心内部不同系统间（服务器、储存设备与网通设备）的数据交换的安全。相比较而言，发生在单一云计算数据中心内的跨系统间的数据交换的安全，更应该成为 CP 关注的重点，因为一方面云计算环境下存在基础设施技术体系的差异性以及云应用数据的多样性，而另一方面产业层面却缺乏统一的数据交换标准，这样就很难向用户保证数据在整个云环境下的传输安全性。

3. 实践建议

前面讨论了静态和传输中的数据加密，其核心是加密 / 解密算法和密钥 / 证书管理。在云计算中数据加密 / 解密常用的算法有 RSA（公共密钥加密协议）、ASE（对称算法，高级加密密钥）、SHA-1（希哈函数）等。关于数据加解密及加密算法等方面的相关知识，网上有大量文章可以参考，本书不做详细介绍。下面仅给出数据 / 传输加密 / 解密和密钥管理过程中的一些最佳实践建议。

（1）数据加解密实践建议

下面是针对 CP/CS 给出一些云计算环境下使用数据加密的建议 ：

- 如在合同中约定加密时，需要确保加密遵循了相关行业和政府标准。
- 尽可能地使用加密，并尽可能把数据使用者与数据保管者分离。
- 除了确保敏感数据在存储时是加密的，还要确保其在云提供商的内部网络传输时也是加密的。在 IaaS 环境中，这将由云用户选择实施 ；在 PaaS 环境中，由用户和提供商共同分担责任 ；在 SaaS 环境中，由云提供商来负责。
- 在 IaaS 环境中，要充分考虑传统加密保护的敏感信息和关键材料可能出现的各种泄密的情形。如虚拟机交换文件与其他临时数据存储位置可能也需要进行加密。
- 尽可能把存放数据的组织（如 CP）与密钥管理者（如第三方）分离，这样即保护了云提供商，也保护了用户，避免由于法律要求提供数据时产生冲突。
- 如果 CP 必须进行密钥管理，了解提供商是否定义了密钥管理生命周期的过程 ：密钥如何产生、使用、存储、备份、恢复、轮换和删除。而且，了解是否每个客户使用了相同密钥或是否每个客户有其自己的密钥系列。
- 了解云提供商的设施是否提供了角色管理及职责分离。

（2）密钥管理实践建议

不管采用何种加密方式，数据加密最关键一个因素是对加解密密钥的选择，不同加密强度（加密强度通常指加密时所用密钥数据位数）适合不同的数据使用场景，数据加密需要选择合适的加密强度才能起到安全有效地保护存储的数据。强加密适用于长期数值不变的静态数据，因为如果这种数据被第三方获取，他们需要足够长的时间才能破解密码以获取重要信息 ；弱加密适合于数值经常变化或敏感性不高的静态数据。

加密信息的安全可靠依赖于密钥系统，密钥是控制加密算法和解密算法的关键信息，它的产生、传输、存储等过程的管理是十分重要的。在云计算环境下，不同数据分级、不同用户需求要求不同的加密算法和不同的加密强度，CP 需要提供一套严密的加密密钥管理方案来保护云中数据加密密钥的安全，或者他们将这些工作都交由用户自己管理。下面是进行密钥管理需要重点关注的一些内容 ：

- 密钥存储 ：密钥必须像其他敏感数据一样进行保护，对其存储、传输和备份过程都必须进行保护，并建立相关策略来管理密钥存储，不适当的密钥存储可能危害所有加密数据，如将密钥存储于公网访问的系统上等 ；
- 密钥使用 ：必须限制只有特定的需要单独密钥的实体可以访问密钥，使用基于角色的策略控制密钥访问，给定密钥的使用实体不能是保管该密钥的实体 ；
- 密钥备份和恢复 ：丢失密钥无疑意味着丢失了这些密钥所保护的数据。尽管这是一种销

毁数据的有效过程，但是意外丢失关键任务数据的加密密钥会毁掉一个关键业务，所以必须制定密钥安全备份和恢复方案。考虑如果密钥丢失或不可用，要知道可以用什么替代方法来恢复数据。

有很多标准和指导方针适用于云中的密钥管理。OASIS 密钥管理协同协议（KMIP）就是云中协同密钥管理的新标准。IEEE1619.3 标准涵盖了加密和密钥管理，尤其适用于存储 IaaS。

此外，对云中的数据加密还需要考虑对加密造成威胁的边信道攻击（Side Channel Attacks，SCA）。边信道攻击是一种针对电子设备在运行过程中的时间消耗、功率消耗或电磁辐射之类的信息泄露而进行攻击的方法。当在加密时，边信道攻击可以对加密机制本身进行攻击。在云计算环境中，由于数据加密操作是在相同物理设施和多租户共享资源环境进行的，所以存在边信道攻击的可能性。网站 www.sidechannelattacks.com 上列出了各种不同类型的边信道攻击，有兴趣的读者可以参阅。

最后需要强调一下的是，尽管 CP 可能会承诺种种数据加密机制，但在目前加密技术体制下，还无法实现数据在云计算环境中的完全透明存储与处理，因此作为数据的最初提供者和最终拥有者，验证 CP 提供的加密机制是否符合用户的需要则是 CS 自己的责任。

24.3.2　数据屏蔽

1. 必要性

对于 CP 即将部署于云计算环境下的 SaaS 应用，由于其多租户及用户规模大等特性，在其上线前需要对 SaaS 应用系统进行严格而规范的测试，这包括系统功能测试、性能测试以及系统支撑能力测试等等。为检测系统在真实环境运行时可能存在的功能与性能的问题与缺陷，这些测试对测试数据的真实性和数据量都有较高的要求。同样，对于需要基于云计算环境开发机构自有应用的用户而言，也存在同样的需求。

这种需求需要 CP 或机构从自己的生产环境中导出数据，然后将这些数据作为大量的测试数据导入到待测应用系统中进行测试。通常对应用系统的测试由应用开发方实施或第三方进行，或至少包括应用开发方人员共同实施。

从生产环境中导出的数据通常会涉及云计算用户或机构的大量敏感信息，如果需要提交给应用开发方或第三方作为测试数据使用，必须经过加密保护。但正如在前一章节所述，数据加密可提供静态存储数据和传输中的安全保护方法，对于应用系统处理中的数据在目前技术体制下还很难实施加密。这样就需要一种技术实现将数据从生产系统导出时对其敏感信息按一定规则进行处理，在保证数据关联关系、数据格式不变和数据量满足测试需求的情况下提交给第三方使用，以降低敏感数据的暴露概率和提高非生产状态下数据的安全。

数据屏蔽（Data Masking）即是解决这一问题的关键技术。按 WIKI 百科给出的定义 [2]，数据屏蔽（data masking）是指在对存储的特定数据进行伪装（屏蔽）处理，以确保敏感数据用实际的（realistic）但非真实的（real）数据所替换，目的是保护敏感用户信息不为非授权环境外的人看到。

数据屏蔽可以避免非生产环境下数据库敏感信息泄露，数据清洗后，数据仍保持可用性，即

数据格式、规则及关联关系等不变，但数据内容进行了处理，因而敏感信息得到了保护。

数据屏蔽主要用于非生产环境中，其应用场景主要有以下几种 ：

- 软件开发与实施测试
- 软件用户培训
- 数据挖掘与研究
- 外包服务与数据有离岸要求的场景等。

2. 关键技术

有效的数据屏蔽要求数据必须以一种确定且不能被逆向工程的方式进行修改，即维护数据的功能外观（functional appearance）以用于测试，数据可以进行加密或解密，但必须保持其完整性。数据屏蔽常用屏蔽方法包括以下几种 ：

- 非确定随机化（Non-deterministic Randomization）：使用随机生成的、满足各种约束条件的值替换敏感字段，确保数据仍然有效。例如，将日期 2009 年 12 月 31 日替换为 2010 年 1 月 5 日 ；
- 模糊化（Blurring）：为原始值增加一个随机值，如使用一个包含原始值不超过 8% 的随机值替换储蓄账户值 ；
- 置空（Nulling）：使用空符号替换敏感字段中的值。例如，将社会保障号 404-30-5698 替换为 ###-##-5698 ；
- 变换（Shuffling）：变换敏感字段中的值的位置。例如，将邮政编码 12345 变换为 53142 ；
- 可重复的屏蔽（Repeatable Masking）：通过生成可重复且唯一的值，保持引用完整性（Referential Integrity）。如自始至终都使用 26-3245870 替换社会保障号 24-3478987 ；
- 替换（Substitution）：使用 “ 值替换表 ” 随机替换原始值。如从一个包含 10 万个姓名的列表中用 “ Mary Smith ” 替换 “ Jane Doe ”；
- 特殊规则（Specialized Rules）：这些规则适用于特殊字段，例如社会保险号、信用卡号码、街道地址和电话号码等，这些特殊字段在替换后仍保持结构上的正确性，并可用于工作流与检验和验证。例如，将 “ 100 Wall St., New York, N.Y. ” 替换为 “ 50 Maple Lane, Newark, N.J. ”，其中的每个随机值（门牌号、街道、城市和州）构成一个有效地址，可以通过谷歌地图等应用查找到 ；
- 标识（Tokenization）：标识是一种特殊的数据屏蔽形式，利用独特的标识符替换敏感数据，使信息可以在以后恢复到原始数据。例如，为灾难恢复目的而存储的数据必须在以后可以恢复，或者在业务运行过程中信息必须通过不可信的域时，标记化非常有用。

数据屏蔽分为静态数据（static data）屏蔽和动态数据数据（dynamic data）。下面对两者进行简要描述，感兴趣的读者可参考相关资料 [3-4]。

（1）静态数据屏蔽

静态数据屏蔽是一种传统方法，它是从生产数据库中抽取行记录，然后对数据值进行替换、重构、替换、置空、乱序等处理后作为列记录存入测试数据库里。数据屏蔽处理流程如图 24-2。

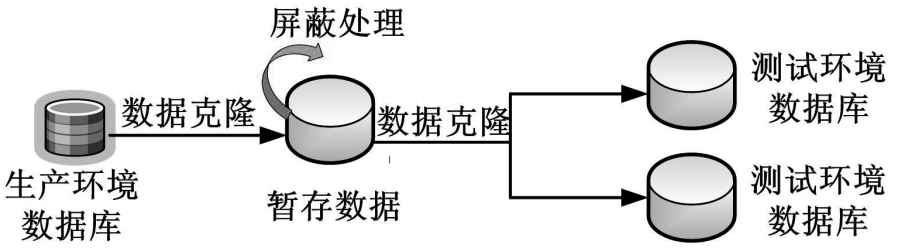


图 24-2 数据屏蔽过程

静态数据屏蔽存在以下几个局限性 ：

- 企业数据量大的情况下很难保证测试数据的引用完整性，因为随着企业数据量增大，这种屏蔽工作量激增，使得设计和创建屏蔽脚本变得冗长乏味且容易出错，对于企业数据量很大的情况，只能截短数据量或放弃屏蔽直接采用原数据 ；
- 静态数据屏蔽还存在一个重要不足就是对数据统计的影响，在 Oracle 系统里数据统计工作通常由系统自动完成，但由于静态数据屏蔽造成的数据分布（data distributions）和数据基数（cardinality）的巨大差异，使得在测试和生产环境中的 Oracle 查询优化器（query optimizer）会采用完全不同的两个函数进行数据统计 ；
- 在动态变化的数据环境下，静态数据屏蔽抽取、转换和装载等过程会消耗大量的计算机资源，因为在这种环境下数据通常需要不断地刷新与重新进行屏蔽 ；
- 静态数据屏蔽在将屏蔽数据导入到测试环境时还需要对测试脚本进行较大修改，该测试脚本用于创建生产环境下的入口数据。

因此，从总体上说，静态数据屏蔽在本质上对生产环境中的软件过程改进很难起到保护作用，静态数据屏蔽通常不适于生产环境。而且，由于静态数据屏蔽方法产生的数据与生产环境中的数据存在很大不一致性，静态数据屏蔽也很难用于精确测试。

（2）动态数据屏蔽

动态数据屏蔽是指在实时环境里在表示层（presentation layer）实现数据混沌化（obfuscation）处理的新技术。

动态数据屏蔽实现原理 ：通过在 SQL 网络协议层的插入一个数据侦听器，侦听任何来自上层应用的内向 SQL，然后采用屏蔽函数将其动态重写，这样数据在表示层进行屏蔽，而不用改变底层数据库或应用源码。

动态数据屏蔽为生产环境下的隐私数据保护提供了一种创新性保护机制，企业可以更安全地管理生产环境下的敏感数据。一个最常见的应用场景是生产环境下的数据库管理员（DBA），他们可以对生产环境中的数据库进行最高级别的操作，但有了静态数据屏蔽技术，企业可以把避免敏感数据信息暴露给 DBA 了。

与静态数据屏蔽在保持引用完整性时带来的复杂性相比，动态数据屏蔽消除了静态数据屏蔽繁琐的设计、耗时的前期规划，以及对数据提取、转换和加载（ETL）脚本的粗放型开发等带来的局限。

作为这种技术的商业实现，已经有很多公司在其数据库产品里集成了这方面的功能，如

Oracle Enterprise Manager、IBM DB2 和 SQL Server 等产品。Data Masking Pack 是 Oracle Enterprise Manager 里的一个重要产品包，该产品包内嵌丰富的数据修改规则，通过各种复杂算法，可自动批量快速完成对敏感数据的修改，从而保证克隆出来的数据库的数据量完全等同于生产库的数据量，敏感数据又做了伪装，如身份证号，电话号码，信用卡号码，姓名，日期，家庭住址，工资等，看起来是真实数据实际上是假数据，从而消除了敏感数据的泄露隐患。更详细的产品使用介绍感兴趣的读者请参阅相关资料 [5]。

3. 实践建议

数据屏蔽技术在不改变信息的格式、关联关系，满足了测试系统开发和质量测试需要，同时又保护了机构或云计算用户的敏感信息。在云计算环境下，采用数据屏蔽技术还需要考虑以下几个关键因素 ：

（1）确定数据屏蔽系统的部署范围

数据屏蔽最佳实践是要先确定数据屏蔽系统的部署范围。对于 CP 和 CS 而言，在实施之前要先明确在生产环境中有哪些敏感信息需要保护、哪些人员可以获得访问授权、哪些应用可以使用受保护的数据，以及这些数据驻留在生产或非生产环境中的什么地方等。对于一个具有复杂应用的大型机构或 CP 而言，由于应用和用户的复杂性和多样性，这个工作将是一个十分艰巨的工作。

（2）确定要采用的数据屏蔽方法

数据屏蔽最佳实践的第二个要素是，确定采用哪些数据屏蔽方法处理敏感信息。现有的数据屏蔽技术具有多种数据处理方法，但并不是所有的方法都可以保持有效的业务环境信息，需要根据不同业务需求确定采用不的处理方法。

（3）考虑引用完整性需求

数据屏蔽最佳实践的第三个要素是要充分考虑机构的数据引用完整性需求，这一点在一开始部署数据屏蔽系统时往往容易被忽略。在机构层面，引用完整性通常要求汇总信息，以满足不同业务范围间的资源共享需求。这意味着，来自同一业务范围应用程序的每种类型的信息都必须使用相同的算法 / 种子值来进行屏蔽。

例如，如果业务范围 A 的应用程序的数据屏蔽系统将客户的出生日期替换为 2010 年 1 月 5 日，则业务范围 B 的应用程序的数据屏蔽系统必须将相同的出生日期输入值也替换为 2010 年 1 月 5 日。利用引用完整性，如果一个企业级应用程序需要访问每个已屏蔽的出生日期，则该应用程序可以关联和操作来自这两个业务范围应用程序的其余数据。

然而，对许多大型企业或 CP 而言，在整个业务范围内使用单一的数据屏蔽工具一般并不可行。由于地域差异、业务需求、不同的 IT 管理组或者不同的安全 / 监管要求，每种业务范围可能会需要部署自己的数据屏蔽工具。尽管这种情况不影响通常数据屏蔽处理，但如果不同的数据屏蔽工具由于某种未知原因而不能同步，则可能会造成工作流难以继续。例如，对一个业务范围应用程序来说，出生日期的随机化可能完全可以接受。但对另一个业务范围应用程序来说，已屏蔽的出生日期必须属于一个该应用程序认为有效的预定义范围（例如超过 21 岁）。

（4）增强数据屏蔽算法的安全性

数据屏蔽最佳实践的第四个要素是，保护数据屏蔽工具使用的种子值或算法的安全性。由于数据屏蔽的基本原则是只允许获得授权的用户访问经授权的信息，所以数据屏蔽工具使用的种子值或算法无疑属于高度敏感的数据。如果有人掌握了数据屏蔽工具使用的可重复的数据屏蔽算法，则他 / 她可以对大量的敏感信息块进行逆向工程。数据屏蔽最佳实践是采用职责分离的原则，允许 IT 安全人员决定使用什么数据屏蔽方法和算法，并只能在初始部署阶段访问数据屏蔽工具以设置种子值，在部署完成之后 IT 安全人员则不能再访问数据屏蔽工具。

由于 IT 安全人员无权访问日常运营系统，而 IT 支持人员无权访问数据屏蔽算法，从而实现了严格的“ 职责分离 ”控制。但是，如果数据屏蔽工具未提供这种“ 职责分离 ”控制功能，则 IT 支持人员必须执行周期性的背景调查，并严格审计系统访问，以确保算法未遭泄漏。

对于需要运营复杂应用（如 SaaS 服务）的大型机构或 CP 而言，实施数据屏蔽具有诸多意义。但实施数据屏蔽并不像向现有应用程序中添加一个模块或开发一个专门实现数据屏蔽的系统那样简单，正如任何数据保护机制一样，在屏蔽第一条信息之前，企业需要制定计划、确定体系结构以及对未来业务如何运行的设想。

24.3.3　数据残余销毁

1. 概念及意义

信息安全保护不仅保护信息在制作、传输、存储及使用过程中的安全，还要包括信息在生命周期最后一个环节过程中的安全，即数据残余销毁。数据残余销毁是数据安全保护的最后一个环节，也是很重要的一个环节。因为无论是机构还是个人，都会有一些信息由于过时、重复存储、失去价值等原因而需要彻底删除，但日常所用的那些方法如删除、剪切或低级格式化等操作并不能真正的让那些数据彻底消失，那些信息仍然可以通过一些特殊手段恢复。而这些机构或个人不再需要的数据一旦被恢复且落到一些不法非子手里可能反倒成了“ 高价值信息 ”，他们可能利用这些信息来攻击或勒索机构或个人，或高价出售给竞争对手等。

在云计算环境下，存储资源的不定期租用与释放是一件很频繁的事情，如果 CP 在存储介质重新提供出去之前没有能将之前客户的残余数据彻底销毁，可能将会给 CS 所在机构带来极大信息安全隐患，如让机构处于不利的竞争环境下，或让机构名声扫地等，同时 CP 的服务信誉也因此受到不良影响而失去重要客户。

而对于 CS 而言，无论你使用哪种云计算模式（IaaS、PaaS、SaaS），数据残余都可能会导致其机构中的敏感信息无意中泄露给非授权的第三方。当使用 SaaS 或 PaaS 服务时，这种数据残余几乎肯定会使你在无意中把信息泄露出去。

回过头来看一下什么是数据残余及数据残余销毁。

按维基百科定义[6]：数据残余（data remanence）是指数据在被以某种形式擦除或移除后所残留的物理表现，存储介质被擦除后可能留有一些物理特性使数据能够被重建。而当这种带有数据残余的存储介质处于一个非受控的环境中（如垃圾桶或第三方）时，就可能造成敏感信息的泄露。

相应地，数据残余销毁是指利用各种技术手段将信息存储介质中的残余信息予以彻底销毁，

避免非授权用户利用存储介质的残余信息恢复原始数据，以达到保护敏感数据的目的。

美国国防部在其《国家工业安全程序操作手册》（DOD 5220.22-M）里对不同的数据残余销毁方式做了更细致的描述（见第 8-301 章节的“清理和消毒（Clearing and Sanitization）”）[7]：

认证主管安全机构（the Cognizant Security Agency，CSA）应给出信息系统媒介的清理、消毒、释放的指导意见。

清理（clearing）是指在对待清理数据提供合适保护措施情形下对存储媒介中的数据残余进行清除的过程。应该对所有的内存、缓存或其他可重用的内存都进行彻底清理，以有效阻止对之前存储信息的访问。

消毒（sanitization）是指对待消毒数据不提供合适保护措施情形下对存储媒介的数据残余进行清除的过程。信息系统资源在从一个分类信息控制环境释放出去之前或释放到一个更低级别分类信息控制环境中之前都应该被消毒。

数据残余销毁作为信息安全的一个重要分支，早已引起世界各国的重视。早在 1985 年，美国国防部（DOD）就发布了残余数据销毁标准（US.DoD.5 200.2 8-STD）；2000 年《中共中央保密委员会办公室、国家保密局关于国家秘密载体保密管理的规定》第六章第三十四条规定销毁秘密载体，应当确保秘密信息无法还原；国家保密局和军队安全局还要求涉密硬盘磁带信息销毁前不得带离办公区。

2. 关键技术

由于存储数据的载体的性质不同，可将存储介质分为纸质存储介质和电子存储介质。对于纸质介质存储的数据，一般通过采用物理焚烧或专用碎纸器等设备，可一次性彻底销毁信息。而对于电子存储介质中存储的数据通过一般的删除或格式化等手段很难将数据销毁，其中残留的数据需要通过专门的技术手段方可安全删除。

电子存储介质中的数据销毁根据采用的技术不同，总体上分为软销毁和硬销毁两种方式。硬销毁则通过采用物理、化学方法直接销毁存储介质，从而彻底销毁其中的数据。硬销毁又可分为物理销毁和化学销毁两种方式。物理销毁可分为消磁，熔炉中焚化、熔炼、粉碎等方法，化学销毁指利用化学物质对存储介质进行破坏性处理。物理和化学销毁仅适用于一次性使用的存储介质上的数据（残余）销毁。在云计算环境下，存储资源（主要是磁盘）是可重复使用的资源，通常会采用软销毁，以下重点介质软销毁技术。

软销毁又称逻辑销毁，数据软销毁通常采用数据覆写法，即通过数据覆盖技术销毁数据。数据覆写是将非保密数据写入以前存有敏感数据的磁盘簇的过程。磁盘上的数据都是以二进制的“1”和“0”形式存储的。使用预先定义的无意义、无规律的信息反复多次覆盖硬盘上原先存储的数据，完全覆写后就无法知道原先的数据是 0 还是 1，这样即可达到清除数据的目的。数据覆写法处理后的硬盘可以循环使用，适应于密级要求不是很高的场合，特别是需要对某一具体文件进行销毁而其他文件不能破坏时，这种方法更为可取。

根据数据覆写时的具体顺序，软件覆写分为逐位覆写、跳位覆写、随机覆写等模式。根据时间、密级的不同要求，在实施软销毁时可组合使用上述模式。

美国国防部的 DOD5220.22_M 标准和北约 NATO 制订的多次覆写标准规定了数据（残余）

销毁时覆写数据的次数以及覆写数据的形式。美国国防部制订的硬盘清洗规范，则要求数据必须对所要清除的数据区进行三次覆盖，在不了解存储器实际编码方式的情况下，为尽可能增强数据覆写的有效性，确定一个合适的覆写次数与覆写数据的格式非常重要。数据（残余）销毁的有效方法是对要销毁的数据的存储位置进行多次覆写。

数据覆写是较安全、最经济的数据软销毁方式，数据覆写法处理后的硬盘可以循环使用。需要注意的是，覆写软件必须能确保对介质上所有的可寻址部分执行连续写入，如果在覆写期间发生了错误或坏扇区不能被覆写时，或软件本身遭到非授权修改时，处理后的介质仍有可能恢复数据，因此，还必须借助硬销毁方式。

3. 实践建议

下面是针对 CS 和 CP 在数据（残余）销毁实践时的一些建议 ：

（1）应根据敏感数据级别，对云上存储数据的存储位置进行合理规划。

在采用存储虚拟化技术以后，数据存放的物理位置位于多个异构存储系统之上，对于应用而言，并不了解数据的具体存放位置，通常应用所带的数据删除功能（如文件删除）并不能真正删除数据，只是删掉了数据索引的入口，因此在存储资源安全释放之前通常需要对数据残余进行彻底销毁。

而对云环境下的存储介质（磁盘或磁带）进行数据残余销毁（以数据覆写为主的软销毁）技术实施的复杂度通常取决于数据残余可能涉及的敏感数据级别，级别越高，需要采用的技术实施复杂程度越大。因此，为便于对数据残余进行统一销毁，需要在数据存储位置分配时进行合理规划。建议根据存储的数据级别，对物理存储位置进行规划，原则是尽可能让同一级别的数据存放在同一物理存储位置上，不同敏感级别的数据存放在不同物理存储位置上，至少在不同的逻辑分区上。这需要在存储虚拟化管理软件里统一配置。

（2）无论是 CS 还是 CP 在将云中存储资源释放的时候都需要采用专用工具或软件来核查存储介质上是否存在数据残余，如有残余，要根据不同数据分级采用相应数据残余销毁技术对残余进行彻底清除。

对于 CS 而言，无论你使用哪种云计算模式（IaaS、PaaS、SaaS），数据残余都可能会导致你机构中的敏感信息无意中泄露给非授权的第三方。当使用 SaaS 或 PaaS 服务时，这种数据残余几乎可以肯定地说会让你在无意之中就把机构的信息泄露出去。对于 CP 而言，在将前一个租户退租的存储资源未经过严密检查而将带有数据残余的资源租给了下一个用户，虽然这通常不会使 CP 机构本身安全性受到重大影响，但由于其所承担的服务安全连带责任也将使得其难脱法律责任。此外，其服务信誉与形象将会因此大打折扣。因此 CS 应该在将云存储资源退租给 CP 之前，或 CP 在将重用的存储资源出租给其他 CS 之前，必须确保其释放的或提供的存储空间上所有的数据残余得到有效清除，这些信息可能包括存放在硬盘上和内存中的。

此外，在销毁数据残余时，为彻底清除这些存储介质上的敏感信息，应对敏感数据存储的所有存储位置进行物理写覆盖。对于涉及敏感数据级别高的场合，根据 CS 的要求，可能还需要采用物理消磁等更低级别的方式进行彻底销毁。

（3）业界相关标准提供了敏感数据销毁最佳实践参考，建议大型 CS 或 CP 在进行敏感数据

销毁时应遵循相关标准。

敏感数据销毁通常会涉及到机构的隐私信息或关键业务数据，没有按照严格的敏感数据处理流程的随意销毁可能会给机构或个人带来法律责任。

另一方面，随着云计算下监管体系及服务规范的建立，行业和政府层面已经或即将出台云计算相关要求，这包括云中数据安全保护要求。如塞班斯法案（Sarbanes-Oxley）要求将企业的记录和文件销毁，并且必须对文件销毁进行认真检测。否则，企业主管就很可能会受到起诉。而我国国家保密局已经制定颁发了强制标准《涉及国家秘密的载体销毁与信息消除安全保密要求》，要求对于涉密载体的销毁要遵照此标准执行。此外，美国国家标准技术（NIST）也发布了“媒介清理指南”（Guidelines for Media Sanitization）专刊（800-88）[7]，虽然该刊物只是指南，但在目前数据残余销毁还没有其他业界标准的情况下，很多公司，尤其是那些受管制行业的机构，都自愿遵从 NIST 指南和标准。而美国国防部的国家工业安全程序操作手册（DOD 5220.22-M）则提供实用的数据（残余）销毁操作建议[8]。

24.3.4　数据沿袭

在云计算环境下，云计算审计者或业务管理者可能会遇到这类问题：“我当前查看的报告的来源是什么？”、“我不信任这份报告的数据。这份报告的数据都经过了哪些转换？”等。同样，而云计算中的 IT 分析师可能会遇到这类问题：“如果修改对这个列的约束条件，哪些资产和流程会受到影响？”以及“如果删掉这个数据库，哪些应用会受到影响呢？”等。

从上述的问题可以看到，云计算中的 IT 技术人员、业务管理人员或审计人员，都需要一种技术手段能够对云计算中的数据从制作、传输、存储、使用到销毁全生命周期过程进行全面可视化自动流程跟踪，以提供机构内关键信息资产的数据可追溯的报告。数据沿袭即提供了这样一种技术手段，它在满足合规性或监管要求的同时，也提供了大型机构或 CP 进行数据可视化治理的重要手段。

目前业界对数据沿袭（data lineage）还没有一个统一的定义。参照国外在数据管理领域从事产品研发的公司 Silwood Technology 对数据沿袭的定义[9]，数据沿袭是指对数据来自哪里、流向哪里以及在所流经的路径上产生的变化整个过程的跟踪和管理。数据沿袭是机构进行元数据（metadata）管理的基础，图 24-3 给出元数据管理对数据沿袭的依赖关系，元数据管理提供对数据源系统发现、源数据清洗（抽取转换装载，ETL）、技术规则定义、数据转换、业务规则定义、报告生成及报告使用等全过程的管理。

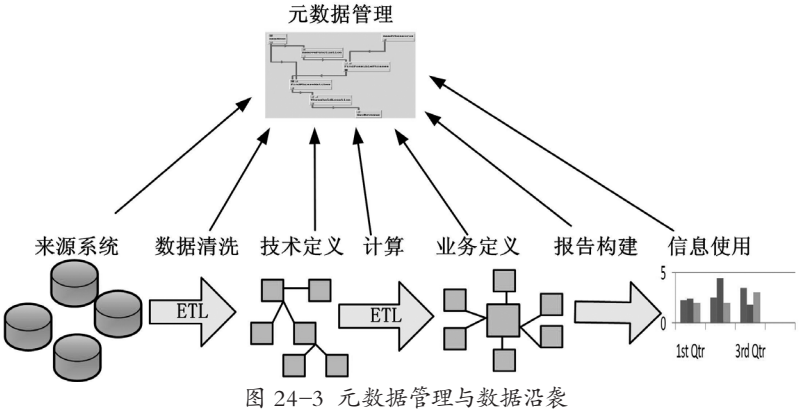


图 24-3　元数据管理与数据沿袭

数据沿袭提供了源数据到派生数据整个处理过程的变化历史。数据沿袭有两种方式，一种是后向模式，该模式向后追溯直到源系统的原始数据项（ancestor items）；一种是前向模式，这种模式是向前跟踪直到目标系统的派生数据项（descendant items）。

在云计算环境中实施数据沿袭的意义：

- 满足合规性（如巴塞尔协议，一个关于银行行业的法律法规[10]）要求；
- 支持深度数据分析（In-depth data analysis）；
- 便于授权管理（Authorization management）；
- 提供影响评估支持（impact assessment support）；
- 支持操作风险评估（Conduct risk assessment）；
- 确保数据治理原则（Ensure data governance principles）；
- 提高重用度与标准化（Enables reusability and standardization）。

数据沿袭技术研究相对要滞后于产品研发，目前国外市场上已经有许多公司提供一些数据沿袭相关产品[11]，如来自加拿大公司 ERwin 提供的 Data Modeler 和 Sybase 公司提供的 PowerDesigner 等数据建模工具，可以支持数据沿袭处理；而资源库供应商如 ASG 公司和 IBM 则分别提供了 Data Warehouse Metadata Management Application 和 Infosphere Metadata Workbench 产品，这些产品允许用户记录数据沿袭规则和映射关系，并提供影响分析功能，该功能有助于用户理解数据条目变化效果。Silwood 公司提供的 Saphir 产品可以提供 ERP 或 CRM 环境下源数据的相关知识，并能直接与主流的建模和资源工具相对接。

24.3.5　数据备份与恢复

数据备份与恢复是保护数据安全的预防性或补救性措施，其重要性也是不必要强调的。从安全的角度看，需要注意以下几点：

- 所有用于备份的外部存储介质应加密并保存在一个保险柜中。
- 内嵌 / 近线系统备份文件应与原始数据具有相同级别的安全性。
- 不同级别的安全数据，应使用与不同策略相匹配的安全级别分开备份。
- 对于加密密钥 / 证书这类更敏感的信息应采取更为严格的安全策略和处理程序。如果可能的话，应遵循供应商建议的备份方法。
- 应严格控制和批准明文模式中的敏感数据恢复。

关于数据备份与恢复，第 28 章“云计算业务连续性保障”一章里有更详细描述。

24.3.6　访问控制

数据的访问控制包含两个层面含义，一个是通过应用系统的身份认证与访问控制实现对数据资源的操作权限，这块内容将在第 25 章“SaaS 服务安全”章节里加以描述；还有一个是通过数据库管理系统实现对数据库数据的访问控制，该功能通常集成于数据库管理系统中，这里不再详述。

24.4　本章总结

数据是云计算环境中的核心要素，保护数据安全是云计算安全的重中之重。

从数据生命周期角度来看，数据在每一生命活动历程中需要相应的安全保护措施，如数据生成、数据传输、数据存储、数据使用、数据安全删除分别侧重于数据备份、传输加密、存储加密、访问控制、数据残余销毁等技术实施。

云计算环境下 CS 及 CP 面临不同的数据安全风险。CS 主要面临数据控制权丧失后所引发的一系统风险，如特权访问、数据管理权限、数据位置、数据隔离、数据恢复、调查支持和持续服务等 ；而 CP 则面临不够安全加密数据引发的泄密风险、网络钓鱼行为导致的数据安全风险，以及审计监控实施风险等。

针对上述数据安全风险，本章重点阐述了云计算环境下可采用的数据安全保护机制。数据存储加密对静止态数据进行加密，有四种加密方法，磁盘级、目录级、文件级和应用级。根据加密实施位置，可分为三种方式，分别是主机端加密、云中专用硬件和存储设备加密。传输加密是对活动态数据进行加密，有三种方式实现 ：在非安全网络传输加密的数据、基于传输的数据由底层协议加密和安全电子交易。对于非电子商务类的应用数据安全访问，多采用基于 SSL/IPSec 的加密方式。

数据屏蔽和数据访问控制则分别提供了非生产环境和生产环境下数据安全使用的技术保护手段。数据屏蔽又分为静态数据屏蔽和动态数据屏蔽，对于大型机构或 CP，在上线大型前多用动态数据屏蔽进行系统精确测试。数据残余销毁是数据生命周期最后一个环节对数据的安全处理。云计算环境下主要采用软销毁即数据覆写方式。数据沿袭是对云中的数据从生成、使用、变化和消亡全过程的跟踪记录。对多租户的云计算环境，数据沿袭对 CP 和 CS 都有重要意义。

参考文献：

[1] IPsec, Wikipedia, http://en.wikipedia.org/wiki/IPsec

[2] Data Masking, Wikipedia , http://en.wikipedia.org/wiki/Data_masking

[3] Paul Preston，" Data Masking: Counter Attack to Identity Theft "，2006，http://doc.wowgao.com/ef/presentations/PPCamouflage.ppt

[4] Oracle, Oracle Data Masking Pack, 2011, www.oracle.com/us/products/database/data-masking-161222.html

[5] Oracle,“ 使用数据屏蔽包替换敏感数据 " , 2013, http://www.oracle.com/technetwork/cn/tutorials/datamask-082472-zhs.html

[6] Data Remanence, Wikipedia , http://en.wikipedia.org/wiki/Data_remanence

[7] Richard Kissel, Matthew Scholl, Steven Skolochenko, Xing Li, “ Guidelines for Media Sanitization "，NIST Special Publication 800-88

[8] US DoD, National Industrial Security Program, DoD 5220.22-M, Feb., 2006 http://www.fas.org/sgp/library/nispom/nispom2006.pdf

[9] Silwood Technology, Data Lineage, http://www.silwoodtechnology.com/data-lineage.html

[10] Richard Kissel, Matthew Scholl, Steven Skolochenko, Xing Li, Guidelines for Media

Sanitizlation，NIST Special Publication 800-88

[11] Bill Lewis, Data Lineage: The Next Generation Exploring Data Dependencies in Business Applications, http://www.tdan.com/view-articles/8151

第25章　IaaS和PaaS服务安全

25.1　IaaS服务用户需重点关注的安全问题

前面已经从服务部署的角度对云服务数据中心服务设施多个层面的安全技术实施进行介绍，那个层面主要是云服务提供商（CP）需要关注的技术实施，本章将着重从服务使用的角度来分析IaaS 和 PaaS 服务存在的主要安全问题，以便 CS 在使用云服务时正确评估自己的安全策略实施。无论是公有云还是私有 IaaS 服务，IaaS 服务的安全问题主要由用户自己来处理，这包括操作系统及其应用程序的安全。而对于公有 PaaS 服务安全，用户则需要负责基于平台开发的应用程序的安全，同时也需要对平台自身的安全机制有所了解。

用户在使用 IaaS 服务时所面临的最大问题就是操作系统（OS）和服务中隐藏的脆弱点。这里之所以用 “ 最大的问题 ” 是因为在基础设施这个层面，用户所面临的安全威胁呈现几何增长的态势。其次是远程管理、DNS 等问题，下面分别进行说明 [1]。

25.1.1　系统基础服务安全风险及应对措施

在 IaaS 服务模式下，处理操作系统和服务中所隐藏的薄弱环节是用户可以采取的最重要的安全措施。当前，Linux、基于 Linux 演变出来的操作系统以及 Windows 操作系统是公共 IaaS 服务的主流选择，至少占据了 90% 的市场。这两类操作系统以及运行在此类操作系统上的服务都存在一定弱点（如 Windows），或者说将会存在一些弱点（如 Linux）。操作系统和服务的漏洞迟早会被通过诸多的出口释放出来，因为在许多案例中，服务的开发工作都是公开透明的。

操作系统漏洞涉及到操作系统基本功能，比如说 TCP/IP 网络协议、系统调用、系统数据库及 Windows 安全帐号管理（SAM），通常安装操作系统功能越全系统存在的漏洞可能越多。而运行在操作系统之上的服务是指那些利用操作系统基本功能来完成任务的应用，如 DNS 服务器、Windows File Sharing 或 NetBIOS。以下是处理这些安全风险的应对措施 ：

1. 删除功能或服务

删除操作系统功能 ：对于 Linux 而言，可以对核心操作系统的代码进行修改，重新构建内核或系统数据库，然后对其进行再编译 ；Windows 下则需要定制安装软件。

删除服务 ：对于 Linux 而言，可以删除可执行文件。对于 Windows 而言，仅部分独立服务可以通过删除其可执行文件方法删除。

2. 禁用功能或服务

禁用操作系统功能 ：对于 Linux 而言，通常可以对其核心配置文件进行修改，或是重建内核。对于 Windows 而言，可以通过 “ 服务 ” 按钮或在 “ 添加 / 移除程序 ” 控制面板中通过 “ 添加 / 移除 Windows 组件 ” 来实现禁用。在此操作过程中，需要查看相关属性，因为移除或禁用操作系统功能或服务会给系统造成不可恢复的影响。

禁用服务 ：对于 Linux 而言，通常可以在 inetd/rc 结构（或是类似结构）中实现这点。对于

Windows 而言，则可以在“ 服务 ”组件里停用一些不必要的服务或设置为需要时手工打开。

3. 阻止针对服务的可能攻击

可以通过使用基于主机的防火墙来确保进入主机的流量都是可信的。但这仅对于处理所有服务相关的威胁很有效，并不能保护防火墙所依靠的操作系统功能，如基础网络功能中的漏洞还是会对防火墙有所影响。

25.1.2　远程管理风险及应对措施

IaaS 服务资源在 CP 侧，用户通常需要远程访问和管理申请的 IaaS 资源。在远程管理过程中，通常有以下几种方案解决用户安全使用 IaaS 服务的问题 ：虚拟私有网（VPN）、远程桌面、远程 shell 及 Web 控制台用户界面等 ：

- VPN ：提供一个到 IaaS 资源的安全连接（如隧道），通常在远程管理应用程序不能保护其传输的数据时使用，最常见的解决方案包括 PPTP（点到点隧道协议），L2TP（二层隧道协议），IPSEC（Internet 协议安全）或 SSL/TLS（安全套接字层或传输协议安全）。
- 远程桌面 ：为图形界面工具提供了一个接口，通常在操作系统本身不支持基于命令行的管理时使用（如 Windows 操作系统），最常见的解决方案是 Windows 远程桌面或虚拟网络计算机（VNC）。
- 远程 shell ：为系统管理提供基于命令行的接口，这种环境的性能是最好的，最常见的解决方案是 SSH。
- Web 控制台 UI ：提供一个自定义远程管理界面，通常它是由 CP 开发的自定义界面（如管理亚马逊 AWS 服务的 RightScale 界面）。

在上述远程管理解决方案中，凭证缺乏或弱口令是远程管理中主要安全风险，如使用空密码、简单的口令、使用重复使用的用户名和密码或长期共享的密钥等。此外，远程访问的协议实现缺陷（协议漏洞）也是远程管理的另一个潜在风险。

下面是针对这些风险可能的应对措施 ：

① 缓解认证威胁的最佳办法是使用双因子认证，或使用动态共享密钥，或者缩短共享密钥的共享期。

② 不要依赖于可重复使用的用户名和密码。

③ 对于下面这些协议 ：

- VPN ：缓解该协议风险的唯一办法是改变协议（如 IPSEC over PPTP），或者是升级到打过补丁的协议版本（如 SSL v3 或 TLS v1，TLS v2）。
- 微软的远程桌面（RDP）：使用强加密，并要求服务器认证。
- VNC（远程桌面的一种）：在 SSH 或 SSL/TLS 隧道上运行它。
- SSH ：使用 RSA 密钥进行认证。
- Telnet（远程 Shell 的一种）：不要使用它，如果必须使用它，最好通过 VPN 使用。
- Web 控制台 ：通常基于 Web 的协议是 HTTP 或基于 SSL/TLS 的 HTTP，因此 HTTP 或 SSL/TLS 存在的安全隐患，Web 控制台也一样存在。

④ 对于自身无法保护传输数据安全的程序，应该使用 VPN 或安全隧道（SSL/TLS 或 SSH），推荐首先使用 IPSEC，然后是 SSLv3 或 TLSv1。

25.1.3　DNS威胁及应对措施

1. DNS 威胁

DNS 服务提供域名与 IP 地址之间的映射与解析，因此任何阻止域名解析或返回错误数据的情况都可能影响 IaaS 服务，尤其是后者甚至可能会给 IaaS 用户或提供商带来某种损失，如经济或声誉上的。

到底有哪些来自 DNS 的威胁会影响 IaaS 服务？根据目前 IaaS 服务应用场景，主要有以下五种 DNS 威胁会影响 IaaS 服务 ：

- 缓存中毒 ：当一个 DNS 服务器没有 “ 域名 -IP ” 映射信息时，它必须找到另外一个拥有这些信息的 DNS 服务器。当它接收到请求应答时，它通常将该信息存放于缓存中以减少了不必要的带宽消耗及客户端延迟，提高 DNS 基础设施的可扩展性。缓存中毒就是指服务器接收到包含错误信息的响应。在缓冲数据的生存期间内，缓存中毒可以扩展到其他名字服务器（如下一层服务器等等），引发较大面积的 DNS 域名欺骗情况的发生。恶意缓存中毒也称为 DNS 欺骗。
- 不安全的动态更新 ：这是另外一种将恶意数据引入 DNS 服务器的机制，其效果类似于缓存中毒。动态更新就是要实现在不影响服务的情况下更新数据。通过 DNS UPDATE 协议，动态更新的客户端通知服务器需要更新哪些数据，服务器在不停止服务的情况下，更新解析数据。但如果递交更新的数据是包含恶意的数据，就会造成对该信息的查询都会返回恶意数据。
- 区域信息泄露 ：用户可以通过多个 DNS 服务器提高域名解析的可靠性和容错性，这就需要利用 DNS 区域传输（复制和同步）来实现管理区域的所有 DNS 服务器中域的记录相同。负责同一子域解析的 DNS 服务器中，需要及时对 DNS 数据记录进行同步，区域传输是区域数据文件从主 DNS 服务器装载到辅 DNS 服务器，使两者数据一致的一个过程。在区域数据文件传送过程中，攻击者可以通过监听网络流量，获取数据中内部网络拓扑、DNS 域名、主机名、IP 地址以及操作系统等信息，从这些信息中可能获取其他存在漏洞的主机或识别其他敏感的网络资源。
- 服务堵塞 ：攻击者采用递归查询的方式攻击 DNS 服务器，使它们无法对正常合法的查询做出响应。如果没有 “ 域名 -IP ” 的解析服务，很多情况下用户将无法访问其 IaaS 服务。
- 域名劫持 ：攻击者通过非法手段获得某个 DNS 服务器的账号、密码信息后，在该 DNS 服务器上添加相应的域名记录，使得用户对某个域名的访问进入到黑客所指定的目标地址。这类针对 DNS 的攻击显得更为直接，也比较难奏效，但一旦攻击成功，造成的损失将是十分巨大的。攻击将导致该 DNS 服务器负责的子域中，所有用户访问的非法重定向。一般重定向所指向的网页与原网页内容、风格十分相似，黑客可以通过这种手段获取用户的关键信息，比如帐户名、密码等等，造成难以预料的经济损失。

2. 应对措施

如果用户控制 DNS 服务器，可以采用手工操作消除或减少这些威胁 ；如果是 CP 或第三方服务方控制 DNS，则需要由相应的供应商来处理。以下是对于消除 DNS 威胁的若干建议 ：

- 只使用 IP 地址或一个本地主机文件 ：如果不使用 DNS 进行域名 -IP 的解析，那么以上所述的 DNS 问题就变得无关重要。虽然这似乎不切实际，但是如果您拥有的 IaaS 数量有限，且可以指定静态 IP，这是一个现实安全的解决方案。
- 随机事务 ID ：确保 DNS 服务器具有一个适当的随机事务 ID，每一个 DNS 查询都被分配一个 ID，其值的随机性将使攻击更难以执行。
- 源端口随机化 ：将 DNS 服务器配置成 “ 查询源端口随机化 ”，攻击者将需要知道事务 ID 以及事务发送的端口（即随机源端口将不一定是 53）才可伪造数据。
- 安全动态更新 ：DNS 服务器和客户端配置成只接收安全动态更新，或限制动态更新源的 IP 地址范围。
- 限制区域传输 ：这将阻止攻击者轻易地收集到 IaaS IP 和主机名等敏感信息。
- 更新 DNS 软件版本或完善 DNS 协议 ：使用新版本的 DNS 服务器软件（如 BIND）可以有效地防止以往 DNS 软件所存在的安全漏洞，像 BIND4 与 BIND8 存在的非随机端口风险已经在新版本中得到修正。此外，一些对 DNS 协议的扩展补充，如 DNSSEC，也从很大程度上解决了 DNS 安全性的问题。

DNS 是一个基础性的互联网协议。互联网及其上的所有服务在缺乏 DNS 的情况下都不能充分发挥作用。但是如同电力一样，在不出现故障的情况下人们似乎并不觉得其存在。在使用 IaaS 时，时刻谨记保持这种服务的重要性及其周围安全的问题。

最后还需要强调一下，使用 IaaS 服务的用户，不仅需要对 IaaS 服务中的操作系统、远程管理及 DNS 服务中的安全风险有充分的认识，还需要对其部署在公共 IaaS 云中的网络应用程序的安全负有完全责任，如部署于 IaaS 资源上的应用程序必须设计为内嵌对抗常见网络漏洞（如开放式网络应用程序安全项目 OWASP 排名前十位的漏洞）的标准安全对策，遵守常见的安全部署做法，并定期测试漏洞，给部署于 IaaS 上的应用程序和运行时平台安装补丁。最重要的是，应当在软件开发生命周期中内嵌安全特性，根据应用自身的特点来实施管理认证和授权，并在设计和实施应用程序时遵照 “ 最小特权 ” 运行时模式。

25.2　IaaS服务用户安全检查清单

IaaS 服务模式可以使 CP 向用户提供 IT 基础设施服务，用户无需再考虑硬件、支持网络和存储基础设施。用户所要做的只是查看服务目录，从目录中选择他们想要的基础设施资源，包括计算或处理器资源、内存、存储和网络资源，并决定要使用服务多长时间。

IaaS 分私有云和公有云两种。一般来说，私有云安全与您在自己的数据中心和内网环境中的部署安全没有什么太大区别，仍旧是硬件 / 物理安全、主机操作系统安全、应用程序安全、网络安全和数据安全等。因此这里所提供的 IaaS 主要是以公有云模式的 IaaS 服务。对公有云模式的 IaaS 服务，安全职责是由用户和供应商共同承担的。双方应该清楚理解、界定安全职责的划分并达成一致共识。用户针对公共 IaaS 服务提供商应重点关注的安全清单包括 ：

- 正确评估云服务提供商，检查他们是否拥有行业认证，如 SAS 70 Type II。对行业认证保持适当跟踪以评估可能出现的新的想法与认证并始终保持处于云计算领域的发展前沿 ；

- 研究可以采用哪些实践防止云服务租户间的安全泄露。你应该得到任何租户系统之间的数据都不会遭到泄露的保证。这种保证必须与详细的技术资料一同加以保存备份，这些技术资料说明了采用何种控制措施来防止跨租户的数据泄漏 ；
- 审查服务提供商的服务历史，获得客户的参考信息，并询问他们之前对服务提供商在隐私、可靠性和安全漏洞处理方面的感受 ；
- 确保你有适当的监测和日志记录，配合其他的检测措施，可以验证服务提供商是否达到承诺的服务安全级别，并可让你知道是谁在试图访问某个文件、是谁打开并读取了某个文件等 ；
- 确保基础设施的安全要求写进你与 IaaS 服务提供商的服务合同中，包括审计条款，这样你或第三方可以定期验证所需的控制措施是否到位 ；
- 获得可靠的服务级别协议（SLA）。SLA 要求服务提供商按规定的级别提供系统 / 服务的可靠性 ；
- 不要接受他们对其服务进行事先毫无宣告的修复策略 ；
- 仔细检查服务提供商的数据恢复政策，以明确如果你要解除合同将需要花多长的时间来取回你的数据，以及他们需要多长时间使数据下线 ；
- 确保你和服务提供商的网站之间的数据传输应有足够强的加密标准和密钥管理策略 ；
- 确定你的客户不是安全链中的薄弱一环。如果指定网页浏览器可以用来访问服务，则要高度注意浏览器自身安全和更新问题。如果可能的话，必须让客户首先登录到你网络以获取其在 IaaS 服务提供商网站上的公司信息 ；
- 如果客户可以访问你租用的服务，要确保自己始终拥有对域名和控制域访问的所有权 ；这样，如果你终止和服务提供商的关系，你将不必重新培训客户怎样使用正确的 URL 找到你 ；
- 确保有渠道或接口允许您设置或自定义应用 / 服务器 / 网络平台的安全性来满足您的业务需求。

25.3　PaaS服务用户需重点关注的安全问题

尽管 PaaS 服务提供商通常会负责 PaaS 服务平台及其基础设施（如防火墙、服务器、操作系统、应用开发及运行环境等）这些层面的安全技术实施，但控制和保证基于 PaaS 服务开发的应用安全的任务还需要用户自己来承担。这里主要指基于 PaaS 平台的应用开发和部署层面的安全。此外，当应用部署于平台之上对外提供服务时，后面讲到 SaaS 模式面临的那些安全威胁也同样存在，那些威胁可参考后面章节提供的相应的应对措施进行处理。

在用户评估可以接受某个 PaaS 服务商提供的 PaaS 服务后，接下来的四个方面将是 PaaS 服务用户需要重点关注的安全问题。PaaS 服务提供商为用户提供了应用开发的全生命周期服务，包括用来实现云服务管理和交互的接口或 API、运行环境和 Web 应用服务、远程访问等。相应于上述服务，PaaS 服务用户需要关注的安全问题主要有以下几个方面 ：安全相关的 API、应用安全部署、远程安全访问等。PaaS 用户应该关注的另一个安全问题是服务锁定风险，这个虽然放在最后说明，但它是用户在决定使用 PaaS 服务前首先需要考虑的问题。上述这些安全问题大都可以依靠用户自己的能力解决，而不需要过多地依赖于 CP。

25.3.1　安全相关的API

PaaS 服务提供商通过服务接口或 API 与用户应用进行交互，这种交互包括服务提供商提供的服务能力、对用户的管理和监控等。PaaS 服务要能防御无意或恶意地攻击行为，这些接口或 API 还必须能提供安全机制，包括认证、访问控制、加密和活动监控等。当然，服务提供商或第三方组织也可基于这些安全接口或 API 向用户提供一些增值服务。但正是因为这些服务接口或 API 的存在，不仅增加 PaaS 服务提供商的实施复杂性，也给用户带来了风险，因为你必须得把一些凭证信息递交给 CP 或第三方才可以使用这些云服务。而一套安全设计薄弱的接口或 API 更会让用户和服务提供商面临更大的风险。以下是安全设计存在缺陷的 API 示例 ：

- 允许匿名访问或重复使用的凭证与密码 ；
- 明文认证或内容明文传输 ；
- 不可扩展的访问控制及不恰当的授权 ；
- 有限的监控与日志记录能力 ；
- 未知的服务或不恰当的 API 依赖关系等。

针对上述情况，用户可以采取的补救措施有 ：

- 分析 CP 的安全实施机制 ；
- 确保用加密传输方式实现强认证和访问控制 ；
- 正确理解与安全相关的 API 间的相互依赖关系等。

基于 PaaS 开发部署的应用其安全需要 PaaS 应用开发商配合，大多数 CP 已将安全机制很好地集成在其平台服务架构中，开发人员需要熟悉平台 API 及与应用部署、配置和监控相关的安全设计内涵。开发人员必须熟悉平台的这些被封装成安全对象和 Web 服务的安全特性，才能通过调用这些安全对象和 Web 服务实现安全的用户应用。

25.3.2　应用安全部署

当你在利用 PaaS 服务接口或 API 开发调测完应用后，就需要在 PaaS 服务环境上部署运行你的应用。

做过一定规模项目实施的人都知道，应用在默认配置下能安全运行的概率几乎为零。因此，如果你需要在云基础架构中部署并运行你的应用时，你就必须改变应用部署时的一些默认安装配置。下面这些应用软件或服务组件，无论是在云架构还是在传统的 IT 架构中，大约 80% 以上的应用都会用到它们，因此，熟悉他们的安全配置流程，对如何确保部署真正安全的应用具有重要意义。

- LAMP ：在利用 LAMP（一种基于 Linux、Apache、MySQL 和 PHP 环境的通用配置）环境作为你的应用最终的运行环境时，你需要熟悉 Linux、Apache、MySQL 和 PHP 这些软件的配置和操作，需要掌握这些软件更多的安全配置技巧 ；
- WISN ：在 Windows 环境下，你需要了解 Internet Information Services（IIS）、Microsoft SQL 和 .NET 安全配置的能力，它是 Windows 环境的 LAMP 环境。

对于上面这两类运行环境，尤其需要重点关注以下几个方面 ：

- 清理安装后遗留下来的默认和示例文件及目录 ；
- 注销或修改用以 Web 或 SNMP 方式进行应用管理的默认用户名和密码 ；
- 删除或关闭这些软件提供的一些不必要的服务，如 WebDAV、FrontPage、Lightweight Directory ACSess Protocol（轻量级目录访问协议 LDAP）、Simple Network Management Protocol（简单网络访问协议 SNMP）等等。

关于其他安全配置信息，还可以到具体的软件提供商网站来查看其安全配置建议。

25.3.3　远程安全访问

无论是开发、调测、部署应用还是访问运行在 PaaS 平台上的应用，PaaS 用户都需要通过网络连接与位于云侧的 PaaS 服务平台进行通信。因此远程连接安全将成为又一个需要重点关注的安全问题。通常情况下，为了确保用户及服务提供方的身份真实性、交互数据的完整性，以及数据在网络传输过程中不被截取及窃听，需要在服务提供方和用户之间建立某种安全通道（VPN）。对于基于 Web 模式的应用，相对 IPSec VPN 而言，SSL VPN 因其部署简便、维护成本低，已成了网络应用服务提供方与服务消费方之间建立安全通道的最常用选择，也是目前大多数 CP 的普遍选择。

SSL（Secure Sockets Layer）协议是一套 Internet 数据安全协议，它位于 TCP/IP 协议与各种应用层协议之间，为数据通讯提供安全支持，当前版本为 3.0。由于其部署简便，已被广泛地用于 Web 浏览器与服务器之间的身份认证和加密数据传输。将 SSL 协议应用在具体 Web 应用上就成了 HTTPS。HTTPS 可以保护用户的页面请求和 Web 服务器返回的页面数据不被窃听，SSL 及后来的 TLS（传输层安全）协议还允许 HTTPS 利用公钥加密验证 Web 客户端和服务器的身份。

但正如针对 Windows 的攻击随着用户数量的增加而倍受攻击者的关注一样，SSL 在获得广泛应用之时，也正是其可能遭致攻击的时刻，众多黑客社区也正对其进行了深入研究，可以预计 SSL 在不久的将来将成为一个主要的病毒传播媒介。因此，PaaS 服务用户必须了解这种局面，并采取可能的办法来缓解 SSl 攻击，而不致将自己的应用暴露于诸多潜在的威胁之下。

同对待操作系统等软件存在的缺陷问题一样，也可以依赖更新这些协议的最新补丁或升级到协议的最新版来减少被攻击的可能性。当然，SSL 的安全还依赖于服务提供商其他相关的安全机制如网络安全工具防火墙和 IDS 等的配合，尽可能避免提供 SSL 服务的服务器遭到攻击。当然，如果服务提供商支持，你还可以选择增强的认证机制，如双因子认证机制和一次性口令鉴别机制等，或采用更为安全的远程访问协议，如 IPSec VPN 等。

25.3.4　服务锁定风险

除上述三个方面的安全问题是用户必须关注的，还有一个问题是用户在使用 PaaS 之前必须得考虑的，那就是 PaaS 服务锁定风险问题。我们知道，PaaS 服务不同于提供纯粹资源服务的 IaaS，PaaS 服务虽然本质上也是一种资源共享的服务模式，但其提供的资源不再是简单的计算或存储资源，而是与用户基于此上开发应用密切相关的组件服务资源。在很多情况下，用户开发的应用离开这些环境无法正常运行。因此在用户决定是否使用 PaaS 服务前第一件要做的事便是对 PaaS 提供商的锁定风险评估。

但很不幸的是，目前对 PaaS API 的设计，还没有一个统一可用的标准，不同服务提供商提供的 API 大多仅限于自己的平台。如 Google 应用引擎只支持 Python 和 Java，Salesforce.com 的 Force.com 只支持一种称为 Apex 的专有语言（Apex 与其他诸如 C++ 、Java 和 .NET 等语言不兼容。与这些语言不同的是，Apex 的范围更窄，并只能在 Force.com 的平台上构建业务应用程序）。从目前看来，PaaS 服务提供商也不会很快积极推动开发一个跨越云计算的通用 API，因此跨越 PaaS 平台的应用程序的移植将会相当艰难，所以用户在使用某个服务提供商提供的 PaaS 服务前需要对其服务环境进行评估，评估将来业务需要移植的概率和风险有多大，方才确定是否使用该 PaaS 服务业务，否则盲目进入后将会使自己变得很被动。

总结 ：目前业界尚没有 PaaS 安全管理标准，不同的 PaaS 服务提供商都有其独特的安全模式及安全功能。开发者还需要熟悉特定平台的安全特性——比如为了在应用程序中配置验证和授权控制，可以通过安全对象及 Web 服务的形式使用这些安全特性。当然在一般情况下，PaaS 服务提供商不愿意分享有关平台安全参数的信息，因为这些安全信息可能被黑客利用，因此企业用户应当明确要求 PaaS 服务提供商提供一系列的安全功能，包括用户认证、单点登录（SSO ）、授权（权限管理）以及远程安全管理支持等，并搜寻必要的信息以实施风险评估和维护安全管理。最后 PaaS 服务用户还应充分理解应用程序对于各种服务的依赖，并评估第三方服务提供商的风险，这样有助于构建安全可靠的应用程序。

25.4　PaaS服务用户安全检查清单

在 PaaS 服务模式中，用户使用 CP 提供的应用开发环境及支持特定的诸如 Java、Python 和 .net 等开发语言的中间件能力来部署自己的应用，并可控制自己的应用，但对基础设施、服务器、操作系统及存储等并没有控制权限。

针对公共 PaaS 服务提供商，用户应该重点关注的安全清单包括 ：

- 正确评估云服务提供商，检查他们是否拥有行业认证，如 SAS 70 Type II。对行业认证保持适当跟踪以评估可能出现的新的想法与认证并始终保持处于云计算领域的发展前沿 ；
- 研究可以采用哪些实践防止云服务租户间的安全泄露。你应该得到任何租户系统之间的数据都不会遭到泄露的保证。这种保证必须与详细的技术资料一同加以保存备份，这些技术资料说明了采用何种控制措施来防止跨租户的数据泄漏 ；
- 审查服务提供商的服务历史，获得客户的参考信息，并询问他们之前对服务提供商在隐私、可靠性和安全漏洞处理方面的感受 ；
- 确保你有适当的监测和日志记录，配合其他的检测措施，可以验证服务提供商是否达到承诺的服务安全级别，并可让你知道是谁在试图访问某个文件、是谁打开并读取了某个文件等 ；
- 确保应用程序和基础设施的安全要求写进你与 PaaS 供应商的合同中，包括审计条款，这样你或第三方可以定期验证所需的控制措施是否到位 ；
- 获得可靠的服务级别协议（SLA）。SLA 要求服务提供商按规定的级别提供系统 / 服务的可靠性 ；
- 不要接受他们对其服务进行事先毫无宣告的修复策略 ；

- 仔细检查服务提供商的数据恢复政策，以明确如果你要解除合同将需要花多长的时间来取回你的数据，以及他们需要多长时间使数据下线 ；
- 确保你和服务提供商的网站之间的数据传输应有足够强的加密标准和密钥管理策略 ；
- 确定你的客户不是安全链中的薄弱一环。如果指定网页浏览器可以用来访问服务，则要高度注意浏览器自身安全和更新问题。如果可能的话，必须让客户首先登录到你网络以获取其在 PaaS 服务提供商网站上的公司信息 ；
- 如果客户可以访问你租用的服务，要确保自己始终拥有对域名和控制域访问的所有权。这样，如果你终止和服务提供商的关系，你将不必重新培训客户怎样使用正确的 URL 找到你 ；
- 确保有渠道或接口允许您设置或自定义应用 / 服务器 / 网络平台的安全性来满足您的业务需求。

25.5　本章总结

作为 IaaS 或 PaaS 服务用户，需要从另外一些角度或层面去关注云服务可能给你带来的一些潜在风险或安全问题。你需要从服务使用的角度去考虑如何保证你基于这些服务设施之上部署的应用是安全的。在 IaaS 服务中你需要关注远程操作系统和基础服务安全风险、远程管理风险及 DNS 服务风险及相应应对措施 ；而在在 PaaS 服务中，你则需要熟悉 PaaS 平台提供的安全相关的 API、应用安全部署相关技能、远程安全访问及服务锁定等风险。

作为 IaaS 和 PaaS 服务用户，在和云服务提供商签订服务合同前，还应重点关注一些安全服务清单，这包括评估其行业认证情况、历史服务情况、服务合同、SLA、监测和日志、数据恢复策略、传输加密策略等等，做好这些，会让你在很大程度上处于主动状态。

其实本章对于 IaaS 和 PaaS 服务提供商而言，仍然有很多可参考的内容，至少可以将本章节客户应知晓安全风险及应对措施告知其服务消费者以帮助其遵照执行。

参考文献 ：

[1] Phil Cox，Top Threats in a PaaS Cloud Service and How to Avoid Them，http://searchcloudcomputing.techtarget.com/tip/Top-threats-in-a-PaaS-cloud-service-and-how-to-avoid-them

第26章　SaaS服务安全

26.1　SaaS服务安全风险

26.1.1　互联网服务安全现状

近年来频繁发生的 WEB 服务被攻击事件，如 salesforce.com 遭受网络钓鱼攻击，亚马逊 EC2 中的僵尸网络，百度 DNS 被劫持等等安全事件，让人不得不对互联网应用的安全性进行重新审视。要知道，外部威胁总是通过系统的内在漏洞或脆弱性获得成功。因此，研究基于互联网的应用漏洞数据将对尽可能减少攻击事件有直接意义，同时对 SaaS 应用的安全机制实施具有重要意义。

Microsoft 在“ SDL 发展报告 ”里根据美国国家漏洞数据库（http://nvd.nist.gov）发布的漏洞披露数据统计表明 [1] ：行业范围内漏洞暴露的长期趋势有以下特点 ：每年有数千个漏洞暴露，其中大多数漏洞的严重性较高，而利用的复杂性较低。此外，大多数漏洞是在应用程序而不是操作系统或 WEB 浏览器中暴露的。这种趋势令人担心，因为就本质而言，它反映了应用程序中存在数千个严重性较高的漏洞暴露，而且其中大都相对容易利用，如图 26 - 1 所示。

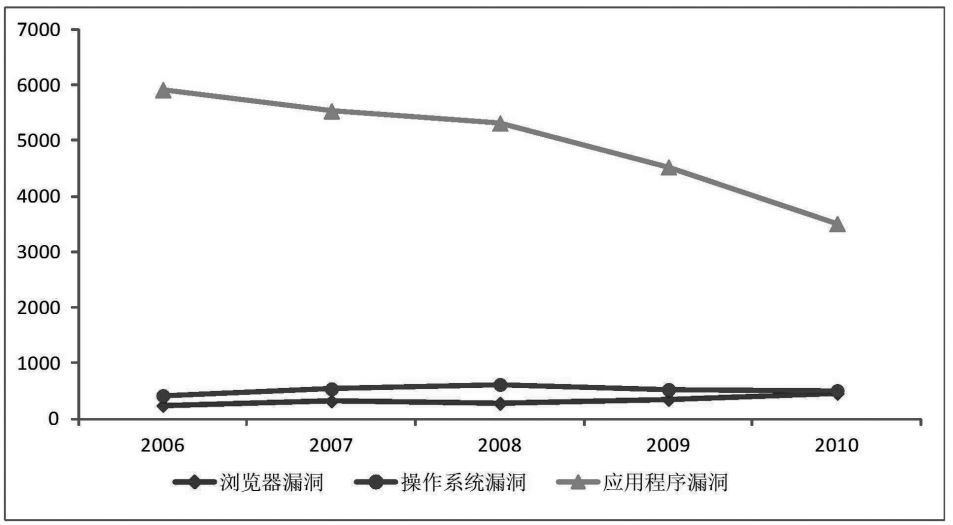


图26-1　行业范围的操作系统、浏览器和应用程序漏洞，2006 － 2010

而来自 Gartner 和 NIST 的数据则更直接地显示出了互联网应用漏洞风险之大 [2] ：Gartner 数据表明有 75% 的攻击发生在应用层，而美国国家标准及技术研究院（NIST）则称在已报导的脆弱性中有 92% 的漏洞是在应用层，而不是在网络层。

26.1.2　SaaS服务安全需求

在讨论 SaaS 服务安全具体问题之前，首先回顾一下互联网服务安全的定义。基于 Wiki 对互联网安全的定义 [3]，互联网服务安全可以定义为 ：保护互联网服务、相关的网络和系统资源及相

关的终端用户 PC 不受到未经授权的修改、破坏、泄漏，并保证服务能够正确地完成其关键功能且不产生其他的有害副作用。简而言之，互联网服务安全是确保互联网服务在有请求时，能够而且只能够被授权的请求安全访问。互联网服务安全包括保密性、完整性、可用性。

SaaS 服务是一种基于互联网的服务，按 wiki 百科定义[4]，SaaS（Software as a Service）也叫软件即服务，它是一种软件交付模式，其本质是应用服务的提供。在 SaaS 服务中，应用服务的软件平台和应用数据位于服务提供商的数据中心，用户通过互联网使用 WEB 浏览器等客户端来访问云中的应用服务。因此，SaaS 服务也具有互联网服务安全的基本特点。

SaaS 服务安全是一个重要而备受关注的话题，对于不同角色、不同业务发展阶段、不同的目的，SaaS 服务安全的意义是不同的。对于 SaaS 服务消费者，根据机构发展阶段不同，对 SaaS 服务安全的需求通常有以下几种情形 ：

- 对于刚起步的公司，在互联网服务安全上，更关注的是“ 有 ”或“ 没有 ”的问题。“ 有 ”是一个商务决定，对 IT 人员来说则是一个技术挑战，即要如何快速找到具有高成本效益的安全解决方案。
- 当公司越来越壮大，对互联网服务安全更多的关注是在提升互联网服务安全的质量，例如高可用性和高扩展性。
- 当公司发展成熟，对互联网服务安全关注的焦点转移到业务流程上，而这种一致的、可重复、可靠的业务流程将不再取决于几个 IT 员工手工作坊式的管理，层次化、体系化的安全技术实施成为必需。

因此，面向不同规模客户的公有云 SaaS 服务，其安全需要在成本效益、可用性、高扩展性、流程化管理等方面做好平衡。而作为 SaaS 服务提供者，SaaS 服务安全应包括以下几个方面 ：

- 数据中心和托管安全

  数据中心和托管安全包括物理安全、系统和资源接入安全。物理安全则指数据中心和机柜访问安全等。系统、资源接入安全则包括网络设备（如交换机、路由器、防火墙、IDS、负载均衡、调制解调器、WiFi 访问节点等）、服务器、存储（包括磁带、CD/DVD 等）的访问安全。
- 终端用户访问安全

  这牵涉到 cookies、令牌、客户端 PC 的数据缓存、敏感信息的输入和传输、Ajax 等。
- 应用安全
  - 应用安全包括第三方应用（例如数据库服务器，WEB 服务器，中间件应用服务器等）和自建的应用。
  - 关于数据中心托管安全及第三方服务中间件安全，已经在前两章中进行了描述，本章将重点阐述 CP 自建应用的安全和终端用户访问安全。

从广义上讲，SaaS 应用从进入云服务环境到下线，要经历构建、部署、使用、下线等过程，在不同阶段，应采用不同的安全技术实施，这些技术实施包括软件安全开发生命周期（SDLC）管理、软件加固、WEB 防火墙、身份管理与访问控制以及终端安全访问等。不同安全机制侧重于对抗不同的安全风险，图 26-2 给出了安全风险、安全保护机制及服务所处阶段间的关系。

需要说明一下的是，软件加固技术可看成是对 SDLC 的补救性技术手段，用于修补或更新 SDLC 遗留下来的可能会对软件安全运行造成潜在风险的安全漏洞，其实施与具体的软件密切相

关，在此不列为一个章节详细阐述，其他安全保护机制和 SaaS 服务风险将在后续章节分别阐述。

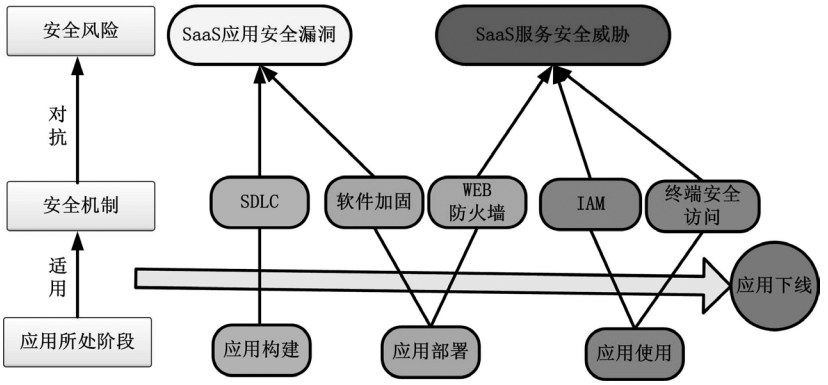


图26-2　SaaS服务安全风险、安全机制及服务生命期之间的关系

26.2　SaaS应用安全保护机制

26.2.1　安全开发生命周期（SDLC）

1. SaaS 应用构建引入安全开发生命周期管理的必要性

从上一节关于互联网服务的安全漏洞统计数据中，可以看出，互联网服务中应用程序的漏洞数量远远高于操作系统和浏览器自身的漏洞。这一现状表明，互联网服务安全在很大程度上取决于应用本身。

由前面 SaaS 服务定义可以看出，SaaS 应用是 SaaS 服务的核心要素。SaaS 应用具有从构建（开发）、交付、运行、使用、下线等完整的生命周期过程。在这个过程中，应用构建（即软件开发）过程中的安全对应用在整个生命周期过程上的安全具有决定性的影响，因为应用构建阶段的安全架构设计、代码实现与代码测试决定了软件存在的脆弱点（漏洞）的多少和严重程度。而这些漏洞一旦被攻击者利用，则可能破坏该软件或该软件所处理的数据的完整性、可用性或保密性，甚至危及互联网服务基础架构（如网络、计算和存储资源等）。

在任何大型应用软件构建过程中，要完全杜绝漏洞的产生是不可能的。只要人类编写软件代码，就会产生导致软件缺陷的错误。某些缺陷（错误）只会妨碍软件完全按预期方式发挥作用，但另一些错误可能会带来漏洞。并非所有漏洞都有同等效果，某些漏洞并不是可利用的，因为特定的漏洞修补技术可防止攻击者利用这些漏洞 ；然而，对于特定软件，其中必然有一些漏洞可能会被利用。因此，要构建一个安全的大型应用需要遵循一定的准则或流程来尽可能地减少这种漏洞的数量。

在云服务环境下，SaaS 应用是一个面向多用户的大型应用软件，构建安全的 SaaS 应用需要一套完整的安全开发保证流程。

安全开发生命周期（SDLC）即提供了一套应用构建安全保证过程，其重点在软件开发过程的所有阶段中引入了安全和隐私原则。

最后需要说明一下的是，通常而言，SaaS 服务提供商提供的 SaaS 应用往往由第三方 SaaS 应用提供商提供，在这种情况下，本节中阐述的 SaaS 应用安全机制应由 SaaS 应用提供商负责实施。CP 则应负责应用安全构建过程的监督、检查和约束。

2. 基于微软的软件安全开发生命周期过程

微软早在 2002 年就提出 SDLC，按 WIKI 基于微软 SDLC 的定义[5]，SDLC 是指一套软件开发过程，微软提出此方法以用于减少软件维护成本，并增加与软件安全 BUG 相关的软件可靠性。

该方法基于经典的螺旋式软件开发模型。图 26-3 是微软软件安全生命周期实践定义。

图26–3　微软应用软件安全开发生命周期实践定义

在微软定义的软件开发生命周期中，包括六个重要阶段 ：

（1）需求阶段

安全系统开发的一个基本原则就是需要“ 自下而上 ”地考虑安全问题，新的发行版或版本的需求阶段和初始规划是构建安全软件提供关键时机。

在需求阶段中，产品小组与中央安全小组联系，请求指派安全顾问（Microsoft 称为“ 安全员 ”），该安全顾问在进行规划时充当联络点，并提供资源和指导。安全顾问通过审核计划、提出建议和确保安全小组规划适当的资源来支持产品小组的日程，为产品小组提供协助。安全顾问在安全里程碑和出口标准方面向产品小组提出建议，这些安全里程碑和出口标准是由项目的规模、复杂程度和风险决定的。从项目开始到完成最终安全审核和软件发布，安全顾问始终充当产品小组与安全小组之间的联系点。

而在需求阶段中，每个产品小组都应将安全性要求视为此阶段的重要组成部分。产品小组应考虑如何在开发流程中集成安全性，找出关键的安全性对象，以及在尽量提升软件安全性的同时尽量减少对计划和日程的影响。在此过程中，产品小组需要考虑如何使软件的安全功能和保证措施与其他可能与该软件配合使用的软件相互集成。

（2）设计阶段

设计阶段确定软件的总体需求和结构。从安全性的角度来看，设计阶段的关键要素包括 ：

- 定义安全体系结构和设计指导原则 ；
- 记录软件攻击面的要素 ；
- 对威胁进行建模 ；

· 定义补充性交付标准。

（3）实施阶段

在实施阶段，产品小组对软件进行编码、测试和集成。在此阶段将采取措施消除安全缺陷或防止引入安全缺陷的作用很大，这些措施将大大减少安全漏洞遗留到软件最终发布版中的可能性。

威胁建模的成果为实施阶段提供特别重要的指导。开发者应特别注意确保代码的正确性以消除高优先级威胁，测试者可集中对这些威胁进行测试以确保将其拦截或消除。在实施阶段中应用的 SDL 要素为 ：

· 应用编码和测试标准 ；
· 应用包括模糊化工具在内的安全测试工具 ；
· 应用静态分析代码扫描工具 ；
· 进行代码审核。

（4）验证阶段

验证阶段是指软件已具备所有功能并进入用户试用版测试的阶段。在此阶段中，在对软件进行试用版测试时，产品小组进行 “ 安全推进 ”，包括进行安全代码审核（超出实施阶段中进行的审核范围）和集中式安全测试（动态二进制程序分析和模糊测试等）。

特别要注意的是对高优先级代码（成为软件 “ 攻击面 ” 一部分的代码）进行代码审核和测试在 SDLC 其他阶段也十分关键。例如，在实施阶段需要进行这类审核和测试，可以尽早矫正问题，并确定和矫正这类问题的来源。在验证阶段由于产品已接近完成，进行这类审核和测试也十分重要。

（5）发布阶段

在发布阶段中，应对软件进行最终安全审核（FSR）。FSR 的目标是要回答下面这个问题。“ 从安全的角度看，此软件是否已准备好交付给客户？ ” 一般在软件完成之前 2 到 6 个月进行 FSR，具体时间根据软件的规模决定。在进行 FSR 之前，软件必须已处于稳定状态，且只剩一些很小的非安全性更改需要在发布前完成。

FSR 是由组织的中央安全小组对软件进行的独立审核。在进行 FSR 之前，来自安全小组的安全顾问向产品小组建议软件所需进行 FSR 的范围，并为产品小组提供资源需求列表。产品小组为安全小组提供完成 FSR 所需的资源和信息。FSR 开始时，产品小组需要填写一份问卷并与派来进行 FSR 的安全小组成员进行面谈。所有 FSR 将要求对最初标识为安全漏洞，但后来经过深入分析确定为对安全性没有影响的缺陷进行审核，以确保分析的正确性。FSR 还包括审核软件是否能抵御最新报告的影响类似软件的漏洞。 对主软件版本进行 FSR 时需要进行渗透测试，可能还需要利用外面的安全审核承包商来协助安全小组。

FSR 不是简单的通过 / 失败测试，FSR 的目标也不是找出软件中所有剩余漏洞，因为这显然不太可行。实际上，FSR 是为产品小组和组织的高层管理人员提供 ：软件的安全水平以及软件发布给用户后抵御攻击的能力的总体状况。如果 FSR 发现某类剩余漏洞，正确的反应是不仅要修复发现的漏洞，还要回到之前的阶段并采取其他针对性的措施解决根本原因（例如，提高培训质量和改进工具）。

（6）支持和服务阶段

尽管在开发过程中应用了 SDLC，最先进的开发方法还是无法保证发布的软件完全没有漏洞，而且有充分的理由证明永远都做不到。即使开发流程可以在交付之前从软件中消除所有漏洞，还是可能会发现新的攻击方式，这样过去“安全”的软件也就不再安全。因此，产品小组必须准备好对交付给用户的软件中新发现的漏洞作出响应。

响应过程包括评估漏洞报告并在适当的时候发布安全建议和更新。响应过程还包括对已报告的漏洞进行事后检查以及采取必要的措施。对漏洞采取的措施范围很广：从为孤立的错误发布更新到更新代码扫描工具以重新对主要的子系统进行代码审核。响应阶段的目标是从错误中吸取教训，并使用漏洞报告中提供的信息帮助在软件投入使用前检测和消除深层漏洞，以免这些漏洞给用户带来危害。响应过程还有助于产品小组和安全小组对流程进行改造，以免将来犯类似错误。

关于基于微软 SDL 更详细的说明，请参考附录相关资料 [6]。

除微软提出的 SDL 外，其他一些大型独立开发商也都已经开始关注软件开发过程中的安全质量问题，例如 IBM 针对自己的实践也提出了软件安全开发生命周期定义 [2]，如图 26-4 所示，并发布用于渗透测试、源代码安全扫描及代码质量分析相应产品 Rational AppScan、Ounce Security Analyst、Rational Software Analyzer、AppScan 以及 HP 的 WebInspect 等。

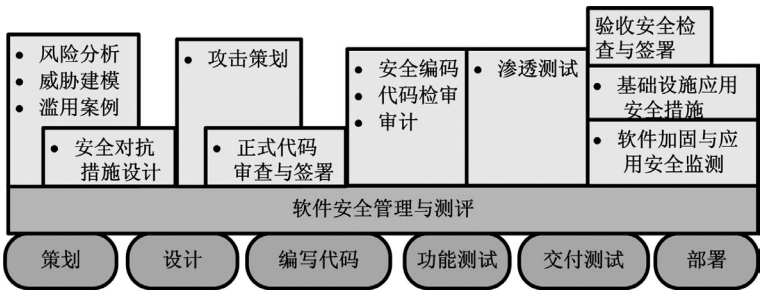


图26-4 IBM软件安全开发生命周期定义

3. SaaS 应用开发生命周期管理实践建议

构建更安全的软件需要从三个方面着手：安全可重复的流程、开发团队培训教育以及软件安全度量标准制定。SDL 的最佳实践是人员、过程、技术、方法的有机统一。因此，即使有大量工具可用于自动化地检测分析和强化 WEB 应用程序的安全，但是，如果在应用软件的设计、实施、测试过程中没有一个训练有素的开发团队和合理的安全度量标准，那么，任何工具都不会真正有效。因此建议：

（1）在软件的整个开发周期，加强开发团队的培训教育，增强整个团队安全开发意识

软件安全开发是风险管理和软件开发的有机结合，需要管理层和开发团队通力合作才能有效降低软件安全风险。

培训教育不仅对于开发人员很重要，它对于应用程序开发所涉及到的全体人员都很有意义，这些人员还包括审计人员和公司决策层。培训教育不仅可以让开发工程师、测试人员、文档管理人员掌握最新影响软件安全性方面的知识，也让他们懂得严格遵循 SDLC 过程的重要性并贯彻于

实践 ；教育更重要的意义是让审计人员和公司决策层理解到不安全的应用可能给企业或客户带来的潜在损失有多大，从而形成从上而下的软件安全开发整体意识，让应用程序所涉及的每一角色充分真正能发挥各自作用 ：公司决策层制定软件需要达到的安全需求总体目标 ；安全团队制定安全策略实现公司安全需求总体目标，并细化具体的实现和控制流程 ；开发团队在公司安全策略的实施和控制下开发安全的质量合格的软件。

（2）明确并制订每一阶段软件开发安全度量标准，并在软件开发每一阶段严格执行检查或测试

“ 您无法管理无法测量的事物。” 一个没有安全度量标准的 SDLC 有等于无。

尽管设计一套能够可靠测量软件安全性的度量标准十分困难，但还是有一些明显可视为软件安全性代表特征的度量标准。这些度量标准包括安全小组人员具备的安全知识（在开发生命周期的开始阶段）以及在已向用户发布的软件中发现的漏洞数。

此外，一些大型的独立软件开发商已经制定出软件安全度量标准，如 Microsoft 已设计出一套从威胁建模到代码审核和安全测试，再到提交 FSR 安全性的度量标准，软件开发人员可参照其制定自己软件安全开发标准，用于产品人员监控其 SDLC 的实施情况。

26.2.2　WEB防火墙（WAF）

1. SaaS 环境中部署 WAF 的重要性

WAF 是 WEB 应用防火墙（Web Application Firewall）的缩写。随着互联网相关技术的飞速发展，基于 B/S 架构的 WEB 应用凭借其良好的交互性和易用性逐渐取代 C/S 架构成为互联网主流应用模式。这种架构可以提供的应用包括审计、协作、客户关系管理（CRM）、企业资源规划（ERP）、发票系统、人力资源管理（HRM），内容管理（CM）及服务台管理等应用。

由于 WEB 应用程序的开发周期较短，开发框架抽象层次高且技术可选性强，因此也造成了基于 WEB 模式的 SaaS 应用开发技术进入门槛较低。如果一个 SaaS 应用提供商在开发中没有能很好地遵循 SDLC 流程，就会造成只重应用功能和外观而忽视应用的安全性等非质量属性要求，这样的 SaaS 应用安全漏洞就会很多 ；另一方面，即使在 SaaS 应用构建过程中严格遵循了 SDLC 流程进行开发，还仍然可能会有一些漏洞存在，这些漏洞就有可能被攻击。据 OWASP 组织（即开放式 WEB 应用程序安全项目）统计，WEB 应用的漏洞已有上百种之多。此外，由于 WEB 应用程序是一种开放性架构，标准性高，而访问用户分寓分散，这都加大了 SaaS 服务被攻击的概率。因此，在不改变应用自身情况下，降低基于 WEB 的 SaaS 应用在生产环境下由于漏洞引发的安全风险已成为保障 SaaS 应用在整个生命周期中安全的重要一环，也是发现 SaaS 应用软件自身缺陷并进行修补的过程。

WEB 应用防火墙即为解决基于 WEB 的应用安全而提出。按 OWASP 组织对 WAF 的定义 [7] 和描述 [8] ：WAF 是 WEB 应用级安全解决方案，从技术的角度看该解决方案不依赖于应用本身。WAF 可以是对 HTTP 会话实施一套规则的设备、服务器插件或过滤器等，通常，这套规则覆盖了常见的攻击，如跨站脚本（XSS）及 SQL 注入等攻击。

此外，WAF 还可提供更多的安全特性，包括安全交付能力、基于缓存的应用加速、木马检查、抗 DDOS 攻击、符合 PCIDSS 的防泄密要求等。

2. WAF 技术简介

WAF 核心技术包括异常检测技术、增强的输入验证技术、应用漏洞修复技术、基于规则的保护和基于异常的安全保护技术、用户访问行为状态识别与管理技术、响应监视和信息泄露保护技术等。

WAF 技术提供了一整套 WEB 应用安全实时防护解决方案。WAF 可以对安全事件发生的事前、事中、事后三个时间点提供防护，这也是与仅对事中攻击进行防护的 IPS 的最大不同点。如图 26-5 所示。WAF 通过一个从主动预防、到攻击防护再到破坏实施控制一个完整过程的安全防护能力，通过将事先预发现、事中可能疏漏的攻击以及事后的弥补，结合起来，可形成一个动态闭环的安全防护机制。即事前用扫描方式主动检查网站并把结果形成新的防护规则增加到事中的防护策略中，而事后的防篡改可以保证即使疏漏也让攻击的步伐止于此，不能进一步修改和损坏网站文件，对于要信誉高和完整性的用户来说，这是尤为重要的环节。

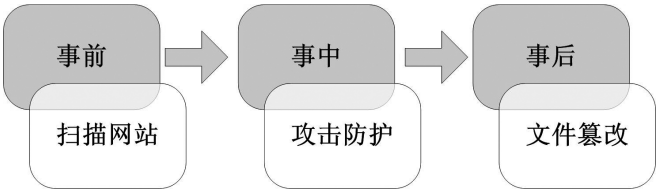


图26-5　WAF安全防护时间轴

随着 WAF 市场需求的激增，目前一些应用厂家和安全厂家都在进入这个市场，WAF 市场上产品形态和功能也差别较大。目前市场上 WAF 产品种类较多，有开源的，也有商业化的产品，但大多为国外厂家所提供。为便于机构选择 WAF，OWASP 组织给出了 WAF 17 个选择标准 [9] ：

- 能够对抗 OWASP 上排名前十的威胁 ；
- 几乎没有误报率（即绝不能禁止授权的请求）；
- 具有较强的默认（开箱）防护能力 ；
- 具有强大且易用的自学习模式 ；
- 可以防御的脆弱性类型多 ；
- 能够对外向响应的消息如信用卡和社会安全码（SSN）检测泄密与非授权内容 ；
- 能够支持积极的和消极的安全模型 ；
- 具有简洁直观的用户界面 ；
- 支持集群模式 ；
- 具备高性能（ms 级时延）；
- 具备完整的告警、取证、报告能力 ；
- 支持 WEB Service/XML 格式 ；
- 能够防止暴力破解 ；
- 能够支持积极（阻止与日志）、消极（仅日志）及旁路工作模式 ；
- 能够限制单个用户只看到当前会话所能看到的内容 ；
- 支持为预防任何任何特定问题（如紧急补丁）的配置能力 ；

·　外形 ：软件与硬件（通常硬件首选）。

从实施部署的角度看，WAF 分为分布式 WEB 应用防火墙（dWAF）和基于云的 WEB 应用防火墙（cWAF）。分布式防火墙是基于纯软件架构的独立部件，可部署于网络的不同区域，适合于大型分布式虚拟化架构，如私有云、公有云和异构云模型 ；基于云的 WEB 应用防火墙是基于平台的，不需要对主机上的软或硬件进行任何改动，只修改 DNS 解析指向。这是一种中央集中模式，威胁检测信息可多租户共享，因此检测率更高，误报率更低，该方式适用于基于云的 WEB 应用或不希望对原有系统软硬件进行修改的机构。

3. WAF 实践建议

无论你采用什么样的 WAF，下面都是部署 WAF 及其在生产环境中运行时应该关注的最佳实践建议 ：

（1）WAF 的部署设计是关键

部署设计不好可能不仅影响了网络整体性能、限制了 WAF 的功能，而且对 WEB 应用也难以取得令人满意的防护效果。

在部署前，需要了解以下事项 ：如果使用软件 WAF，应该考虑 WAF 产品支持的 OS 和硬件 ；如果使用硬件 WAF，首先要确认它能支持的网络拓扑工作模式（如路由器、代理及网桥等），还要清楚它怎么处理 SSL 数据流，如是直接终结 SSL 连接、还只是被动解密数据流或干脆什么都不做。此外，还要确认 WAF 能支持的用户或客户身份认证方法。

在部署时，WAF 必需要与现有环境中的 WEB 基础设施整合，并且要注意到由于 WAF 引入可能给基础设施带来的一些变化。部署 WAF 时尤其要注意的是要尽可能地不改变现有安全策略，一个典型的例子是 SSL 终结 ：如使用硬件 WAF 时则会终结在 WEB 服务器之前，但这种方式通常不会被采用，尤其在高安全要求的应用中 ；可以通过使用基于插件的 WAF 产品直接部署在 WEB 服务器上来保持 SSL 终结还是由 WEB 服务器完成。

在集中式基础设施环境里，WAF 可作为集中式基础设施部件或硬件设备部署，但这种模式下难以预测对整体环境带来的变化 ；然而，对于非集中部署模式（这种部署模式增长很快）的基础设施（如在线商店），使用分布式 WAF 也即插件式的 WAF 直接部署在 WEB 服务器上可能会更合适。从基础设施部署方面考虑，这种部署模式更灵活，它结合了分布式实施与中央管理点两种方法的优点。

此外，对于直接部署在 WEB 服务器的前面的 WAF 产品，由于是串行接入，不仅在硬件性能上要求高，而且要不能影响 WEB 服务，因此，WAF 产品必须具备 HA 和 bypass 等功能，而且还要与负载均衡、WEB 缓存等 WEB 服务器之前的常见产品协调部署。

最后，需要注意的是，基于虚拟化的硬化基础架构成为未来开发越来越重要的基础架构，因此，也要考虑 WAF 能被无缝集成到虚拟化架构里。

（2）关注 WAF 技术实现细节

关注 WAF 的连接处理方式 ：不同的 WAF 在过滤流量的方式不同，如有的是复位 TCP 连接，有的丢弃流量，而有的则脱去可疑的内容，而另一些 WAF 则会结合使用上述技术。你要确定哪

种模式更适合你。

关注 WAF 的流量处理 ：每个网站由一系列独特的应用、协议、数据和动态内容来支撑，检查一下 WAF 能够监视的元语言、编码类型和网站发布的非 HTTP 应用数据流（或收到的递交数据），检查 WAF 过滤的协议、URL 及与标准不一致的 cookies 等。许多 WAF 能够实施对象级或参数级的验证策略。看看你的 WAF 能提供什么级别的 WAF 过滤策略，以及这个策略能否支持基于用户、基于源 IP 地址或时间实施。

关注 WAF 的检测技术 ：WAF 在检测和过滤各种攻击或可疑对象的方式也不同，通常都是用基于签名的方式过滤已知攻击流量的模式。你应该向供应商了解其采用的标准技术和提供的签名数据库、数据库如何更新、是否有客户可用的 API 以及是否支持对其检测功能的扩展等。

关注 WAF 的保护技术 ：在各种各样的攻击与威胁面前，网站是很脆弱的。WAF 供应商通常会在其 WAF 产品里包含特定的对抗措施以过滤一定类型的攻击。要向你的 WAF 供应商了解其采用了什么保护措施来对抗基于 cookie、暴力破解、会话和 DDOS 的攻击，以及其他产品支持的用于消除或减轻风险的措施。

关注 WAF 技术性能，选择的 WAF 基础架构要能支持现有 WEB 基础架构中主要的关键性能参数。不要光看那些标着支持 GB 级硬件吞吐量的字面参数，通常写着的这些数值很难在实际中兑现。更重要的是那些典型的 WEB 关键性能参数，诸如应用的并发用户数以及在此基础上的平均或峰值时的 HTTP 请求数等。需要提醒注意的是许多应用只在极少数时间，如对在线商店来说可能只是在圣诞节才会出现高负荷情况。

最后不要忘记检测一些必要却是基础的功能和事项，如日志报告（本地和远程的、可以支持的格式）、事件通知、报告发送方式、高可用性、安全管理等。当然，还要考虑 WAF 供应商的声誉和技术支持、管理工具的复杂性和可用性、性能、可扩展性、初始成本及服务续订成本等。

（3）WAF 部署之后的维护管理工作也很重要

在一次性的部署工作完成后，后续 WAF 的成功使用实质上取决于 WAF 与其他所有应用基础设施的无缝整合。这包括两个显而易见的话题，如对 WAF 发出的错误及告警信息的理解和响应，以及随着被保护的应用变化而修订 WAF 规则集。例如，为了充分利用 WAF 作为一个中央安全会话管理服务点的能力，要求与应用的开发团队要进行积极的协作。换句话说，为了完全利用 WAF 的能力，仅把它当成一个基础部件是不够的。

最后需要强调的是，无论是否部署 WAF，都不要忽视应用软件的自身漏洞修补与版本升级的重要性。即使采用了 WAF 的主动扫描分析，或是其他漏洞主动扫描工具，还要尽可能早地发现 SaaS 应用漏洞并进行在修复或更新，这是非常重要的。另外，也没有一种安全设施的实施能解决所有安全问题，基于网络的防火墙、基于网络的 IPS、WAF、基于主机的防火墙以及基于主机的防病毒产品均侧重于不同的安全威胁和应用场景，在 SaaS 应用环境下，应尽可能地全面部署。

26.2.3　身份识别与访问管理（IAM）

1. 云服务环境下实施 IAM 的意义

身份识别与访问管理（IAM，Identification and Access Management）是一套建立和维护数字身份，并提供有效地、安全地访问 IT 资源的业务流程和管理手段，利用 IAM 可以实现组织信息资产统一的身份认证、授权和身份数据集中管理与审计。通俗地讲 ：IAM 是让合适的自然人或系统（统称访问实体）在恰当的时间通过统一的方式访问授权的信息资产，提供集中式的数字身份管理、认证、授权、审计的模式和平台。从传统意义上来看，机构实施 IAM 可以提高运营效率，并满足法规、隐私和数据保护等方面的需求。

在云服务环境下实施 IAM 具有更多意义，它除了提高运行效率和合规性管理效率，IAM 还可以实现新的 IT 交付和部署模式（作为一种云应用服务）。例如，身份联合，作为 IAM 的关键组成部分，可以实现跨信任边界的身份信息连接和携带。因此，IAM 使机构和云服务提供商通过 WEB 单点登录及联合的用户开通，在安全信任域间建立通道。

（1）云服务的用户需要 IAM 保护云中自有应用

在云服务发展初期至中期，云服务市场格局尚未形成垄断型格局，企业在将一部分应用移入到云中（如 Amazon EC2）的同时还会保留一部分应用在企业内部（可能某些关键应用永远也无法放在云环境中），同时还会在不同 CP 那里订阅 SaaS 服务（Salesforce.com CRM 和 Google Docs），这将形成企业应用混和模式，图 26-6 给出了云服务环境下企业应用混和格式，在此格局下可能包括企业内部应用、基于 IaaS 和 PaaS 部署的专有应用、SaaS 应用等。

在云服务环境下，企业的网络、系统和应用程序的边界将延伸到 CP 所在的 IT 服务域内（对于大多数从事电子商务、供应链管理、外包和与合作伙伴及社团协作的大公司来说，这早已是事实）。另外，在目前情况下，云服务环境对用户而言透明度极其有限，用户对于服务的部署方式和位置以及它的控制方式，可能只知一二甚至一无所知。云服务可以由多个供应商的众多服务的“混搭”（mash-up）组成，可以是在不同地理位置的数据中心进行物理托管的。这种信任边界的模糊及服务非透明化的模式使得企业用户几乎完全丧失了对 IT 控制的能力。这种控制权的丢失，对企业已有的信任模型和控制模式（包括对于员工和承包商的可信来源）形成了很大的挑战。若没有得到妥善解决，将阻碍企业对云服务的使用。

为了弥补网络控制权限的丢失并加强云环境下应用安全风险管理，企业将不得不根据数据价值（或数据安全等级）对迁移到云服务环境下的应用和数据采取更高级别的安全控制措施，即用户认证和访问控制（IAM），这些安全控制表现为强认证、基于角色或声明的授权、身份联合、单点登录（SSO）、用户行为监测以及审计等。

同时对于订阅的可以产生高价值数据的 SaaS 服务，如 CRM 应用等，企业应该根据数据价值（或数据安全等级）向 CP 确认已经实施了严密的安全控制的措施，其稳固程度至少应该与企业的安全水平相当。

对于使用私有云的企业，这种架构通常是专用 IT 基础机构基础上发展而成，因此，仍然面向单一租户（即企业）专用，这种架构的应用保护同样要求有类似企业传统架构的安全访问控制措施。

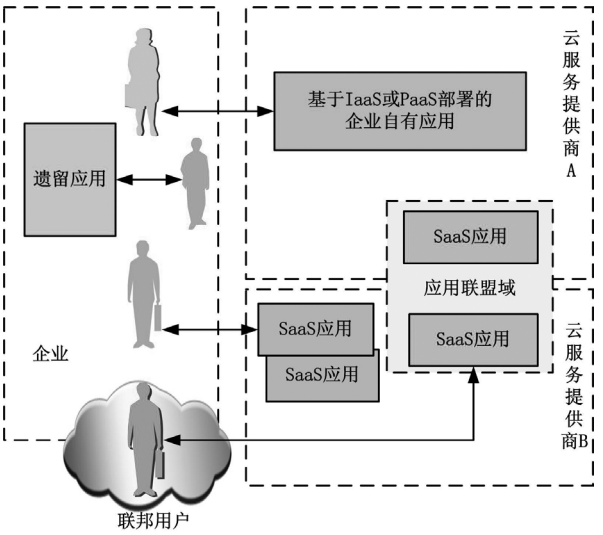


图26-6　云服务环境下企业应用混搭格局

（2）云服务运营商需要 IAM 保护 SaaS 应用安全

CP 不仅要关注于管理不断发展的虚拟化环境，还要关注于他们规模庞大而涉及范围广泛的客户群及其用户可以访问的云基础架构和客户数据。而无论是合法还是欺骗形式的后门应用的访问会使云中的开发环境、生产、存储、数据库、管理与报告等子系统曝露于众多有意或无意的攻击者面前，并会危及到云环境中的路由器和防火墙这样的物理基础架构。

公有云服务环境下，云中应用及数据安全是用户关注重中之重，也是影响用户是否采用 SaaS 应用的重要因素。公有云为多个租户提供服务，用户群体（客户、机构和管理人员）分散在全球各地，因此，更需要强调数据及应用安全，以便保存与保护诸如用户与数据的隐私、知识产权、销售预测和财务与健康记录等数据，并限制特权用户访问敏感数据。CP 通过采用高强度的应用访问控制技术为 SaaS 服务及数据提供安全使用环境，打造良好的服务信誉。特别地，通过提供身份联合架构和流程，可以加强 CP 之间以及企业与 CP 之间的控制和信任。

此外，随着云服务规范的逐步形成，对公有云服务的合规性检查与审计要求将越来具有可操作性，同时也要求 CP 的安全控制措施越来越透明，越来越规范化，这些都要求 CP 在云服务的访问控制技术实施上越来越严密。

另一方面，CP 如果在实践中已经形成强有效的 IAM 服务，同样可以作为 SaaS 服务对外提供。

最后需要强调的是，IAM 是一条双行道。CP 需要支持主流开放的 IAM 标准（如 SAML）和实践，这不仅可提升用户使用云服务的感知，也会加速传统 IT 应用程序从可信公司网络向可信云服务模式的迁移；对于用户而言，良好实施的用户 IAM 实践和流程将有助于保护存储于云服务中的数据的保密性、完整性及管理合规性。

2. 云服务环境下 IAM 应用场景

一直以来，IAM 技术和产品受到拥有众多应用和用户的大型企业与政府机构的关注，他们一般利用 IAM 为其内部大量应用提供大型、专用 IT 基础支撑架构。因为在这种大型异构环境中要

统一管理用户的身份，并赋予用户访问应用的权限是一个很大的挑战，通过使用这种专用的 IT 支撑架构，可以使他们更专注于业务系统自身的设计与实现。在云服务环境下，这类企业与机构或者是私有云的使用主体，或者成为 CP，或 CP 的合作伙伴，因此他们必须考虑他们现有的 IAM 基础设施服务如何能快速扩展到云服务环境，为自己的业务或其他群体如小型机构、CP 和政府机构中的公有云应用提供 IT 基础设施服务。

这类机构将面临需要提供 IAM 支持的场景会有如下几种 ：

- IT 管理员访问服务控制台，为使用企业身份的用户提供资源和访问能力，如 Newco.com 的 IT 管理员在亚马逊弹性计算云中提供虚拟机及虚拟机管理，并在其中配置了虚拟机操作（如开始、停止、挂起和删除等）的身份、权利和证书 ；
- 开发人员在 PaaS 平台为其合作伙伴用户创建账户，例如，Newco.com 开发人员在 Force.com 中为签约的 Partnerco.com 员工提供账户，而后者执行 Newco.com 的业务流程 ；
- 终端用户使用访问策略管理功能在域内及域外访问云服务中的存储服务（如亚马逊 S2 储存服务）并与用户分享文件和对象 ；
- 机构的员工及相关承包商使用联邦身份联合来访问 SaaS 服务，例如，销售和支持人员使用企业身份和凭证访问 Salesforce.com ；
- CP 内的应用程序（如亚马逊弹性计算云）通过其他云计算服务（如 Mosso）访问存储。

此外，一些小型机构也面临着如何构建适用于云环境中企业自有应用安全管控的挑战，因为一方面关键业务应用仍然是（在某段时间内将依旧是）在企业内部部署，IT 依旧需要控制用户对这些应用的访问。另一方面他们的一部分应用需要从一个传统的受控环境迁移到一个超出其 IT 信任边界的云环境中去，其机构内现有的 IAM 产品在云环境下是否仍然在安全管理方面起着至关重要的作用 ；他们可能还要面临着如何从以前的以 Microsoft 以 Active - Directory 为中心的身份认证环境迁移到一个其 IT 服务来自于多样异构的环境中去的问题。这些问题对于一个已在内部部署（on - premise）IAM 且进行大量投资的大型企业而言，是一个迫在眉睫的问题。

而对一些具备 IAM 实践经验的服务供应商来说，云服务环境的到来将带给他们一些令人激动的机遇，他们可以基于已有的 IAM 技术积累和实践经验，在云环境下提供基于云服务基础设施提供公有的 IAM 服务。

综上所述，可以看出，IAM 在云服务环境下 IAM 的应用场景应分为三大类，如图 26 - 7 所示 ：

- 企业自建 IAM ：企业将一部分传统应用利用 IaaS 或 PaaS 服务迁移至云环境，为保护他们的应用，他们需要构建支持其云中自有应用的 IAM ；
- 云服务内部 IAM ：为保护多租户环境下 SaaS 应用安全，并提供 CP 提供的多个 SaaS 应用的单点登录和多个 CP 间的应用联邦身份访问，CP 或第三方合作伙伴构建的 IAM 服务 ；
- 云环境下的公用 IAM 服务 ：由第三方服务提供商（ISP）基于 IaaS 或 PaaS 云环境构建的 IAM 服务，是一种 SaaS 服务。

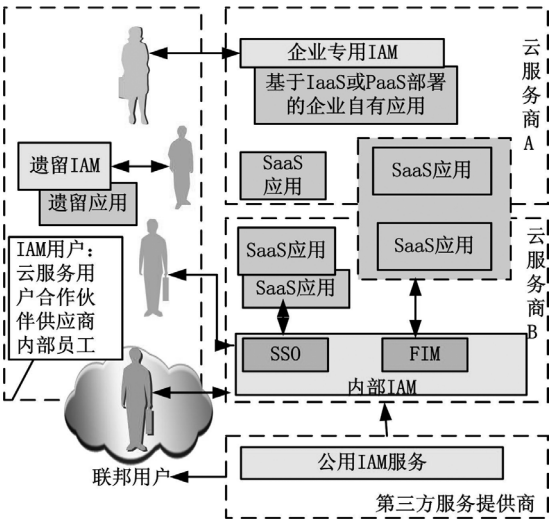


图26-7　云服务环境下IAM应用场景

3. 云服务环境下 IAM 体系架构

尽管如上所述，云服务为 IAM 引入了一套新的应用场景，但这些都不是对 IAM 的革新，而是对其延伸和发展，面临的业务问题仍然是相似的，即如何为云中的应用和数据服务提供安全便捷的身份认证和访问权限控制。一个标准的提供云中应用和数据服务的 IAM 仍是一套由多种技术组件、过程和服务实践有机结合组成的完整体系架构，其包括的主要过程有 ：用户管理、认证管理、授权管理、访问管理、数据管理和提供、监控和审计等，如图 26-8 所示。这种为 CP 的云应用和数据服务提供 IAM 的架构通常还要支持诸如业务开通和取消、证书及属性管理、授权策略、合规管理、身份联合管理和集中化的认证和授权等业务活动。

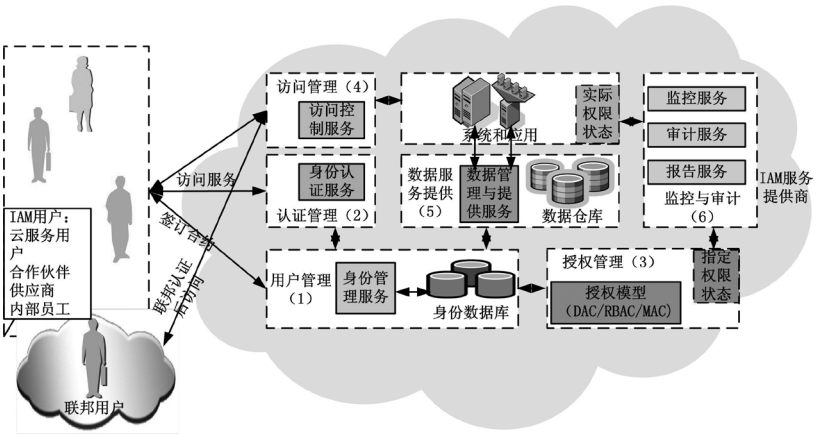


图26-8　云服务环境下IAM体系架构

IAM 部署体系架构的核心是目录服务（例如轻量级目录访问协议或活动目录），目录服务是机构用户群的身份、证书和用户属性的信息库。目录与 IAM 技术组件进行交互，这些组件包括认证、用户管理、在机构内提供并支持标准 IAM 实践和进程的身份联合服务等。由于特殊计算环境的缘故，机构通常会使用多个目录（例如 Windows 系统使用活动目录而 UNIX 系统使用轻量级目

录访问协议）。

在实践上，云服务供应商已经开始向独立软件开发商 ISV 寻求更成熟与广泛的 IAM 基础产品，在其基础上构建分散的 IAM SaaS 服务或与其他服务混搭（mashup）的 IAM 服务。这些服务将具有很高的广度与灵活性，将为云服务环境下的用户和企业带来更多创新性体验。如 CA Technologies 公司提供的 IAM 产品可以为云服务供应商提供身份认证与访问管理功能，实现公有云及私有云下的 IAM 解决方案，包括身份认证联合、记录管理、用户管理、WEB 访问管理与配置在内的云服务安全，可以更好地适用于管理大型、分层、多租户的云应用。

26.2.4　终端用户安全

WEB 应用模式下，浏览器成为终端用户访问云服务的主要方式，如 Microsoft IE、Mozilla Firefox、Google Chrome、360chrome。但不幸的是，几乎所有的互联网浏览器都毫无例外地存在着软件漏洞 ；而浏览器的脚本和插件功能更成为众为攻击者向用户 PC 注入其恶意程序的重要手段。浏览器软件漏洞的存在及其体系架构设计模式都加大了终端用户被攻击的风险，而一旦攻击成功，获取用户访问 SaaS 服务的帐号密码等敏感信息后，不仅使得客户云上数据及隐私受到损害，还会对云中的 SaaS 应用程序的安全造成威胁。在 SaaS 应用程序安全设计中将终端用户安全纳入到其安全考虑范围之内是十分必要也是非常重要的。

然而，对于 SaaS 服务提供者来说，要评估和控制终端用户 PC 机的安全性是非常困难的。根据业务需求，可考虑在应用程序设计中实施以下几个方面的安全机制来加强终端用户 PC 机的安全。

1. WEB 会话管理和安全

HTTP/HTTPS 是个无状态协议，这意味着 WEB 服务器无法通过用户侧的连续请求来维持一个完整的状态。

为克服这一缺陷，往往需要在 WEB 服务器或有时在 WEB 应用里采用各种会话管理机制。WEB 会话管理的基本想法是服务器与用户交互的初期生成一个会话令牌，将令牌发送到用户浏览器并确保这个令牌和后续请求能被浏览器同时送回。这样，会话令牌就变成用户识别令牌，服务器可以使用令牌来保持会话数据（如变量）并为用户创建类似会话的历史记录。

通常，会话令牌不只是识别标记，也是身份验证。这意味着，登录时，用户通过凭据（用户名、密码、数字证书）获得认证，并且获得一个会话令牌，该令牌作为访问会话的临时静态密码。

在 WEB 环境中，广为采用的维持会话的方式有三种 ：URL 参数，隐藏的表单域和 cookies。每一种方式都有其优势和不足，cookies 是三者中最方便的、也是最安全的一种方式。

WEB 会话的安全性侧重于防止三种针对会话 ID 的攻击 ：截获、预测和暴力攻击。

会话安全管理的最佳实践是 ：

- 用保密性强的算法为验证的用户生成一个唯一的会话 ID，这点很重要。理想情况下，会话 ID 应该是一个随机值。
- 会话 ID 的长度要足够长，这样攻击者在有限的时间里就无法通过暴力破解获取有效的 ID。鉴于当前的处理器和带宽的限制，建议使用 50 个以上的随机字符来构成会话 ID。

- 如果会话持续时间很长，应该使用 https 协议。
- 如果会话保存在 cookies 中，应该给会话加密并设置一个较短的有效期。
- 在大多数金融 SaaS 的网站，使用多因素认证来加密、解密一个特殊的初始会话 ID。这个特殊的会话 ID 将作为用户和其所用 PC 机的指纹。基于这个指纹，结合常规用户认证才可建立一个持久的常规会话 ID。
- 有时，指纹也和一个嵌入的表单域一同使用。
- 设备的指纹也可以通过其他机制创建，比如客户端 SSL 证书（也叫双向 SSL）、智能卡或 / 和安全狗等。
- 基于一些特殊的安全模型，设备指纹还可用于监控和判定设备的安全性，设备的安全性也称作设备信誉。通过判定其安全性，可以调整相应的数据流。

2. 客户端数据缓存

有时，为了加速 WEB 服务性能，应用程序、浏览器将一些表格、图像缓存在终端用户的 PC 机上。一些站点也可以使用缓存的 cookies 作为用户和 PC 的指纹。

为了保护基于互联网的 SaaS 服务安全，在进行客户端数据缓存可采用以下方式 ：

- 将缓存的敏感信息加密。
- 对缓存数据采用安全域沙箱技术（domain sandbox）。
- 如果业务需要，强制执行刷新以确保应用 / 服务从服务器上获取实时的时间。不要让浏览器替您做决定。

3. 跨站点脚本

虽然跨站点脚本通常给我们总带来一些负面印象（如攻击），可在有些情况下它却是满足我们业务需求的唯一解决方案。如果你需要把它作为你唯一的解决方案，你最好需要先考虑好以下几个问题 ：

- 安全性较高的 Iframer 框架和 /Ajax 技术是否可提供类似的解决方案？
- 你怎么正确评估这个方案的潜在安全性？
- 有人可以使用你的代码来劫持会话以向其他站点发起攻击吗？
- 有人可以使用你的代码向你的站点或其他站点注入恶意代码吗？
- 你是否会在 XSS 代码中泄漏任何敏感信息？

4. 记住个人 PC/ 设备信誉

大多数服务提供商都可以为 PC 专用的用户提供终端用户选项，用户可选择记住该电脑上的一些个人访问信息。大多数时候，在用户专用的 PC 上保存与 Cookie、指纹相关的个人信息是比较安全的。

这些选项可以通过方式来增强 Cookie 安全性 ：

- 几个只有终端用户知道的预先定义答案的问题 ；
- 多因素认证（MFA）：通过手机短信或电话发送一个令牌 ；
- 一封带令牌的验证 / 确认电子邮件（带令牌）；
- 设备信誉 ；
- 带有个人指纹的 cookie 等。

如有更严格的安全需求，服务提供商还可以实施如下基于业务需求和资源的解决方案 ：

- 使用带有加密密钥的硬件加密狗 ；
- 使用硬件安全令牌作为一次性密码 ；
- 使用带钥匙或指纹的 USB 盘 ；
- 发行专用客户端协议互惠协议 ；
- 为交互式 SSL 发行专用的客户端 SSL。

对于使用 CS 来说，首先要保证访问 SaaS 服务的 PC 或其他设备的安全。这不仅包括在终端上部署安全防护软件，包括防病毒、木马及恶意程序检测软件、个人防火墙，更重要的是要及时给浏览器和操作系统打补丁和更新版本，并谨慎使用浏览器插件和脚本功能。

此外，CS 需要增强安全防范意识，这包括提高对网络钓鱼的识别能力和一些可疑插件或脚本的运行警惕性，更重要的是，不要在可能泄露个人敏感信息（如帐号、密码、身份证、证书密钥等）的地方放松访问 SaaS 服务。要记住，SaaS 服务的安全是云服务提供商和云服务用户双方共同的责任。

需要强调一下的是，由于 SaaS 服务客户通常只承担操作层面的安全实施职责，所以选择 SaaS 服务提供商需要特别慎重。目前对于 SaaS 服务提供商评估通常的做法是根据保密协议，要求提供商提供有关安全实践的信息，该信息应包括设计、架构、开发、黑盒与白盒应用程序安全测试和发布管理及生产环境中的安全措施等。

26.3　案例研究：桌面云服务安全部署方案

26.3.1　桌面云服务概述

桌面云是 SaaS 服务的一种，它允许用户通过瘦客户端或者其他任何与网络相连的设备来远程访问跨平台的应用程序，以及整个客户桌面。服务提供者在数据中心服务器上运行用户所需的操作系统和应用软件，然后采用桌面交付协议将操作系统桌面视图以图像的方式传送到用户端设备上。同时，服务器将对用户端的输入进行处理，并随时更新桌面视图的内容。用户只需要一个瘦客户端设备，通过专用程序或者浏览器，可以访问驻留在服务器端的个人桌面以及各种应用，用户体验和使用个人电脑几乎一样。

26.3.2　设计挑战

用户由传统的桌面终端迁移至云桌面服务模式时，其在接入方式、应用访问模型等方面都发生了较大的改变，因此，在安全方面云桌面除存在与传统桌面终端相同的一些安全问题，如面临用户身份、访问控制、终端管理、桌面安全、数据保护和防泄露以及终端安全接入控制等问题外，还存在与传统桌面终端不同的特点，这些特点表现在 ：

- 用户的访问模型由二层（终端→应用）变为三层（客户端→桌面云→应用），这意味着，身份管理和访问控制由终端延伸至客户端、桌面云 ；
- 用户数据由分散（终端）到集中（云中），既带来集中数据的安全保护问题，但由于云桌面的集中可控，又使得对客户端的外设管理、恶意代码防护、补丁管理和桌面行为审计

等得到大大的简化 ；

· 云桌面在传统的操作系统之下引入了新的虚拟化层，带来新的安全风险。

26.3.3　设计要点

1. 总体架构

从总体上来讲，云桌面的安全性既与传统的桌面终端有共同之处，又有别于传统的桌面终端安全 ；既有有利于安全的地方，也引入了新的威胁和脆弱点。基于此分析，并根据云桌面部署架构，在实际设计中从两个维度上考虑云桌面安全设计架构，如图 26-9 所示，以实现对云桌面服务的分级防护和协同保障。

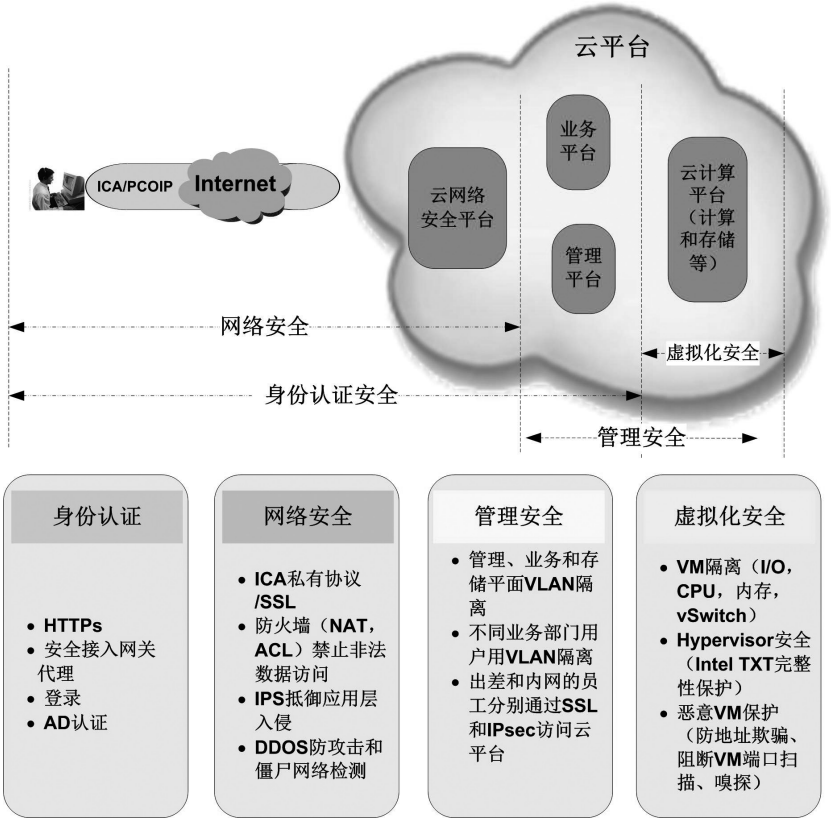


图26-9桌面云安全设计架构

- 分级防护 ：根据用户接入客户端类型区分不同的接入区域，同时根据用户处理信息的敏感程度区分不同的云桌面分区，在这些区域内执行一致的安全策略、并对区域间的信息传输进行控制 ；
- 协同保障 ：云桌面安全主要关注接入端点的信任管理和访问控制，以及虚拟机的安全性，这些都是云桌面引入的新的安全问题，而传统桌面安全手段仍然可以在云桌面和用户接入的传统 PC/ 笔记本电脑等客户端上发挥应有的作用，二者相互补充，共同建立起全面覆盖的多重防御保障体系。

对传统的终端安全措施，已有成熟的解决方案，且各企业也大都已选择并部署了适合自身需要的终端安全解决方案，例如基于 802.1x 的终端接入控制和包含桌面补丁管理、恶意代码防护、上网行为控制、行为审计、系统身份认证与访问控制等功能的桌面安全管理系统，只不过在云桌面环境下这些都是在云中以集中的方式进行，这些都仍然可以完整地应用于云桌面架构下的安全保障，而在 WEB 应用防护以及云平台安全治理方面的实施与其他云服务应用没有特别的区别。因此，以下仅重点描述针对云桌面环境下接入端点的信任管理和访问控制、虚拟机的安全性以及数据安全防护的概要设计 [10]。

2. 设计概要

（1）端点接入安全

如前所述，由传统桌面环境到云桌面环境，访问模型也由二层变为三层，即接入客户端 -> 桌面云 -> 应用，因此设计的核心思想是根据不同的应用场景设置不同的访问控制策略 ：

- 根据不同的接入客户端类型（手机瘦客户端、PC 胖客户端等）设置不同的接入客户端到桌面云之间的访问控制策略 ；
- 根据不同的桌面云类型（永久型和随机型）设置不同的桌面云到应用之间的访问控制策略。

（2）虚拟机安全

基于 VDI（虚拟化桌面基础架构）技术的云桌面实现架构，因其较高的安全隔离性、性能隔离性和平台兼容性成为目前云桌面的主流实现模式。在这种架构中，虚拟机是其核心要素，因此，虚拟机的安全也成为桌面云要重点考虑的一个安全问题。

虚拟机的安全保障，按照其特性可分为服务器虚拟化软件自身提供的安全特性和外部虚拟化安全应用提供的安全特性两大类。以下是基于服务器虚拟化软件自身应提供的安全设计概要 ：

- 通过内存隔离、CPU 隔离、网络隔离、IO 隔离等技术使同一物理机上的不同机之间相互隔离，互不影响 ；
- 虚拟机（VM）无法访问虚拟化实现层（Hypervisor）；
- 在同一物理机内部网络中，虚拟交换机（vSwitch）要支持虚拟局域网（VLAN）功能，同一物理主机的不同 VM 可通过 VLAN 进行隔离 ；
- 采用 Intel TXT（Trusted Execution Technology）技术，在启动时对 Hypervisor 和控制域内核进行完整性校验，确保系统完整性，避免恶意程序入侵 ；
- 防地址欺骗，限制 VM 只能发送本机地址的报文 ；
- 支持对 VM 端口扫描、嗅探等行为的检测和阻断。

外部附加的虚拟安全应用，应包括常见的防火墙、IPS 功能，以及恶意代码防护、完整性监控和日志分析等功能。其部署既可按照传统的桌面终端防护方式给每个云桌面安装防护软件，也可安装一个实例在物理服务器上，通过服务器虚拟化软件提供的安全 API 接口对该物理服务器上的所有云桌面进行防护，两者所提供的功能是一致的。后者不仅可对在线云桌面进行防护，也可对离线的云桌面存储文件进行扫描和防护，既提升了管理效率，又极大地提升了桌面云环境整体的安全性。

（3）数据安全防护

数据防泄漏和数据备份设计概要 ：

- 自动发现和监控从接入客户端、桌面云到存储数据源的敏感数据，并强制执行适当的数据保护策略 ；
- 与文档权限管理系统集成，基于信息内容、上下文关系和身份对敏感数据进行保护。

集中化的云桌面环境使得需要考虑备份的数据量大增，因而在设计备份方案时需要考虑适当的备份范围、备份方式。

3. 效果评估

在完成上述桌面云服务安全设计方案后，在上线实施前，该方案还要通过方案内审、问卷调查、白盒测试、黑盒测试、用例测试、渗透测试等安全验证评估环节。目前基于该架构的安全方案已部署到某电信大型网管系统和某银行营业厅，并已稳定运行两年多。

26.4　本章总结

SaaS 服务提供用户利用互联网来访问共享应用的能力，用户不再管理或控制底层的云基础设施。因此，CS 也不对 SaaS 服务安全承担主要职责，但作为应用数据的最终拥有者，CS 对 SaaS 服务安全的需求同样存在，甚至更为强烈。且处于不同发展阶段的 CS 对 SaaS 服务安全关注的侧重点有较大区别，初创型公司更关注找到一个高成本效益的安全服务解决方案，而随着公司发展壮大，其更关注层次化和体系化服务安全技术实施。

在 SaaS 服务模式下，CP 管理和维护整套 SaaS 应用，因此 SaaS 服务提供商应最大限度地确保提供给客户的应用程序和组件的安全，这包括网络、虚拟化、操作系统、存储、服务组件及 SaaS 应用等层面的安全。其中网络、虚拟化、操作系统、存储及服务中间件层面的安全已在前面两个章节里描述，本章重点阐述 SaaS 应用层面的安全。SaaS 应用也具有特定的生命周期，这包括应用创建、部署、使用及下线，且每一阶段安全技术实施的侧重点及其对抗的风险有所不同。本章给出了 SaaS 应用在生命周期每一阶段侧重的安全实施机制以及可对抗的安全风险关系图。

在 SaaS 安全风险及需求分析基础上，本章按照 SaaS 应用生命周期发展阶段给出了不同的安全实施机制，应用构建中的软件安全开发生命周期过程管理（SDLC），WEB 应用部署及运行时安全机制（WAF 技术），以及 SaaS 应用运行安全机制身份识别与访问管理（IAM）。

终端用户安全是 CS 和 CP 都需要关注的安全实施机制。云服务提供商应在安全设计实现过程中增强用户端的 Cookie 和会话管理安全 ；用户则主要负责操作层面的安全功能，包括用户接入和访问管理。此外，还需要积极配合，及时更新浏览器补丁和升级版本，并增强用户个人敏感信息保护安全意识等。

无论是作为公有云还是私有云服务，云桌面由于其带来的便捷访问、集中化管理、节能环保等优势，已在国内得到获得众多认可，尤其在窗口服务型或终端操作维护型场景（如营业厅、网络管理等）。作为案例，本章最后给出云桌面应用安全部署实践。案例详细介绍了云桌面服务概

念、服务架构及云桌面服务安全设计总体架构及设计概要等，以供读者在部署云应用服务实践中参考。

参考文献：

[1] Steve Lipner，SDL 发展报告，http://www.microsoft.com/downloads/zh-cn/details.aspx?FamilyID=918179a7-61c9-487a-a2e2-8da73fb9eade

[2] 许舟平，基于渗透测试和源代码扫描的软件安全测试和开发，IBM，2009，http://www.ibm.com/developerworks/cn/rational/r-cn-appscanouncersar/

[3] Internet Security, Wikipedia, http://en.wikipedia.org/wiki/Internet_security

[4] Software as a Service, Wikipedia, http://en.wikipedia.org/wiki/Software_as_a_service

[5] Security Development Lifecycle, Wikipedia, http://en.wikipedia.org/wiki/Security_Development_Lifecycle

[6] Steve Lipner，可信赖计算安全开发生命周期，http://msdn.microsoft.com/zh-cn/library/ms995349.aspx

[7] Application Firewall，Wikipedia, http://en.wikipedia.org/wiki/Application_firewall

[8] OWASP，Best Practices: Use of Web Application Firewall, https://www.owasp.org/index.php/Category:OWASP_Best_Practices:_Use_of_Web_Application_Firewalls

[9] OWASP，Web Application Firewall，https://www.owasp.org/index.php/Web_Application_Firewall

[10] XX 电信桌面云平台项目技术方案建议书（内部资料）

第27章　云计算安全运营治理

27.1　组织架构与过程模型

前面章节主要讲述了云计算安全的技术实施。云计算安全如同传统的信息安全一样，也是技术、流程和管理（人）等多种安全要素的有机结合，本章及后续章节将主要从管理和流程的角度阐述如何治理云计算安全。缺乏有效的安全治理将导致关键业务需求很难被满足，因而也很难保证云计算用户规模和云计算产品的市场竞争力。

27.1.1　组织架构

在不同领域（如医疗、金融等），云计算安全治理方法与控制机制有很大不同。但无论是在哪个领域，在进行云计算安全治理活动前，CP 都需要有一个明晰的云计算安全的高层战略，这包含机构实施云计算安全治理要达到的总体目标及由此产生的总体需求。一个内容清晰而合理的云计算安全高层战略是构建云计算安全治理可持续运营模型的基础。

要制定高层战略，通常还需要成立一个云计算安全委员会。云计算安全委员会提供与机构业务和 IT 战略相一致的云计算安全行动指南，如云计算安全章程。在这个章程里通常会规定云计算安全治理团队以及其他与云计算安全相关的人员角色及职责。这些安全角色可能包括面向内部安全治理和外部客户安全服务的安全角色，如负责基础设施安全的人员、负责数据安全的人员、负责应用安全的人员、面向客户安全服务的人员、负责安全技术统筹实施的人员和负责安全管理的人员等 ；以及与服务安全治理相关的外部安全角色，如外部专家与技术顾问和终端用户等。一些角色可以合并，也可以划分得更细，这取决于云计算规模及机构整体业务战略。

需要强调一下的是，云计算安全委员会不仅要在章程里明确规定安全治理组内部安全角色及其相应职责，还要明确定义外部安全角色的职责，尤其是终端用户，虽然 CP 可能并不能完全控制其职责的实施，但应在云计算合约签订时明确告知终端用户应该负责提供的安全职责，如实施终端设备安全增强机制、采用符合 CP 要求的终端安全访问措施、加强加密密钥及证书等敏感信息保护意识等。

云计算治理要取得一个持续可运营的安全治理效果，除需要云计算安全委员会及安全治理组外，一般还需要一个云计算安全规划组。云计算安全规划组根据云计算安全委员会制定的高层战略，结合云计算安全治理组的实施建议来制定云计算安全实施规划。图 27-1 是云计算安全治理组织架构示例，它包括机构管理层、云计算安全委员会、云计算安全规划组、以及包含内部和外部人员组成的安全治理组。

云计算安全战略与云计算组织架构是云计算安全治理的输入项与基础，基于此，有助于构建一个目标明晰、过程可持续改进的云计算治理模型。

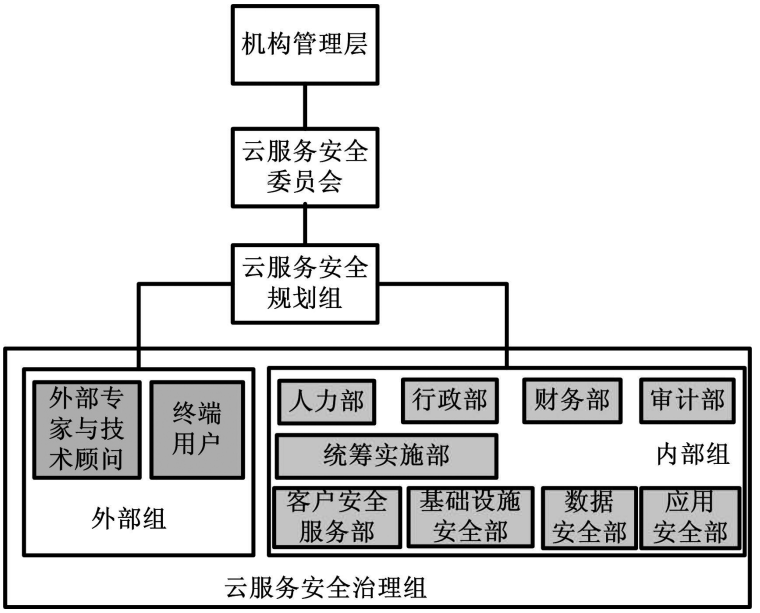


图27–1　云计算安全治理组织架构示例

27.1.2　风险管理

云计算安全治理的过程也是云计算运营安全风险管理的过程。按 ISO 27005_2008（《信息技术、安全技术、信息安全风险管理》）对风险管理的定义，风险管理包括确定范畴、风险评估、风险处置、风险接受、风险沟通、风险监视与评审以及风险管理方法改进等，其中风险评估包括风险识别、估算与评价，风险处置则包括风险降低、风险转移、风险保持与风险避免。风险管理过程通常需要对风险评估和风险处置循环进行以获得在高风险识别与控制措施上的不断改进。

云计算环境下有效的风险识别应包含如下内容 ：

- 信息资产的识别，包括基础设施资产的识别、数据类型及保护级别识别、业务流程识别、应用类型识别、以及数据所有权与监管责任的识别等 ；
- 威胁识别，包括威胁种类识别、威胁来源识别、威胁影响识别及威胁频率识别等 ；
- 脆弱点识别，可在组织架构、治理流程、安全相关人员、云数据中心物理环境、网络及基础资源、系统及服务配置等方面进行识别。

根据不同的云计算交付模型和部署模型，云计算环境下的风险处置（降低、转移、避免、接受）主体不同，尤其对公有云计算，由于涉及利益多方，风险处置前的风险沟通显得更为重要。如对公有云的 IaaS 服务，数据及应用层风险处置则由用户自己选择处理，CP 对基础资源层的风险处理需要与其相关资源的供应商及 CS 进行风险沟通后选择风险处置选项 ；此外，CS 可以通过选择不同的服务交付模式进行风险转移，如选择公有云的 SaaS 应用，则将数据和服务设施等风险转移至 CP。

安全风险评估为机构提供一种平衡安全控制实施与资产保护需求的重要依据。

27.1.3　过程模型

云计算是一种 IT 服务，也应遵循 ISO 27001-2005（《信息技术 安全技术 信息安全管理体系要求》）的 PDCA 过程模型。云计算安全治理从规划到云计算安全治理改进形成闭环流程，云计算安全治理的过程即是云计算安全风险管理的过程。如图 27-2 给出云计算安全治理过程与风险管理关系。

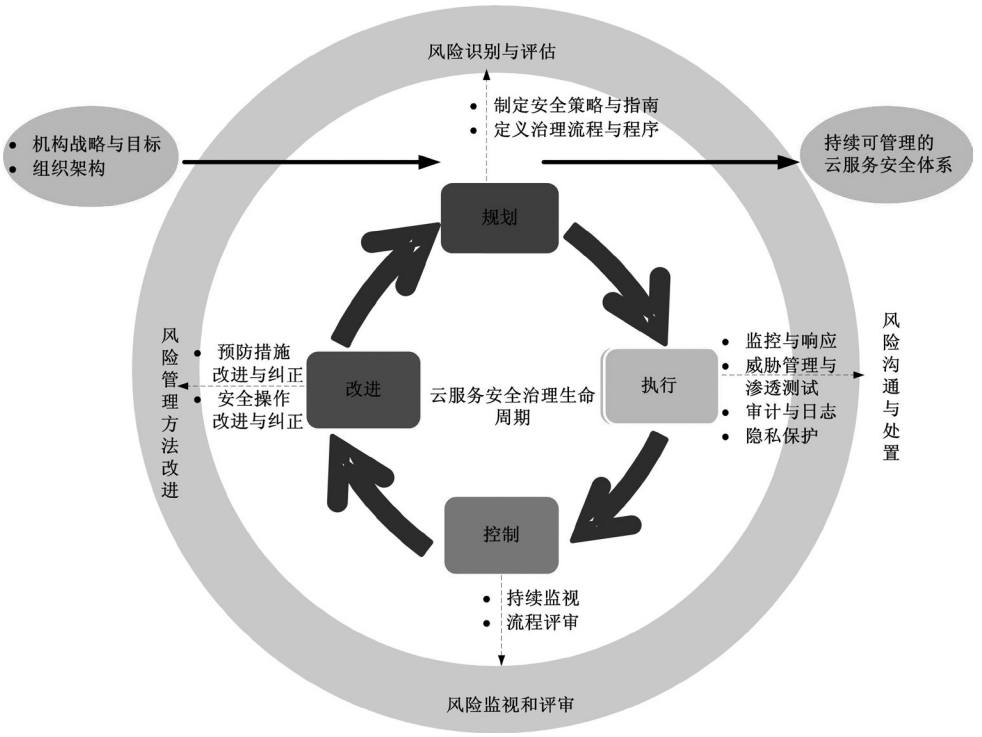


图27–2　云计算安全治理过程模型及与风险管理关系

云计算安全治理过程包括四个阶段 ：

- 规划阶段 ：根据管理机构的云计算安全高层战略指导思想（输入项），由云计算安全治理组和云计算安全规划组共同制定云计算安全治理实施策略、云计算安全治理流程与处理程序，并对安全风险进行评估等 ；
- 执行阶段 ：根据安全策略、治理流程定义以及风险评价清单实施安全操作并进行风险处置，安全操作包括如安全监控与告警、安全扫描与渗透测试、变更管理、安全审计及隐私保护等 ；
- 控制阶段 ：根据安全策略、目标及既有经验评估与测量执行阶段的过程与风险管理效果，并向管理层报告结果，并进行评审和持续监视 ；
- 改进阶段 ：根据内部审计及管理评审结果等信息，给出安全技术实施预防措施及安全治理操作纠正和改进建议，最终实现可持续、可管理的云计算安全治理体系。

控制和改进阶段方法具体实施请参照 ISO 27001-2005 第 7 和第 8 章节描述。本章后续章节主要针对规划阶段的策略制定以及执行阶段安全操作进行描述或给出实施建议。

27.2　云计算安全治理操作

27.2.1　安全策略与指南制定

云计算安全策略与指南是 CP 关于云计算安全治理的白皮书，是满足 CP 云计算业务发展要求和相关法律法规而制定的云计算安全治理的指导文件，是对机构信息安全高层战略及目标的细化与解析。

云计算安全策略与指南是指导机构员工正确操作、使用和管理云计算环境下信息资产，并保护这些资产使其拥有更好的保密性、完整性和可用性的明确指南。云计算安全策略与指南是云计算安全体系防范潜在威胁的重要组成部分，这些策略在察觉并防范社会工程学攻击时尤为有效。云计算安全委员会应根据业务战略和目标制定清晰的方针指导，并通过在整个机构中颁布和维护云计算安全策略来表明对云计算安全的支持和承诺。

建立云计算安全策略和指南的目的如下 ：

- 在 CP 机构内部、合作伙伴中建立一套云计算资产及服务安全相关的通用的、行之有效的安全机制 ；
- 在 CP 内部员工、合作伙伴员工中树立起安全责任感 ；
- 增强云计算信息资产的可用性、完整性和保密性 ；
- 提高内部员工、合作伙伴员工及 CS 的信息安全意识和信息安全知识水平。

云计算安全策略与指南应包含明确的目标、管理意图、职责与权限、相关术语及操作细则定义等，具体涵盖范围如下 ：

- 定义与云计算安全策略和指南相关的术语 ；
- 明确云计算安全策略和指南的目标 ；
- 确定云计算安全策略和指南适用的信息资源及工程资源的范围 ；
- 明确整个云计算安全治理相关组织和人员的角色与职责，以维护云计算安全和报告安全漏洞或事件 ；
- 识别云计算安全风险，并评估可能发生的影响 ；
- 定义变更审批流程和责任 ；
- 定义灾难恢复相关的资源、流程和责任 ；
- 定义一套从开发到运营的云计算安全控制实施标准，包括 ：
  - 物理和逻辑访问控制 ；
  - 事件响应与管理 ；
  - 系统及网络配置备份 ；
  - 周期性安全测试 ；
  - 数据及通信加密（加密算法及密钥长度）；
  - 密码标准 ；
  - 持续性监控。
- 定义确保以下几个方面的安全所需的处理流程和责任 ：
  - 数据中心 ；

- 网络 ；
- 服务器和操作系统 ；
- 虚拟化软件 ；
- 数据库 ；
- 应用中间件 ；
- 代码开发和应用程序等。

· 指定文档的审查和修订流程。

27.2.2　安全监控与事件响应

1. 云计算环境下安全监控意义

在云计算运营环境下，云数据中心规模庞大，应用系统多样，涉及运维及合作厂家人员多而杂，需要有一种机制能实时可视化地记录云数据中心人员及系统的活动行为，并对安全事件提供告警。

安全监控即通过对云计算数据中心的操作人员、网络、主机及应用系统活动行为的监视，识别、记录并统计网络、系统及应用服务的脆弱性、外部攻击与异常行为以及安全违规行为等，通过事件告警及通知机制以便安全管理人员有效地监视、控制和评估网络或主机系统的运行状态，及时对安全事件进行响应，消除安全隐患或降低安全风险。

全方位的持续性监控机制和定义清晰的事件报警流程是治理良好的云计算数据中心的基础。随着云计算基础设施增长及云计算业务的不断扩展，安全监控的重要性将更加明显。在云计算环境下实施安全监控的具体意义有以下几个方面 ：

- 发现事先无法防御的威胁 ：对于一些难以事先阻止或防御的攻击，安全监控则成为最后一道防线。
- 验证安全控制措施的有效性 ：安全控制措施通常是对安全策略执行，其结果反映了安全控制措施实施是否正确。如果在安全事件流里出现一些安全策略禁止的事件，则表明安全控制没有按安全策略正确实施。
- 暴露系统脆弱性 ：通过运行监控可以识别一些未曾发现的系统脆弱性或安全错误。
- 记录安全行为 ：安全监控捕获的安全事件数据合法地记录了用户或执行过程的活动行为。
- 提供电子取证重要数据支持 ：在多租户云计算环境中，安全监控可以为攻击过程提供视频回放，安全事件数据则提供重要信息记录。这对于需要分析攻击过程和识别损失范围的电子取证具有非常重要的价值。

此外，通过采用一些先进的安全监控解决方案，CP 可以把安全监控作为服务对外提供。

2. 安全监控分类

安全监控通常分成两类，一类是物理监控，如视频监控、门禁、火、水及其他环境传感器、设施巡检等。这些活动通常由数据中心安全人员负责 ；一类是电子监控，包括内部系统监控、威胁监控。无论哪种方式，都要有一个定义清楚的流程来确保记录了能满足安全策略要求的日志 ；

内部系统监控通常用于监控所有的服务器的系统补丁和防病毒补丁是否及时更新，及其 CPU

和 RAM 利用率情况等。为提高查询效率，通常要将从这些服务器搜集到的数据存放于一个数据库中（如配置管理数据库）。

云计算环境下的威胁监控首先要从 IDS/IPS 传感器、病毒日志以及各种设备的系统日志搜集事件及报警数据。对于一些中小型数据中心可以通过人工方法收集数据，而对于一些大型数据中心，则需要利用一些工具实现自动化信息收集，这些工具包括威胁关联引擎和各种各样的安全事件管理工具等，如 OSSEC、TripleWire、Splunk、WEBtrends、logAnalysis 等。这些自动化工具能够缩短出现在事件流中的假阳性事件数量，可以识别更复杂的攻击以及对出现故障的传感器进行报警，并可以将报警分组发送并对传感器的数据进行整合关联等。

3. 云计算环境下的安全监控要求

安全监控主要用于收集环境数据、进行网络安全监控及审计日志等。在云计算环境下，安全监控的要求如下 ：

- 安全监控应该是一个高可用的硬件化服务设施，可通过安全方式从内部和远端进行访问 ；
- 安全监控要能对发生或检测到的关键安全事件以自动化方式生成告警 ；
- 关键告警要能通过多种方式上报并及时提醒安全管理人员 ；
- 安全监控审查的日志至少要能涵盖以下几种 ：
  - IDS/IDP 日志。
  - 防火墙日志。
  - 用户帐号日志。
  - 互联网设备日志（路由器、交换机等）。
  - 应用程序日志。
  - WEB 服务器日志。
  - WEB 应用程序防火墙日志。
  - 数据库服务器、备份和恢复日志。
- 根据安全事件分级，安全人员应有多种事件响应方式，如对重要安全事件进行立即调查或处理，或仅简单浏览日志以完善告警机制等 ；
- 安全监控必须确保是可靠且正确的，即使事件生成和报告数据收集出现故障也要保证递交的事件是可靠的，安全日志必须符合法律及安全策略。

此外，还可以考虑一些其他扩展性功能，如允许客户对 PaaS 或 IaaS 实施入侵 / 异常检测，甚至允许他们发送安全事件或告警到 CP 的安全监控系统里。

最后需要提醒的是，由于 SaaS 服务面临的威胁和攻击的类型与传统的基于基础设施和物理环境的威胁及攻击有很大不同，因此，SaaS 服务机构需要扩展其传统的安全监控能力以包含对应用及数据层活动的监控。同时还需要组建一支面向云中应用安全及隐私保护等专业领域的安全团队，以保证客户数据安全并提供应用服务的稳定性。

附 ：安全告警列表

CP 可以通过订阅不同资源的安全列表来丰富自己的安全监控告警信息库。不同资源的列表有 ：

- 供应商的安全报警列表，例如 Microsoft，Oracle，Cisco，Juniper，RedHat，Symantec

等 ；

- US - CERT 提供 4 个可以订阅的邮件列表。您可以选择以下一个或多个文档 ：
  ①技术网络安全警报 ；
  ②网络安全公告 ；
  ③网络安全提示 ；
  ④当前动态。
- www.securityfocus.com 提供许多可以通过邮件订阅的警报信息列表。最常用的有 ：
  ① BugTraq（提供专注于计算机安全话题的电子邮件列表订阅服务）；
  ②聚焦于微软产品的 ；
  ③聚焦于病毒产品的 ；
  ④聚焦于苹果产品的 ；
  ⑤ WEB 应用程序安全 ；
  ⑥ IDS ；
  ⑦渗透测试 ；
  ⑧取证。
- http://www.infosecnews.org

4. 事件响应

安全监控的最终目标是对识别和检测出的安全威胁进行处理，以消除安全隐患或降低安全风险。安全监控和事件响应是两个密切联系、相互依赖的安全域，如果缺少任何一个环节整个安全监控将达不到预期效果。

在对安全事件进行响应之前，通常需要制定一个按安全事件严重等级区分的事件响应计划。首先给出安全事件的等级定义，并用不同的方法标示，如低 / 中 / 高或主要 / 次要等，并且对每个事件规定一个合理的响应。对于低级别安全事件，可以由运营人员作为日常管理活动的一部分进行处理 ；对于第二级别的安全事件，如机架电源故障或影响一段网络的网络故障等，除要修复此问题，还要跟踪事件处理过程，并根据需要决定是否发起一个根本原因分析（RCA）流程以确定原因在哪和是否需要改变策略、基础架构等以制止此类事件再次发生 ；对于影响大量用户或可能引发重大安全危害或影响机构信誉的最高级安全事件，做好响应计划成功响应这类事件的关键。通常情况下，这类事件的响应不仅仅要涉及运营人员，还可能涉及很多人员，包括供应商及合作伙伴等，而且这类事件要求认真对待和专业处理。此外，处理过程中的证据保留很重要，如果采取的处理步骤不当，证据也很容易被毁坏。

27.2.3　威胁管理和渗透测试

大多数恶意软件都是通过网络远程攻击基础设施、部件、网络服务和应用中存在的漏洞，这使得基于网络的云计算面临更大的威胁。无论是对于使用公有云模式的 IaaS 用户而言，还是对于提供 PaaS 和 SaaS 服务的 CP 而言，都需对其部署的操作系统或应用系统的脆弱性、补丁和配置管理承担主要责任，都需要一种主动式的系统或服务的脆弱性发现、修复机制可以用来降低漏洞被攻击的概率。

威胁管理即通过对系统和基础设施中可能存在的脆弱性进行主动性扫描，根据漏洞可能引发

的风险大小对发现的脆弱性进行分类评估，并根据脆弱性类型进行补丁修复，以降低风险或消除安全威胁。从流程上看，威胁管理包括三个阶段 ：安全扫描、脆弱性评估及补丁修复。

CP 和 CS 对云基础设施的威胁管理均负有责任，这取决于 SPI 服务模型。对于 SaaS 服务而言，由 CP 负责基础设施、网络、主机、应用和存储和第三方服务的脆弱性、补丁、配置管理（VPC），SaaS 提供商需要定期评估新的漏洞，并对 SaaS 服务中涉及的所有系统软件和固件进行补丁修复。在这种服务模式下，CS 基本上对威胁管理不承担责任的 ；而在 IaaS 服务中，CS 则需要对其管理的整个软件栈（操作系统、应用及数据库等）的威胁管理负责 ；另外，CS 也需要为其部署在 PaaS 平台上的应用提供威胁管理，但这些都必须是在 CS 不泄漏其他用户隐私且不涉及 CP 商业机密的前提下进行，用户可以获取所需的安全配置信息以及系统运行状态信息，并在一定条件下获得部署专用安全管理软件的权利。

需要说明一下的是，为提高安全治理效果，通常将安全监控、威胁管理（包括配置管理）及变更管理结合在一起，以防止误报、提高事件响应效率、降低安全风险。

1．脆弱性管理

脆弱性管理是保护主机、网络设备和应用、避免已知漏洞攻击的重要威胁管理要素。其通过扫描网络及信息资产中存在的脆弱性，并对其进行分类，以获得更有效地威胁消除或风险降低的措施，如补丁和系统升级。脆弱性评估通常与威胁发现（来自安全监控或安全扫描等）、补丁管理、升级管理过程整合在一起，以便更有效、及时地修复漏洞。

成熟的机构通常有一个制度化的脆弱性管理流程，包括对网络环境下的各个系统的脆弱性进行扫描、评估脆弱性可能给机构带来的风险并进行分级，解决风险问题的补救措施等（这个通常由补丁管理流程来完成）。脆弱性管理的最佳实践是安全运行人员先发布一些代码开发安全指南，并执行安全可接受性测试及相应标准，将脆弱性问题尽量在开发阶段解决。

安全扫描是脆弱性管理的基础，CP 应定期（如一周一次）进行安全扫描以主动检查服务和应用中存在的脆弱性。安全扫描分两类，一类是端口扫描，一类是漏洞扫描。目前这两类扫描都有很多工具可用，最著名也是最常用的端口扫描工具是 Nmap。而流行的漏洞扫描工具有 Nessus（免费、但不开源）、Core Impact 和 QualysGuard 和 ISS 的互联网扫描器等。在大多数情况下，可以使用开源工具做内部安全扫描，而使用商业解决方案以满足审计和合规需要。

端口扫描和漏洞扫描的区别往往模糊不清。端口扫描用来从远程网络位置收集有关测试目标的信息，尤其是找出每个目标主机或系统可提供的网络服务或开放的端口。而漏洞扫描用于扫描主机操作系统上已知的弱点和未打补丁的软件，以及文件访问控制和用户权限管理缺陷等配置问题。因此，端口扫描器和基于网络的漏洞扫描器的基本区别是漏洞扫描器扫描所有系统及应用服务缺陷，包括脆弱性、访问控制缺陷和配置问题等，对于外部的恶意漏洞扫描，还会试图在目标系统上针对这些缺陷执行相应攻击 ；而端口扫描器只是产生可用服务的清单。

此外，设计良好的漏洞扫描器是渗透测试的一个重要工具，它为探测每个目标主机上可用的网络服务提供了必不可少的手段。漏洞扫描器可以扫描数据库记录的网络服务安全缺陷，并在目标范围内的主机服务上测试每个缺陷，这可以快速全面地发现目标系统常见的配置弱点以及未打补丁的网络服务器软件。

2. 补丁管理

与脆弱性管理类似，安全补丁管理也是保护主机、网络服务及应用、避免非授权的用户利用已知漏洞进行攻击的重要威胁管理手段，通过消除内部和外部威胁来降低机构安全风险。

补丁管理流程需要遵从变更管理框架（见 27.2.4 变更管理一节），其输入来自脆弱性管理。

在这种持续运营的云计算环境下进行补丁管理是非常必要也是非常重要的。以下是补丁管理最佳实践建议 ：

- 所有云计算环境中运行的软件或系统所需的补丁，其开发、测试、部署及生产都需要在一个相互隔离的环境中进行 ；
- 在云计算环境下，必须要为网络部件、服务器、存储、虚拟化软件、应用和安全部件的软件或固件补丁管理定义一个明晰的流程 ；
- 必须要为漏洞修补或补偿性控制措施定义一个整合策略，策略包括从重大威胁响应到非关键性补丁处理策略等，以提高云计算的安全性或运营可靠性。

3. 安全配置管理

安全配置管理也是一个重要的威胁管理手段，可以保护主机、网络设备免受非授权用户针对其配置缺陷进行的攻击。安全配置管理与脆弱性管理程序密切相关，是全部 IT 配置管理的一个子集。

保护网络、主机、应用的配置缺陷不受攻击需要进行安全监控，以及对关键系统和数据库配置文件的访问控制，包括操作系统配置及防火墙策略等。

4. 渗透测试

渗透测试（penetration test）并没有一个标准的定义，国外一些安全组织达成共识的通用说法是 ：渗透测试是通过模拟恶意黑客的攻击方法，来评估计算机网络系统安全的一种方法。这个过程包括对系统的脆弱点、技术缺陷或漏洞的主动分析，这个分析是从一个攻击者的角度进行的。

渗透测试还具有两个显著的特点 ：一是渗透测试是一个渐进的并且可逐步深入的过程 ；渗透测试必须要采用不能影响业务系统正常运行的攻击方法来进行测试。

同脆弱性扫描一样，云计算环境下的渗透测试也需要定期进行。渗透测试往往需要较强的专业技巧和专业知识，对云计算环境下基础设施及应用安全测试要求测试人员对虚拟化和云架构有深入理解。如果内部安全团队不具备这些能力，可以外包给有资质和能力的第三方来执行。

渗透测试和脆弱性扫描都可以发现大量漏洞，但与脆弱性测试有所区别的是渗透测试着眼于整个云计算设施而不仅仅是单个的服务器或部件漏洞，因此，渗透测试往往把云计算作为一个黑盒进行测试 ；而脆弱性测试则侧重于单个主机上系统或应用的脆弱性识别及配置缺陷等问题。

渗透测试和脆弱性扫描发现的漏洞并不都必须或能够被修复，这些漏洞可以按关键 / 高 / 中 / 低进行分级。通常对于关键或高级别的漏洞需要进行修复。而对于中或低级别的漏洞，根据不同的云模型及 CP 的业务需求，可以选择接受或进行修复。对于没有被修复的漏洞需要进行残余风险评估。此外，为提高安全修复效率，对于多个相同架构的服务器的脆弱性，可以制作一个黄金镜像来对多个服务器同时进行修复。

27.2.4　变更管理

无论是云服务中的软件还是硬件，在运营过程中，通过运营中的 BUG 管理、需求管理及风险控制等，CPs 都需要定期优化、完善、升级对外提供的服务或用于构建服务的内部功能模块。在新版本部署到生产环境之前，需要在一个尽可能与运营环境相同的环境中进行测试。由于云服务环境是由多个离散的部件组成的，这些部件可能包括运营商级交换机、路由器、目录服务器、安全基础设施、应用服务等，因此，这种优化或升级将是一个非常艰巨的任务，尤其对于公有云而言，更不允许因升级服务或完善基础设施而导致长时间的停电。

云服务变更管理即通过对云服务及其基础设施变更需求及实施流程的管理，尽可能地降低变更实施风险、满足业务需求，同时提高合规性。

虽然变更管理本身并不是云服务安全治理操作内容之一，但一个没有满足安全需求或存在安全漏洞的变更申请将会导致客户数据丢失或服务中断。成功的云服务安全团队通常需要与开发团队紧密合作，并积极跟进正在开发和测试中的产品变更情况。对于一些复杂且重要的产品变更，为提供变更的自服务能力及优化安全团队的时间及资源利用，安全团队也可以创建一个标准且具有最小变更内容的安全指南。

变更管理流程如图 27-3 所示。特别需要说明的是，上文中描述的配置管理及补丁管理在实施流程中也需要遵循变更管理框架。

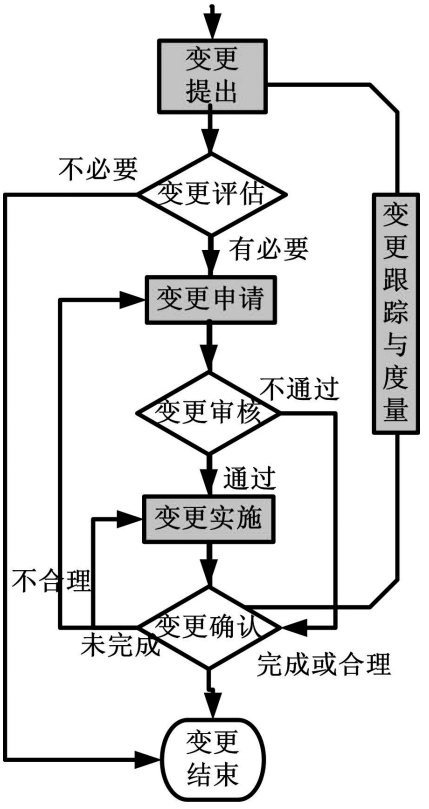


图27-3　变更管理流程

27.2.5　安全审计与日志

云计算是一种构建体系复杂及用户群体多样的生态环境。在这种环境下，即使在相关的基础服务设施或系统内都实施了一定的安全保护机制（如访问控制和加密等），并且也在整个云计算环境中采取了一定的云计算安全治理措施，如安全监控、威胁管理和渗透测试等，但仍然不能保证某些基础设施或应用系统不会遭到破坏或攻击。而一旦采用的这些安全防御体系被突破我们还能做什么？另一方面，作为面向广大用户群体提供服务并受到监管机构监管的云计算提供商，除了上述的事前预防和事中监控的安全措施外，还必须要有一种能够提供预后追查和事后取证的机制，这种机制即是安全审计。安全审计一方面能及时发现预先定义的攻击性行为，另一方面也为法律或监管调查取证提供重要线索。此外，安全审计对于 CP 的业务恢复也具有重要意义，因为如果清楚了系统是怎么遭到攻击的，才有可能更快地恢复系统或服务。从这些意义来讲，云计算环境下的安全审计如同飞机上使用的“黑匣子”。

安全审计是指识别、记录、存储、分析与网络和系统安全行为相关的信息的过程。安全审计是计算机和网络安全的重要组成部分，也是评判一个系统是否真正安全的重要尺码。在云计算环境下，面向系统和网络的安全审计至少要实现以下几个方面的安全目标：

- 跟踪或监测系统中的异常事件；
- 确认并保持系统活动中每个人的职责；
- 确认并重建已发生的安全事件；
- 评估损失；
- 提供有效的灾难恢复依据；
- 提供阻止不正当使用系统行为的依据；
- 提供网络安全案件侦破证据。

ISO/IEC15408 在“信息技术安全性评估通用准则 2.0 版”（又称 CS 准则）中对安全审计定义了一套完整的功能，包括安全审计自动响应、安全审计数据生成、安全审计分析、安全审计浏览、安全审计事件存储、安全审计事件选择等。在云计算环境下，安全审计系统应跟踪所有服务层次产生的与安全相关的事件，包括系统事件、安全事件、网络事件、应用事件以及其他事件等。图 27-4 是安全审计系统框架。

安全审计有两种实施情况：

- 内部审计，是应内部治理需要而做，如为彻底审视云计算安全治理状况，这通常由云计算治理安全组织架构中的审计部门来负责实施；
- 外部审计，通常是由监管部门的合规检查或客户的合规性要求而做，多由外部代理或顾问来实施的。目前，外部审计情况居多。

此外，无论是内部安全审计还是外部安全审计，以下都是进行安全审计的实施步骤建议：

第一步：定义审计的范围和目标

- 创建一个主列表，列出审核应该包括的工作。大多数情况下，如果以前做过同样或类似的工作，都会有一些可以借用的模板或范例。

第二步：差距分析

- 主要是检查目前的状况、分析需要做出多大改进、评估有多大风险和任务的紧急性等等。

第三步 ：实施和部署解决方案

- 我们一旦知道了差距，下一步是针对每一个问题找到解决方案，并按照一定的优先级实施解决方案。

此外，对审计师的资质要求和人员选择也很重要。对于外部审计来说，很多时候组织没有权利去选择审计师或安全评估人员。由于目前阶段很多审计师对云和虚拟化技术还很不是很熟悉，如果机构可以选择的话，强烈建议选择一个对云计算技术和安全都有一定了解的审计师，这可以从询问这些人员关于 IaaS、PaaS、SaaS 相关术语的熟悉程度作为一个评判的起点。

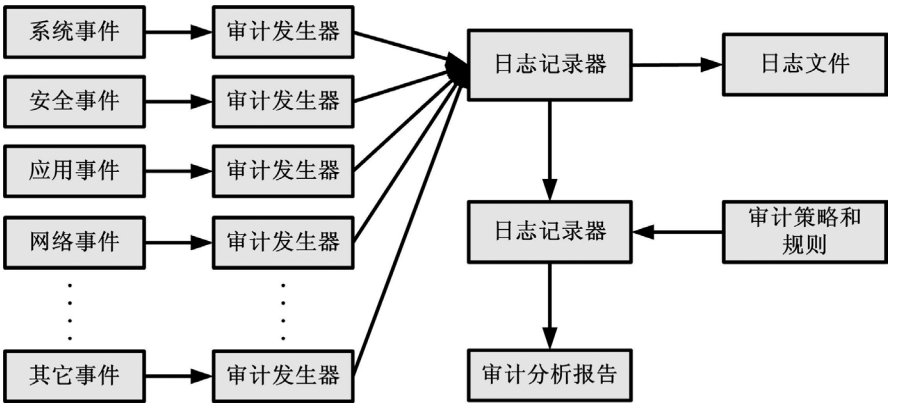


图27–4　安全审计系统框架

27.3　隐私保护

27.3.1　云计算环境下的隐私保护概念

随着互联网的普及与推广，近年来通过互联网来披露、公开或传播他人隐私行为的方式及事件已越来越多，如通过“ 人肉搜索 ”就可直接探测个人隐私信息，将诸如家庭住址、年龄、身份证号等隐私信息轻而易举地挖掘出来。而随着由于互联网经济的快速发展，这种通过网络传播个人隐私信息的事件呈现扩大化、侵权形式也更加多样化、侵权手段更为智能化，危害结果将趋严重化。

在讨论云计算下隐私保护问题之前，先看看什么是隐私。隐私的概念最早由美国法学家沃伦（Samuel D.Warren）和布兰德斯（Louis D.Brandeis）于 1890 年在《哈佛法律评论》发表的一篇文章《论隐私权》中提出，后得到全球业界人士的普遍认同。它是指自然人自身所享有的与公众利益无关的且并不愿他人知悉的私人信息。这是隐私的普遍定义，而对于基于网络的隐私，法学界至今还没有形成统一的看法。有的学者认为网络隐私是指自然人在网络上的个人数据信息、隐私空间以及任何与个人网络活动有关的信息。网络隐私权保护即指禁止他人非法知悉、侵扰、传播或利用的权利。由于网络环境的开放性、虚拟性、交互性、匿名性等特点，使得基于现实环境的一些隐私权保护手段在网络环境中则变得难以实施。

在云计算环境下，只要用户愿意，任何可以存放在本地机器的数据都可以存放在云上，包括邮件、健康档案、财务信息、约会计划、广告策划、商业计划、演示文档、图片、视频等等。在云计算环境下，首先要保护这些存放的数据隐私信息，如不允许第三方在未经同意的情况下存储或读取用户云上存储的邮件信息等 ；其次还要保护发生在云环境下的用户操作行为的隐私性，如用户访问的站点应不允许用于商业目的收集 ；更重要的是，用户个人相关信息不允许共享给与用户云计算使用无关的他人等。因此，概括起来说，云计算环境下的隐私保护最主要目的是防止 CP 恶意泄露或出卖用户隐私信息或机密数据，或者搜集用户数据进行分析以挖掘用户隐私信息，例如，分析用户潜在有效的盈利模式，或者通过两个公司之间的信息交流推断他们之间可能有什么合作等。

在基于网络提供服务的云计算环境而言，由于用户数据控制权的转移及安全边界的消失，用户个人隐私信息及数据隐私保护问题，尤其是服务侧隐私保护问题将变得更加突出，用户对云计算下的个人隐私信息及数据隐私保密性担忧已成为阻碍用户使用云计算的最大障碍之一。按 Ponemon 学院及 TRUSTe 所做的 “ 2008 年隐私保护最受信任的公司调查 ” 结果，隐私保护是如今电子世界里最为关键的市场竞争区格因素。调查认为云计算的消费者愿意和一个他们相信他们可以信任的 CP 建立业务关系。因此，对云中个人隐私及数据安全提供保护无论是对 CP 还是 CS 都具有重要意义。

27.3.2　云计算环境下哪些信息是隐私数据

在云计算环境下，个人及数据隐私保护强调的是个人存放在云上的数据、云计算环境下的用户访问行为（如访问的站点）或双方签约相关信息（如身份证号码等）等不允许非授权访问或非正当原因的收集、使用与披露，最终目的是保护客户个人隐私信息及机密数据，并防御各种欺骗行为，如身份盗窃、垃圾邮件、钓鱼等。为便于阐述云计算环境下的隐私保护对策，先来看一下云计算环境下哪些信息是用户的隐私信息。

云计算环境下的用户隐私信息可以分成两块 ：一块是与用户使用云计算相关的数据，叫用户数据（User Data）；另一块是用于唯一识别、联系、定位个人或与其他信息一同可唯一识别个人的数据，即用户可识别信息（PII），也叫个人数据（Personal Data）。

用户数据是从客户收集的信息，包括 ：

- 任何直接从客户处收集的数据，如通过用户界面用户输入的信息 ；
- 任何间接从客户处获得的信息，如文档中的元数据 ；
- 任何关于用户使用行为的数据，如日志和历史等 ；
- 任何与客户系统相关的数据，如系统配置、IP 地址等。

个人数据可能包括以下几种 ：

- 联系人信息 ：姓名、邮件地址、电话、邮寄地址等 ；
- 身份信息 ：身份证、社交号码（SSN）、驾驶证、护照、医保卡、指纹、营业执照等 ；
- 人口特征 ：年龄、性别、种族、宗教信仰、性取向、犯罪记录等 ；
- 职业信息 ：工作头衔、公司名称、所属行业、从事专业等 ；
- 健康信息 ：健康计划、健康历史、保险情况、遗传信息等 ；

- 财务信息 ：银行存款、信用卡 / 借记卡账号、购买记录、信用卡记录等 ；
- 在线行为 ：IP 地址、Cookies、Flash cookies、登录凭证等。

需要强调的是，对于企业用户而言，上述的用户数据是以一个公司为实体的隐私信息，也属于个人隐私信息保护范畴。

上述用户数据和个人数据，尤其是个人数据，CS 通常并不是在使用云计算情况下提供给 CP 的，如身份证信息和营业执照等信息通常是在签订合约时提供的，或在生成订单时提交给 CP 的。这些信息均与用户使用云计算系统时存储在云上的其他业务数据有很大的不同，需要针对其不同的特点采取不同的保护措施。

27.3.3　云计算环境下隐私数据保护对策

从上述关于用户隐私信息的数据类别来看，用户的隐私信息涉及范围较广，从业务数据到用户输入访问信息，从用户个人特征信息到用户在线行为信息等，因此，对于云计算环境下的用户隐私信息的保护，仅仅依靠单一手段如数据加密是远远不够的，需要有一套完备的隐私保护体系，涉及多个层面，以下仅从法律、技术、监管、约束机制四个层面分析其对策。

1. 完善立法及相关的法规制度

云计算环境下保护隐私信息最好的实践就是要有法律来做保证，许多国家已经颁布相应法律来保护其个人隐私，如加拿大的个人信息保护和电子文档法案（Personal Information Protection and Electronic Documents Act，PIPEDA）、欧盟委员会的数据隐私保护指示 [1]、瑞士联邦数据保护法案（The Swiss Federal Data Protection Act，DPA）、瑞士联邦数据保护法令（the Swiss Federal Data Protection Ordinance）等。在美国，个人隐私权利还要受到行业相关管制法案的保护，如健康保险责任法案（Health Insurance Portability and Accountability Act，HIPAA）、金融服务法案（The Gramm - Leach - Bliley Act，GLBA）和美国联邦通信委员会客户专属网络信息规定（FCS Customer Proprietary Network Information Rules，CPNI）等。本部分的第 29 章云计算的合规性将会对上述部分规约进行介绍。

目前，国内现行法律对用户信息隐私的保护还比较滞后，还无专门针对网络环境下的用户隐私保护的体系化专门的法律或法规，更没有针对云计算环境下的用户隐私信息保护相关法律或法规。要适应云计算环境下的用户隐私信息保护需求，我国乃至全球的立法机关都应加快加强用户隐私信息保护方面的相关立法，明确云计算环境下用户隐私信息的保护内容及流程，建立一套完善的用户隐私信息保护体系。

2. 运用数据安全保护及隐私增强技术

对于存放在云上的用户数据其隐私信息保护主要采用数据加密、安全认证、访问控制、数据屏蔽等技术。需要说明的是，在保护云中存放的数据其所包含的隐私信息方面，数据加密能较好地解决无意或恶意带来的隐私信息泄密问题。此外，双因子增强认证方案为云中数据访问认证提供更好的安全保护。

关于隐私增强技术（Privacy Enhancing Technologies，PET），目前业务还无统一的定义，目前学界认可的说法是，PETs 是指任何可以用来保护或增强个人隐私信息安全的技术，这包括数

据加密技术，智能卡和可信任的令牌环，增强型便携式客户端设备，分布式应用，面向对象的封装（数据与方法的紧耦合），基于 XML 的消息（基于格式与规则的紧耦合），用于分布式数据与处理的 WEB 服务架构，更细粒度级的断言、隔离认证、授权与属性以及零知识证明技术等 [2]。

3. 设立政府监管机构

除了法律对隐私信息保护违规行为的制裁和增加安全技术实施外，云计算服务下的用户隐私信息保护更多时候依赖于相关机构的有效监管。设立以政府为主导的可信的第三方监管机构不仅可以为云计算环境下的用户隐私信息保护提供公平、可靠而权威的监督与管理，还可定期对云计算提供商的服务可用性、安全技术实施与安全治理进行审计、评估等。

此外，监管机构还可以在 CS 与 CP 双方间的 SLA 履行过程中起到很好的监督与制约作用。服务等级协议（SLA，Service Level Agreement）是明确云计算相关各方权利与责任的合约，一方面可以较好地保证云计算的质量 ；另一方面，SLA 也确定了双方的经济关系与赔偿机制。监管机构应该根据 SLA 的要求，对云计算涉及的多方进行监督，保证用户和服务提供商的权利和利益。一旦出现违反 SLA 的行为，监管机构应该协助权益受到损害的一方要求侵权实体提供合法的赔偿。

4. 建立云计算环境下用户隐私信息使用约束机制与流程

由于监管审计或法律查证等需要，仍然存在需要将涉及云用户隐私信息的数据提供给第三方的可能。在这种情况下，无论是宏观环境或安全技术实施对用户隐私信息提供多大保护，最终的用户隐私信息保护还需要云计算提供商与用户间有一个严格而规范的用户隐私信息使用约束机制与流程来保证，这可能包括 ：

- 用户隐私信息收集前的正式声明 ：根据监管或法律等要求需要收集、使用、持有、分布、传输用户隐私信息的公司或提供商必须对数据拥有者提供一个清晰而明确的正式声明，声明中必须如实说明收集用途、期限、执行实体等信息 ；
- 隐私信息拥有者的确认 ：隐私信息拥有者必须提供一个清楚而正式的确认函，用于表明同意收集、使用、持有、披露、传输和保护用户隐私数据 ；
- 用户隐私信息收集 ：必须要有一个符合相关法规、规范和监管政策的用户隐私信息收集、使用和传输应用系统 ；
- 用户隐私信息的使用 ：用户个人数据只能用于与上述声明中陈述一致的目的 ；
- 用户隐私信息的安全 ：必须要采取合适的安全措施（如加密）来确保个人数据在传输、存储及使用过程中的安全性 ；
- 用户对隐私信息的管理权 ：隐私信息拥有者对个人数据有浏览及更新权，而对个人数据访问仅受限于相关的授权人员 ；
- 用户隐私信息的持有 ：必须有一个合适的流程（收集、使用与销毁等）来确保个人数据仅限于实现商业目的或法律限定的有期限内持有 ；
- 用户隐私信息的处置 ：过期或不用的个人数据必须采用安全且妥当的方式进行销毁处理，如使用加密盘销毁或碎纸器。

最后需要强调一下的是，无论是 CP 还是 CS，都将得益于更透明的云计算风险管理体系、更公平标准化的云计算使用条件、以及更健全的法律保护和监管环境。

27.4　案例研究：金融业的电子支付运营安全治理

27.4.1　需求分析

从广义上讲，电子支付是一种云计算服务，它是指电子交易的当事人（包括消费者、商家和金融机构或第三方支付服务提供方），使用安全电子支付手段，通过网络进行货币支付或资金流转的活动。电子支付已成为电子商务发展过程中最重要的一个环节。但在目前社会整体信用度欠缺、相关法律法规对电子支付的权利和义务界定尚不清晰的情况下，电子支付存在密码管理、网络病毒、木马、网络钓鱼等安全问题。调研数据表明，在不愿意使用电子支付的网民当中，高达70% 的人是因为担心交易与资金的安全性。

针对电子支付的各种不同的安全性要求，需要从技术实施、法律规范、运营治理等多个方面来保障电子支付数据的保密性、数据的完整性、交易者身份的确定性以及交易的不可否认性。

从技术上看，目前在电子商务界，已开发出相应的技术措施，较为成熟的有加密、访问控制与安全认证、防火墙、入侵检测、漏洞扫描等技术；国际上也已经形成了一些比较成熟的安全机制或协议，如基于信用卡交易的安全电子交易协议（Secure Electronic Transaction，即 SET 协议）、用于接入控制的 SSL 协议（Secure Socket Layer 安全套接层）、Netbill 协议、安全 HTTP（S-HTTP）协议、安全电子邮件协议（如 PEM、S/MIME 等）、用于公对公交易的 Internet EDI 等。

从法律上看，由于网络特殊的跨国性交易特性，除需要从业者的自律规范，更需要通过各国有关银行或金融的相关法规加以规范。

对于电子支付服务而言，相对于技术和法律两个层面，运营安全治理则显得尤为重要，因为支付服务更强调支付信息的保密性、支付流程的严谨性、支付过程的可追溯性，因此，支付提供方的组织架构的合理性、安全治理操作的完备性及对客户隐私的保护对保障消费者账户等金融信息的安全至关重要。

27.4.2　设计考虑

X 公司是一家国内领先的独立第三方支付企业，主要为各类企业及个人提供安全、便捷和保密的综合电子支付服务。根据国家金融服务领域的相关政策法规，X 公司基于 ISO/IEC27001 安全管理体系建立了信息安全管理架构，用于支持与保障公司业务发展目标，保证为客户提供优质的支付服务，并确保服务的安全、稳定、便捷、合规。

为此，X 公司在不仅技术层面针对不同安全层次实施了分层次的管理：物理环境安全管理、网络安全管理、主机安全管理、系统安全管理、数据安全管理、业务持续性管理、支付流程安全管理。更重要的是，X 公司建立了明确的安全管理组织与方针以及严密的安全监控措施，以保障安全管理措施能够被有效的落实，并与公司安全管理及业务目标保持一致。

27.4.3　安全运营治理实施

1. 运营组织架构

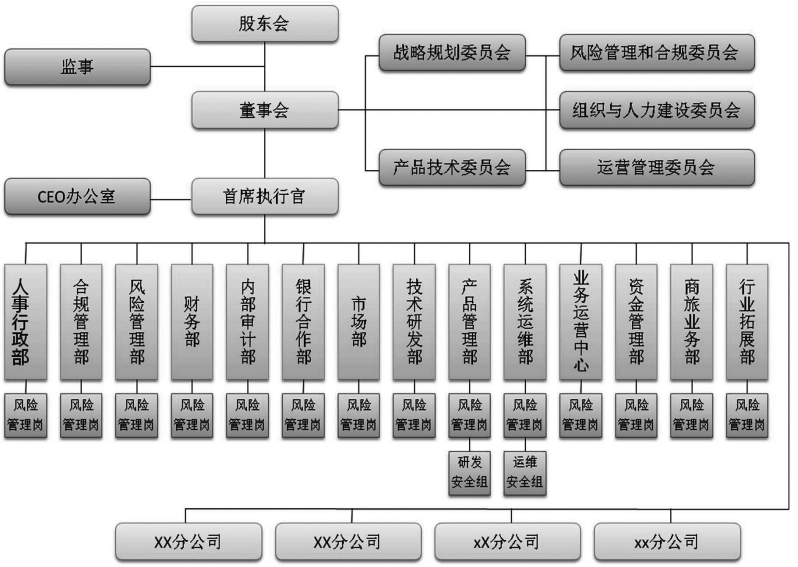


图27-5　X公司电子支付服务运营保障组织架构

（1）信息安全目标。信息安全管理必须从公司业务目标与需求出发，确保支持业务的信息资产具备有效的内部控制，通过持续改进机制防范风险事件，实现公司信息安全目标。安全目标主要包括下述 ：

- 保密性，即确保任何非授权主体无法访问或使用信息资产 ；
- 完整性，即确保信息资产的准确与完整，防止未授权的修改 ；
- 可用性，即确保授权主体在需要时能够访问及使用信息资产 ；
- 可靠性，即确保相关信息资产在规定条件内能完成规定功能 ；
- 不可抵赖性，即确保行为各方在行为发生后无法否认 ；
- 问责性，即确保所有信息资产都有明确的部门与岗位对其负责 ；
- 合规性，即遵循法律法规、合约责任及公司规章制度。

（2）安全管理组织。如图 27-5 是 X 公司电子支付服务运营保障组织架构。公司最高管理层为公司信息安全负责，并给予充分、明确的支持与指导、提供必要的资源保障、创造积极的内部环境。公司任命了信息安全管理层代表，设立了专职的安全中心、内部审计部、以及各业务部门的安全管理岗等职能，保证安全管理有充分的组织支持。

（3）风险为导向的管理方法。安全管理必须以风险为导向，通过对信息资产的识别，分析评估相应的风险，并且根据风险水平实行相应的处置策略与内部控制措施，兼顾成本与效果的平衡，最终使风险水平控制在可以接受的容量内。

（4）持续改进机制。安全管理体系通过有效的自我评估、内部审计、ISO/IEC 27001 认证机构审计、以及管理评审活动，及时识别内部控制缺陷，实施纠正和预防措施，不断完善信息安全

管理体系。

（5）技术与管理并重。快钱在信息安全管理体系建设过程中，坚持技术是手段，管理是保障，两者缺一不可，一方面提升信息安全管理中的技术应用水平，另一方面持续加强信息安全管理流程与内部控制的落实。

2. 安全运营监控

X 公司对电子支付服务的整个运行过程进行了全年 7x24 小时专人严密的监控，监控内容主要包括 ：服务可用性监控、交易正确性监控、产品服务监控、安全性监控等，涉及范围包括用户应用异常、用户交易异常、入侵检测、已知攻击检测、传输过程中数据窜改、峰值异常、网络异常等，其目标是对系统与产品事故进行及时预警、发现和处理，防止事故影响扩大，维护系统的稳定性，并对事故行为进行分析汇总，以建立相应预防措施，保障用户交易顺畅及用户账户安全可靠。

系统监控通过探针形式实现，监控探针形式包括实时可用性探针、交易正确性探针、产品合规性探针。其中实时可用性探针，由生产系统主动汇报自己生命特征，把生命数据放入监控数据库，通过监控控制台显示报警 ；交易正确性探针可以生产程序每步 DOUBLE CHECK 的方式来做，一旦异常，报警通知，也可以事后交易数据合规检查的方式来报警 ；产品合规性探针可以通过 SQL 查询统计的方式来监控报警。探针的数据获得后存放到监控数据库，控制台对每种业务的监控做好规则配置，一方面作为报警规则，另外一方面也作为显示报警的依据，方便监控人员监控跟进。当监控发现问题时，某些报警出现后，可以通过自动规则触发修复命令，系统自动修复，修复后再报告修复完毕，以便事后检查。监控过程如图 27-6 所示。

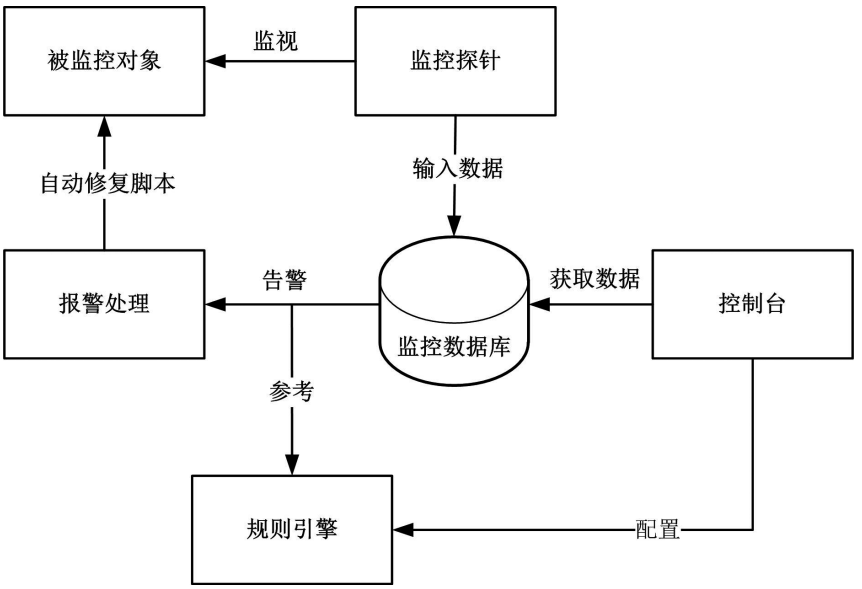


图27-6　电子支付运营过程中的监控

除上述从技术层面的安全监控实施外，X 公司还根据中国人民银行公布的《支付机构反洗钱和反恐怖融资管理办法》中的相关要求，建立了正式的反洗钱内部控制制度，对商户身份识别、身份资料和交易记录保存、可疑交易报告、反洗钱融资调查等环节进行了明确规范。同时，X 公

司还启用了反洗钱可疑交易监控系统，对商户可疑交易进行实时监控，防止商户通过该公司的服务进行任何非法交易。

27.4.4　成效评估

X 公司严格遵守金融服务领域的相关政策法规，以安全合规为前提，积极推进各类创新型金融服务的发展和应用。目前 X 公司正在与超过 180 万家商业合作伙伴一道，共同创造信息化金融服务的巨大价值。仅 2011 年，X 公司的交易量总额就突破了 12,000 亿元人民币。

27.5　本章总结

云计算安全治理是云计算安全在生产环境下安全运营的重要保证，一方面验证了安全技术实施策略的有效性，同时也是发现云计算在构建中遗留的安全问题并进行修复的重要环节。

云计算安全治理本身是一个不断改进、逐步提升治理效果、降低安全风险的过程。在此过程中，云计算安全治理策略为云计算安全治理过程提供了指导方针与实施指南，它从安全治理目标、管理意图、职责与权限、相关术语及操作细则定义等方面规划云计算安全治理内容。

安全监控与事件响应、威胁管理和渗透测试、安全审计及日志以及隐私保护是云计算安全治理主要控制措施。安全监控对云计算运行环境的安全状态提供全局性监视机制，其提供的安全告警或安全事件是云计算运行过程进行事件响应的重要依据。事件响应按预先定义的安全级别进行相应的安全事件处理，以降低安全风险或消除安全隐患。威胁管理则从“点”的角度提供了对网络及应用系统脆弱性检查与修复机制，增强了系统的主动安全性，降低了网络或系统被攻击的风险。渗透测试基于“黑盒子”测试机制对云计算整体环境进行测试，以主动发现可能存在的漏洞与风险。

在云计算环境下，安全审计和隐私保护既是监管和法规要求，也是云计算提供商内部安全治理的需求。安全审计不仅为事后取证、业务恢复提供重要依据，定期的审计也可让 CP 提前发现可能存在的安全隐患，避免安全风险。隐私保护是一个相对复杂的云计算安全保护问题，它不仅依赖于与立法、技术、监管等宏观外部环境，更依赖于云计算提供者与消费者之间相互作用的微观内部环境。

参考文献：

[1] EC, Concerning the Processing of Personal Data and the Protection of Privacy in the Electronic Communications Sector , Official Journal of the European Communities , L 201/37 , July 2002，http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2002:201:0037:0047:en:PDF

[2] FTC, What's happened to PETs, http://www.ftc.gov/os/comments/privacyroundtable/544506-00076.pdf

第28章　云计算业务连续性保障

28.1　云计算业务连续性及其挑战

28.1.1　什么是业务连续性

业务连续性（Business Continuity）是一个服务提供商为了确保其客户能够随时访问到其的业务功能而进行的一系列的运营活动。这些活动包括许多日常运营任务，如系统备份、数据备份、系统变更控制、系统实时监控等。业务连续性不是在灾难发生时才去实施的系统 ；业务连续性是指为维护服务的高可用性、一致性和可恢复性而执行的日常业务活动。

业务连续性是一个很大的范畴，它囊括一个服务商的日常运行的各个方面，如基础设施、运营团队、灾备系统、及灾难恢复流程等等。本节将着重剖析云计算实现业务连续性挑战和实践。

28.1.2　云服务提供商面临的挑战

近年来，恐怖活动、种族冲突、SARS、禽流感、海啸、雪灾、地震等各种天灾人祸，使不少企业陷入困境。2011 年三月，日本发生的 8.9 级大地震和随后的海啸造成了日本全国各地的数据中心大面积的电源和网络服务中断，这也导致日本企业重新考虑其传统的灾难恢复策略。几个星期后，由于一个故障路由器的升级失败导致亚马逊在美国东部一个数据中心的弹性快存储（Elastic Block Storage，EBS ）系统的失败，这一 EBS 失败导致了连锁反应，使得几百个客户的服务停机，包括许多 Web2.0 公司，如 Foursquare 和 Reddit。

今天大多数的云计算的使用者，无论是企业或者是个人用户，都无法容忍服务中断或数据丢失。其对服务可用性的要求达到 99.9% 至 99.99% 的服务可用度要求。这是来自前端用户的最直接的挑战。对后端的要求是 ：技术方案的实施与管理，及其成本效益。

面对这些灾难，云计算服务企业高层和管理人员如何未雨绸缪，妥善应对这些冲击呢？简单的灾难备份是远远不够的，答案只有一个 ：进行业务持续性管理，即 Business Continuity Management（BCM）。BCM 的一个重要任务是找出业务最大容忍的中断时间以及最低可以接受的服务水平或高服务可用性，这是非常关键的一步，这不但意味着要满足行业监管和利益相关方的要求，也意味着资源的投入，包括人员、场所、设备、技术、供应商等。

服务的可用性是指任何时候，任何地点，任何环境用户都能够获得他所需要的服务的比率。最简单的比方，一个手机客户希望在任何时候，任何地点，任何环境，他 / 她能够拿起电话就能打电话。 这要求服务运营商有完备的服务高可用性解决方案。这个方案需要涉及到系统的各个层次，首先要有全球网络，使得服务可以生存于不同的地理位置，以应对大的自然灾害和某一局部地区的网络故障 ；其次要求相同的服务被部署到位于不同的地理位置的数据中心，使得用户的请求可以快速地被转移到另立个地点 ；再者，还需要数据复制来保证多个地点相同服务的数据是同步的。只有实现了这些多个层次的高可用性部署和自动化，才能实现服务的转移对用户透明。

服务的可用性和投入是成正比的。高的可用性需要更高的投入，越往上，投入的份额越大。

要获得 99.99% 的服务可用性比要获得 99.9% 的高可用性的投入可能要高很多倍。服务运营商是企业，所以在实际操作中要做可用性和投入产出比的权衡。决策过程中要考虑这些因素 ：

- 用户群是谁，他们对可用性的要求是怎样的。他们能承担的价格是怎样的。
- 用户群有多大，公司有没有足够的收益能保证正常运营下去。
- 当前的技术条件的限制，包括网络连接状况和带宽，软件是否具备，等等。

28.2　云计算的业务连续性方案概述

28.2.1　业务连续性的管理

云计算业务连续性管理（Business Continuity Management，BCM）是支持整体云计算服务业务的连续性的管理过程，它的目标是通过确保云计算所需的 IT 技术和服务设施（包括计算机系统、网络、应用程序、数据存储库、电信、环境、技术支持和服务台）能够在预先协定的时间内恢复服务。更具体一些讲，云计算业务连续性管理要达到以下的目的 ：

- 维持一套服务连续性计划（Service Continuity Plan），以支持整个组织的业务连续性计划。
- 进行定期的业务影响分析（Business Impact Analysis，BIA）演练，以确保在业务的要求不断变化的情况下业务连续性计划得以维护。
- 定期进行风险分析，特别是在结合业务可用性管理和安全管理流程来控制 IT 服务的业务风险。
- 确保适当的业务连续性和恢复机制能够满足制定的业务连续性目标。
- 确保在成本合理的前提下采取和实施积极主动的措施来提高服务的可用性。

业务连续性管理（Business Continuity Management，BCM）包括业务接续运营的决策过程、在哪里以及怎样进行业务接续、需要采取什么行动，以及接续哪些业务到何种程度等等。下面将简要介绍业务连续管理的五大关键部分[1-3]。

（1）业务连续性项目启动和管理

首先要确定业务连续性计划（Business Continuity Plan，BCP）过程的需求，包括获得管理支持、以及组织和管理项目使其符合时间和预算的限制。业务接续计划是由一系列简单明确的指令构成，恢复团队完全可以按照这些指令进行恢复操作。各种操作之间的相互关系也必须加以明确说明。所有的指令和说明必须明白无误，以免因可能引起误解或不明了而导致时间损失。

（2）业务连续性风险评估和控制

第二步要确定可能造成业务中断的灾难、具有负面影响的事件和周边环境因素，以及事件可能造成的损失、防止或减少潜在损失影响的控制措施。进而需要进行成本效益分析以适当调整控制投资并且达到消减风险的目的。

风险评估是预案编制的基础。风险评估技术比较成熟，有很多成熟的风险评估模型可供借鉴，组织可以参照这些评估模型和方法，建立自己的评估模型。对于大型组织，建议选用一款风险评估专业软件，以便提高风险评估的效率和准确度。对于识别出的风险，按照风险发生的概率、影响程度等进行风险排序和分类，确定主要风险，找出预防风险的防范措施和减缓风险的控

制措施，同时要注意分析每个风险可能引发的衍生风险。

完成主要风险识别后，还需要分析这些风险对组织业务的影响程度。 业务影响分析是确定由于中断和预期灾难可能对组织造成的影响，以及对这些影响的定量和定性分析，确定关键环节的相关性、恢复优先顺序，以便确定恢复的时限。业务影响分析包括对中断后果、中断对业务的影响和中断造成的损失的评估 。中断后果评估内容包括 ：提供产品或服务的能力中断、资产（人员资产、数据和信息资产、有形资产和无形资产）损毁或损失、违反法律法规、引起公众关注。中断对业务的影响包括经济影响、客户和供应商的影响、公共关系和组织信誉的影响、法律方面的影响、对组织运作的影响、对人力资源的影响、对其他资源的影响。对损失的定量分析包括 ：财产损失、收益损失、罚款、现金流、法律责任、人力资源、额外支出。

业务影响分析结果的正确与否，直接关系到各种预案的有效程度。为了保证业务影响分析的准确性，很多组织引入外部的专业机构，借助专业的工具软件。

（3）业务连续性计划的制定

下一步是设计、制定和实施业务连续性计划以便在恢复时间目标范围内完成恢复。一个 BCP 必须周期性地加以检查和维护。一旦有新的系统、新的业务流程、或者新的商业行动计划加入企业的生产系统或者信息系统，引起企业整体系统发生变化时，就更应该强制启动这种检查程序。业务连续性计划包括应急响应和运作的步骤和各种预案，包括建立和管理紧急事件运作中心，该中心用于在紧急事件中发布命令。

所有的策略、流程，都应该围绕着降低业务中断发生的概率、快速恢复以减少业务中断造成的损失、控制业务中断的影响范围、维护社会的和谐与稳定等目的。在制定这些策略、流程时，不能局限于组织内部，要有大局观，这是因为，有些业务中断事件的发生，受影响的不仅是组织内部，可能还会蔓延到周边地区，甚至会对国家的发展造成影响，有的还会引起国际争端。应急策略是应急的基本原则。应急策略应该包括 ：

事件分类分级原则 ：按照事件的性质、对客户影响的严重程度、可控性和影响范围等因素一般可以将事件分为四级 ：I 级（特别重大）、II 级（重大）、III 级（较大）和 IV 级（一般）。

事件处置原则 ：包括应急响应处置原则，应急救援处置原则，以及恢复与重建处置原则。应急响应处置原则包括事件报告、现场紧急处置、现场事件初步评估等原则和处理流程，这是编制应急响应预案的基准。最大限度地降低服务中断对客户业务的影响是应急救援处置最基本的原则。尽快恢复关键流程、设施，以实现业务的早日正常化，是恢复与重建的原则。

事件发布原则 ：事件发布包括组织内部的事件通报、与供应商和客户的沟通、媒体信息发布。事件发布要及时、准确、透明，必须由专人 / 部门发布，切忌多渠道发布。对于容易引起社会问题的事件，应建立信息发言人制度，并密切关注网络传播的动向。对负面的或恶意信息如何应对，更需要事先制定出应对之策，以避免临时决策可能出现的失误。

（4）维护、演练和实施业务连续性计划

业务连续性计划制定后一大重要任务是建立对机构人员进行意识培养和技能培训的项目，以便业务连续性计划能够得到制定、实施、维护和执行。 需要定期对预先计划和计划间的协调性进行演练、并评估和记录计划演练的结果。通过与适当标准的比较来验证 BCP 的效率，并使用简明

的语言报告验证的结果。

保证业务连续性计划中使用的信息、数据是最新的，这一点很重要。因此，只要这些信息或数据发生变化，哪怕是很微小的变化，只要有可能影响到业务连续性计划的实时性、实用性或完整性，就必须对业务连续性计划进行更新。对于任何一个组织，变化是司空见惯的，所以说，业务连续性计划维护的工作量是非常大的。

根据业务连续性计划的性质和类别，业务连续性计划的维护可以分为定期维护、不定期维护两种形式。当业务流程、关键设施、重要数据等发生变化时，必须立即对预案进行更新维护。

业务连续性计划管理由组织中的紧急事件运作中心或相关部门负责。业务连续性计划管理团队的主要职责是协调各方的关系，检查每个过程的交付成果，负责与组织高层的沟通，以确保业务连续性计划规划、编制、维护全过程的顺利进行，并负责制定业务连续性计划培训计划与演练方案。

（5）公共关系和危机通信

业务连续性的最后关键环节是制定、协调、评价和演练在业务连续性危机情况下与媒体交流的计划。制定、协调、评价和演练与员工、主要客户、关键供应商、业主／股东以及机构管理层进行沟通的计划。确保所有利益群体能够得到所需的信息。 必要时还要建立适用的规程和策略用于同地方当局或政府监管部门协调响应、连续性和恢复活动以确保符合现行的法令和法规。

28.2.2　业务连续性的技术方案：灾备系统概述

灾难恢复对于保证商业运作的正常运行至关重要。在这一章将介绍服务灾难恢复系统和服务高可用系统。

那么怎样利用云计算技术来达成服务系统的高可用性设计和部署呢？一种普遍流行切实可行的方法是将应用设计冗余与云管理层的自动化有机的结合在一起。第一步是要求应用架构的解决方案能够承受单个节点的故障，无论这些节点是服务器、存储卷、甚至是整个数据中心。第二步是必须独立地考虑每一层组件（例如，Web 层、应用层、数据层）抗单点失败的有效而经济的设计方案，要考虑到的因素包括数据中心基础设施、互联网带宽、架构的成本和应用的性能。用户可以利用多种多样的技术来设计解决方案，本章的 28.3 “ 灾备系统架构 ” 章节中有详细地描述。

仅仅建立了服务应用的高可用性系统是不够的，真正的关键在你的应用服务架构是如何运作的。系统的哪些部分应该对失败自动响应，哪些部分需要人工响应呢？更具体一些讲，如果一个给定的云资源失败了（无论是磁盘驱动器、服务器、网络交换机、SAN，或一个完整的地理区域里的所有应用设施失败了），如何无缝地启动或故障转移到另一个资源或系统来保持业务的正常运行呢？在理想的情况下，当然是越多故障转移自动化，服务运营就越平稳。

实现这一自动化运营水平就要求系统的设计和配置很容易复制。例如，装有应用程序的服务器能够在不同的云基础设施上迅速地重新部署以缓解某一群服务器失败。正确的云管理解决方案是应通过可定制的最佳实践来简化整个部署过程。它还应该通过一个中央管理仪表板来提供所有基础设施的完全可见性。这样管理员可以通过监控应用服务的性能并基于实时需求进行容量变化。这样当灾难袭来时，相同的自动化和控制能使企业使用更多服务器扩大容量，或能够迁移整

个服务器部署到一个新的基础设施上。

以下的章节将集中阐述什么是服务备份系统以及如何利用服务备份系统来达成云计算的业务连续性的目标。全球备份系统是实现云计算的业务连续性的一种通用的服务高可用性系统解决方案，不同的云计算的服务备份系统的具体的实施方案因云计算性质和架构的不同而不同。将首先阐述服务备份系统如何保障云计算服务的业务连续性，然后以思科 / 网讯（Cisco/WebEx）的服务备份系统为例深入讨论其具体框架与实施以及该系统如何保证 WebEx 网络会议服务的业务连续性。

1. 什么是灾备系统

灾备系统（DR，Disaster Recovery）是一种基于不同地理位置的服务高可用性解决方案。它通过在不同的地理位置建立相同的云计算服务实例，实例之间进行数据实时复制，以此来实现当城市断电，主干光纤网络中断，以及地震等自然灾害造成的大规模的区域云数据中心失效时，将云计算服务自动转移到另一个地理位置的云数据中心，从而保证服务可以继续的一种策略。

相对于本地高可用性的策略（比如双机备份、群组），服务的灾备系统是更高级别的高可用性方案，同时也是 IDC 级别的备份系统（相对而言，双机备份、群组等只是机器级别的，或组件级别的备份），当一个地点的服务完全中断，其他地点的客户被透明地转移到另一个地点。

2. 灾备系统的重要性

对于重要的云计算服务来说，服务的不间断性对于客户是至关重要的，比如通信、股票交易、电子商务平台、社会网络服务平台、搜索引擎等。这些云计算的特点是全球化，不太有区域的限制。比如说某电子交易平台，它的数据中心在美国的西部城市硅谷，他的客户可以是来自亚洲、非洲、美洲等不同的地方。现在假如硅谷发生 8.5 级地震（加州是地震多发地带），整个数据中心瘫痪，可是对于硅谷以外地区的用户来说，他们仍然希望交易可以继续进行。怎么办？有效的措施是在硅谷以外的地区建立第二个数据中心，并且能够实现服务的自动转移。

另外，对于一个庞大的云计算系统，软件或硬件系统的升级和维护，也是一个非常重要的方面。由于系统的庞大，为保证服务终端用户不受影响，比较安全可靠的做法是，把整个服务转移到另一个地理位置，然后对本地的系统进行升级和维护。在这种情况下，一旦升级失败或出现暂时的异常情况，维护人员在不影响终端用户的情况下，仍然有足够的时间来处理异常情况。这种异常往往会发生在网络设备的维护和升级上，某些关键的网络设备（比如负载均衡器、路由器等）一旦出现异常，就可以引起某个地理位置的整个服务的中断。

第三，系统的实时性也是终端用户所关心的一个方面。对于某些云计算系统（例如云存储），可以把服务分散在全球的多个位置，当所有的地点都是可用的情况下，可以采用就近服务的原则，使得服务的实时性增强 ；另一方面，当某一点发生异常，服务可以实时地转移到就近的位置。从这个意义上来说，不同地理位置的服务已经没有主服务和备份服务之分了，实际上是主 - 主（Live - Live）的云计算服务模式了。

所以总结起来，灾备系统的重要性主要体现在三个方面 ：

- 提供跨区域的服务级别的备份系统，从而应对某点发生大型自然或人为灾害的情况 ；

- 提供更可靠的系统维护方案 ；
- 提供全球多点服务，缩短系统的响应时间，提高系统的实时性。

3. 灾备系统的挑战

在服务的灾备系统中，当主系统有问题时，网络设备要能够及时感知，并且能够及时地把用户的请求无缝地转移（Failover）到备份系统 ；同时还要求用户相关的数据要全部存在于备份系统中。这就要求服务备份系统能够很好地实现以下的特性 ：

（1）作为网络层，要能够感知整个云计算系统的健康状况。

任何一个关键组件的失败，要能够触发网络系统的服务转移。这个一般是通过定义每个组件的运转状况检查（Health Check）来实现，难点在于如何适当地定义云计算 Health Check 的方法，方法要有效并且高效。所谓有效是说，这个 Health Check 的方法要能够涵盖各种组件失效的情况，不能遗漏，否则会造成云计算系统已经不能正常服务于终端客户，可是却没有触发网络系统的服务转移。所谓高效，是指方法被调用后，能够很快地返回。这是因为网络设备会频繁的调用 Health Check 方法（一般间隔是 5 秒），一方面如果方法不能及时返回，会造成超时，从而引起错误的服务转移 ；另一方面，如果返回太慢，可能是由于消耗的系统资源太多，从而造成整个云计算系统的性能下降。

（2）数据的同步和复制是另一个很棘手的领域。

数据一般包括缓存数据（内存中的数据）和在线数据（永久存储的数据，包括文件和数据库）。因为云计算转移很多时候是无法预料的，这就要求主系统中产生的数据要能够尽快的被复制到备份系统中（理想状况是实时复制）。可是服务备份系统是分布在不同的地理位置，所有复制需要跨若干个网段，复制的效率就成了一个关键性的问题。另一种情况是当一点的系统失效后，数据复制可能无法进行。当这一点恢复后，在另一点产生的大量的数据要能够被尽快地复制回去。由于大量数据的复制需要比较长的时间，可是服务已经恢复，新的数据已经开始产生，这样有可能造成两点同时修改一份相同的数据，这就产生了数据冲突。在这种情况下，需要根据具体的业务逻辑设计合理的冲突解决方案。

28.3　灾备系统架构

通过上面的介绍，可以看出服务备份系统是一个庞大而复杂的系统，它包括以下几个子系统：

- 网络
- 系统
- 应用系统
- 数据同步系统 ：存储数据（storage）及数据库数据（database）的同步

图 28-1 是服务备份系统的基本框架 ：

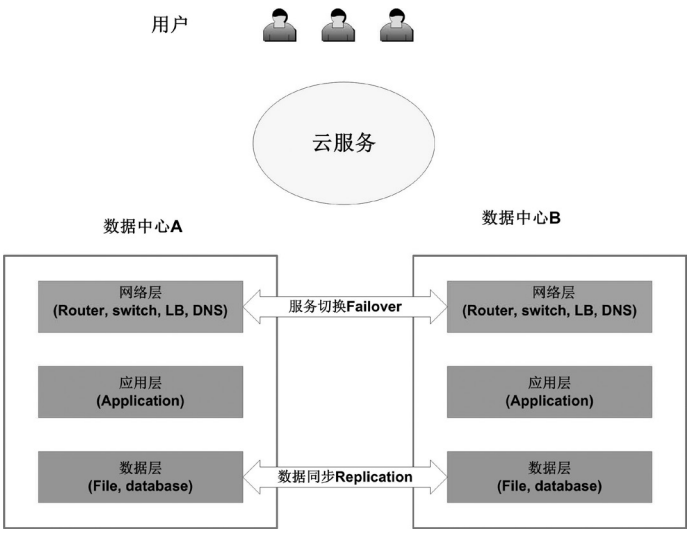


图28–1　服务备份系统基本框架示意图

28.3.1　网络系统

网络层是整个云计算系统中的最高层，终端用户通过网络访问服务，所以首先要求主系统和备份系统所在的不同的数据中心之间要有必要的网络互联 ：

- 终端用户要能够自由地通过广域网来访问任何一个数据中心，这样才能保证无论服务是在哪个数据中心都可以被客户访问。
- 数据中心之间要有专线连接，这样才能保证跨数据中心之间的带宽和稳定性，这个主要用于跨数据中心间的数据复制和应用程序的跨地域访问（这在 Web 2.0 的应用中很普遍）；还有用于控制服务转移的网络设备也需要稳定的跨数据中心的数据交换，如 Load Balancer 的 GSLB（Global Server Load Balance，全球服务器负载均衡）、DNS server 间的同步等等。

网络层所承担的功能有 ：

- 转发 ：把客户的请求转发给适当的应用服务器 ；
- 负载平衡 ：通过虚拟 IP 地址来绑定一组应用服务器，当大量的客户端请求进来后，负载平衡器（Load Balancer）会根据后端 N 台应用服务器的状况来把所有的请求平均分配给所有的服务器 ；
- 自动剔除故障服务器 ：当其中的一台或多台服务器发生故障时，负载平衡器感知后，会自动的把请求转发给其他的健康的服务器，从而避免客户端的异常 ；
- 云计算服务转移 ：在服务备份系统中，当网络设备感知主服务有问题时，会自动关闭主服务的虚拟 IP（VIP）。当客户端重连时，就被自动地重定向到备份服务的虚拟 IP。

服务备份系统服务转移过程如图 28 - 2 所示。一般客户端都是通过域名（DNS name）来访问服务，假设服务的域名是 ：boo.com，在域名服务器上会有两条记录，分别对应着主服务和备份服务的虚拟 IP。 如果主服务是健康的，当客户端访问 boo.com 时，域名服务器会返回主服务的虚拟 IP 给客户端，客户的请求就被定向到主服务 ；如果主服务有问题，主服务的虚拟 IP 就会

自动关闭，当域名服务器感知后，就会返回备份服务的虚拟 IP 给客户端，这样客户的请求就被自动转移到备份服务，从而实现服务转移。

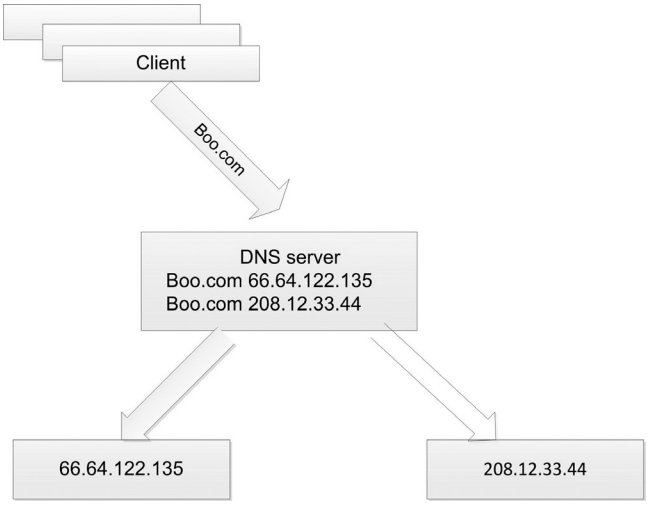


图28-2　服务备份系统服务转移示意图

28.3.2　云计算应用系统

应用层是云计算的业务逻辑层。当终端用户通过网络访问服务时，网络层最终把请求转发给了相应的应用服务器（物理的或虚拟的），一般现代的企业级的云计算系统都用一群功能相同的服务器（集群）来实现一个服务。关于集群的结构，如图 28-3 所示。

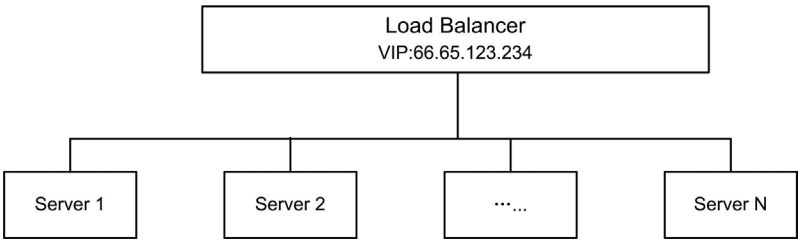


图28-3　应用层集群的结构示意图

集群的好处有 ：

- 动态扩展容量 ：第一次部署可能只有 2 台服务器，后面如果发觉访问量增加了，可以加一台或多台服务器到已经存在的集群。
- 一定的容错能力 ：当一个集群中有多台服务器时，如果其中的部分服务器发生故障，负载平衡器会自动地把这些服务器从集群中屏蔽掉，从而保证用户的请求永远只会被转发给健康的服务器。

那么负载平衡器是根据什么来判断和某一个 VIP 相关联的服务器的健康状况呢？这依赖于每一个服务器本身提供的健康检测方法（Health Check Method）。当某一个服务器被加入到负

载平衡器的某一个特定的 VIP，负载平衡器就会每隔一定的时间去访问这个服务器提供的一个特殊的方法，比如 TCP call、HTTP call、ping 等等。当服务器接收到来自负载平衡器的特殊的调用时，就进行自我检查，如果发现自己的服务正常，就返回一个固定的结果给负载平衡器（比如“ OK ”），负载均衡器就知道这个服务器仍然是健康的 ；相反，如果服务器在做自我检查时，发现已经不能提供正常的服务了（比如应用服务器不能连接数据库），就会返回一个固定的结果给负载平衡器（比如“ NOT_OK ”），负载均衡器就知道这个服务器已经有故障了，就会把它从这个 VIP 中剔除。当某个 VIP 中所有的服务器都有故障，这个 VIP 自己的状态就会变成不可用。

同时域名服务器也会对主和备份的 VIP 做同样的健康检查，当主 VIP 是活的，域名服务器永远返回主 VIP 给客户端 ；当主 VIP 下的所有的服务器都死了，主 VIP 也就不可用了，这时域名服务器会返回备份 VIP 给客户端，这时就发生了服务转移（Failover）。

可以看出，最终对 VIP 以及服务转移起决定作用的仍然是服务器本身。更准确地说，是服务器提供的健康检查方法（Health Check Method）。所以应用服务器的健康检查方法的设计就至关重要。在应用服务器的健康检查方法中，有以下几点需要注意 ：

（1）全面性

比如说一个 Web 服务器，它要想能够为客户提供正确的服务，需要访问数据库，需要访问网络文件系统，同时需要自己的 Web Server 能够在规定的时间内返回结果，所以在这个 Web 服务器的健康检查方法中，应该包含对相应的数据库连接的检查，对相应的网络文件系统的挂载点的检查，以及对特定标志的检查（用于手动服务转移，后面有详细介绍）。

（2）高效性

由于负载平衡器会很频繁地进行健康检查，所以如果健康检查方法的实现不够高效，会消耗掉很多的系统资源，这是不合理的。如果这种情况发生，可能会使云计算系统越来越慢，最终可能导致负载平衡器端超时，从而认为这个服务器有故障。一般来说，对于磁盘等的 IO 操作要尽量做到精简 ；对于服务器本身的服务的检查，也尽量只检查服务的状态而不涉及具体的业务逻辑的检查。

（3）标准化

当一个云计算系统很庞大时（比如有 5 ～ 10 种以上的应用服务器类型），尽量做到用相同的健康检查方法（比如 HTTP 请求），这样便于网络设备的配置和管理，也便于不同的开发人员的实现。

28.3.3　数据同步系统

数据同步系统包括数据库和文件系统。在服务备份系统中，永久存储层存储着用户的数据，当服务转移发生时，用户仍然需要能够访问他们的数据，这就要求永久存储层要有一个健壮、高效的跨网的数据复制通道。这通常是整个系统中最具挑战性的部分。

通常关系数据库已经比较成熟，比如针对 Oracle 的 Stream 和 GoldenGate 数据库复制工具，以及 Quest Software 的 Shareplex，都是比较成熟的实时双向复制工具。数据库复制工具的工作机制如图 28-4 所示。

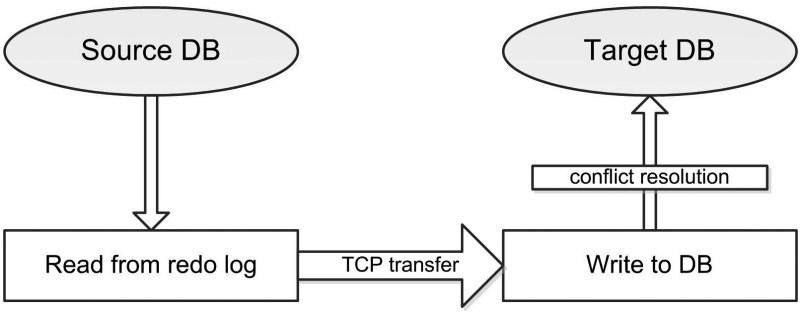


图28-4 数据库数据复制工作机制

文件系统的实时双向复制机制类似于数据库复制工作机制。

另外由于复制是实时双向的，所以必然会存在两边同时修改同一个文件或数据库的记录，当复制到对方时，必然会产生数据冲突。这时要根据应用程序和客户的需要，进行取舍（无论如何最终只能保留一个结果）。这就要求在数据复制工具中，要能够根据客户和应用程序的需要，灵活地定制数据冲突解决方案，并且要求数据复制工具能够感知数据冲突的发生，并且能够选择调用合适的冲突解决方案来修正数据。

通常除了实时双向的数据复制之外，还要开发高效的批量数据同步工具，这个主要用于手工维护。当某种异常情况发生后，积攒了大量的数据需要复制，并且需要有选择的复制，这个时候利用同步工具手工同步就更加灵活和高效。

28.3.4　管理工具: 手动服务转移

在日常的云计算系统维护中，由于需要进行软件、硬件升级，单个云计算系统不可避免地需要停止服务一段时间，可是客户的服务还要继续。在这种情况下，可以通过手动服务转移来实现。手动服务转移的工作原理和自动服务转移的工作原理是一样的。一般采取的措施是，让单个应用服务的健康检查程序检查一个特定的标志（特殊的文件，一般放在网络文件系统上，供同一个集群的所有应用服务器共同使用），当需要手动服务转移时，就改变这个标志，让同一个集群的所有应用服务器的健康检查全部失败，这就造成主服务的 VIP 变成不可用的，于是域服务器就返回备份服务的 VIP 了。如果集群很多的话，一个集中管理所有集群的服务转移状态和方便手动服务转移操作的平台（GUI）是很有必要的。

28.4　灾备方案的成本效益

对一个云计算运营商而言，要完成一个灾备中心，技术方案只是一个方面。换句话说，只要投入足够的财力和人力资源，灾备中心是可以建立起来的。

但是对于管理者而言，灾备中心成本是一大挑战。灾备中心如果只是作为备份，平时闲置，对运营而言是很难接受的。因此，灾备中心的成本效益是另外一个挑战。

今天的灾备系统已经不是单纯的灾难备份系统，同时也希望把它作为常规系统的一部分，在正常情况下可以起到均衡负载的作用，使用户可以获得更好的用户体验（比如更短的响应时间）。

28.4.1　灾备资源的合理使用

这里所讨论的服务备份系统，其实仍然是主 - 备（primary-backup）的工作模式，也就是说在同一时刻，只有一边服务于客户，另外一边处在待命（hot standby）的状态，只有当服务转移发生时才会被使用。如果主服务和备份服务的配置相同，理论上只有 50% 的利用率。通常可以通过以下方式来提高系统资源的利用率 ：

（1）主系统和备份系统共用硬件资源

例如有 A 和 B 两个数据中心，每个数据中心部署相同的硬件资源，每个硬件资源上启动逻辑上相互独立的两组服务，一组是主服务，另一组是备份服务。这样在 A 和 B 分别有一个主服务和备份服务，同时 A 的备份服务是 B 的主服务的备份，B 的备份服务也是 A 的主服务的备份，如图 28-5 所示。

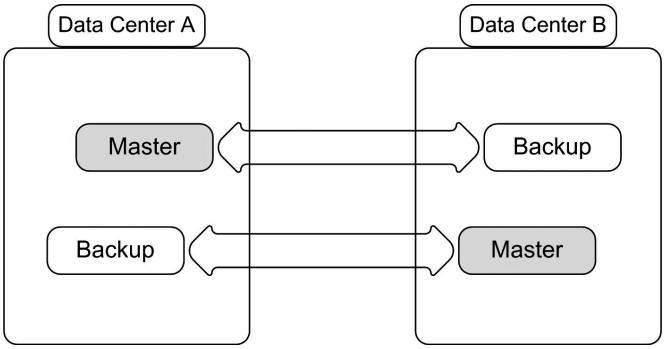


图28-5　主系统和备份系统硬件资源共用示意图

（2）备份服务的资源共享

如果有两个以上的主服务集群，可以考虑多对一的模式，比如有三个主服务集群，考虑到三个集群同时不可访问的几率很小。假设平时主集群的负载上限是 60%，那么三个集群公用同一个备份集群，备份集群的系统资源的规模可以设为一个主机群的 2 倍，这样既可以保证三个主服务同时发生服务转移时仍然可以工作，又起到了节省系统资源的目的，如图 28-6 所示。

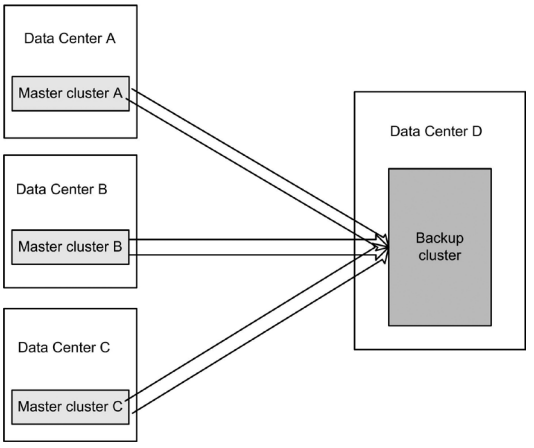


图28-6　备份集群的系统资源共享示意图

28.4.2　公有云和私有云之间的结合

在云计算的提供商中，大部分的服务提供商是 SaaS 和 PaaS。因此，SaaS/PaaS 的服务商可以充分利用 IaaS 提供的公有云的服务，来做自己的灾备的方案，或作为方案的一部分。

这里的挑战是，业务系统需要能够运行在不同的基础设施之上，无论它们是公有云，如亚马逊、Rackspace 公司，或者是企业的私有云，而且能够在不同的公有云之间或公有云与私有云之间快速和高效地切换。

安全性是另一个重要的挑战，如何把企业的敏感数据放到公有云里并且保证其安全性是另一个很大的话题。企业应该根据不同的客户的需求和目前的技术水平来确定备份方案。如果数据很敏感，要求级别高（比如美国税务局的纳税人账户信息），仍然需要考虑私有云的相互备份方案。

28.5　案例研究：云服务提供商WebEx的灾备系统

如果考虑到技术上的功能和性能，运营管理和成品效益，一个灾备系统的建立的难度是很大的。因此，选用灾备系统的建立做为案例做进一步的探讨。

这里介绍 WebEx 的灾备方案 GSB。 GSB 是 WebEx 在私有云上建立的。但是，它所面临的挑战，要解决的问题，相应的设计思想，方案的要点和难点，都是云计算提供商，无论是用私有云或者是公有云，都通用的。

28.5.1　背景介绍

WebEx 是全球最大的网络会议服务运营商。GSB（全球服务备份）项目的需求是，当 WebEx 在原来的系统中想把可用性从 99.9% 提高到 99.99% 遇到了瓶颈，这个时候要求进行技术的革新和系统的改造升级。在技术分析的过程中，发现不只是突发性的故障影响了可用性，其实有计划的产品维护和升级也会造成服务中断。所以原来的位于一个地理位置的集中式服务部署方案很难保证 99.99% 的可用性，我们意识到位于不同地理位置的 GSB 系统是必须的。

WebEx 的网络会议云计算系统的可用率一直保持在 99.99% 已经有很多年了，服务备份系统发挥了关键性的作用。

WebEx 总共有 7 个数据中心遍布全球，40 多个网络会议集群，对于这么一个庞大而复杂的全球云计算系统，WebEx 在四个主要的数据中心总共建立了四个大的备份集群，平均每十个主集群公用一个备份集群，备份集群的系统资源是单个主服务集群的 4 倍。

28.5.2　WebEx GSB架构

WebEx GSB 是标准的主 - 备模式，下面以 San Jose 和 Denver 的两个数据中心为例（见图 28-7）。网络层和服务备份系统有关的主要设备是域名服务器（GSS）和负载平衡器（ACE），位于 San Jose 和 Denver 的数据中心的 GSS 之间是实时同步的，相同的域名在两边的配置是完全一样的，这样保证无论用户访问到哪一边，获得的结果是完全一样的。负载平衡器主要对 Web Server 做负载平衡。

为使系统设计和维护简单化，应用服务器尽量保持较少的 cache 数据，对有需要同步的数据尽量保存在文件系统或者数据库中，这样便于复制。

WebEx 的数据库绝大部分是 Oracle，数据库复制采用 Quest 公司的 Shareplex 来完成，数据库的复制是实时双向的。

WebEx GSB 的文件复制完全是自主研发的，对于跨数据中心的文件传输，自主开发了一个文件复制服务程序（GFRS），它使用 TCP 协议实现跨网络传输。为了提高传输的效率，实现了对大文件切片后多线程并行传输 ；对于小文件，可以打包批量传输，从而提高文件传输的效率。

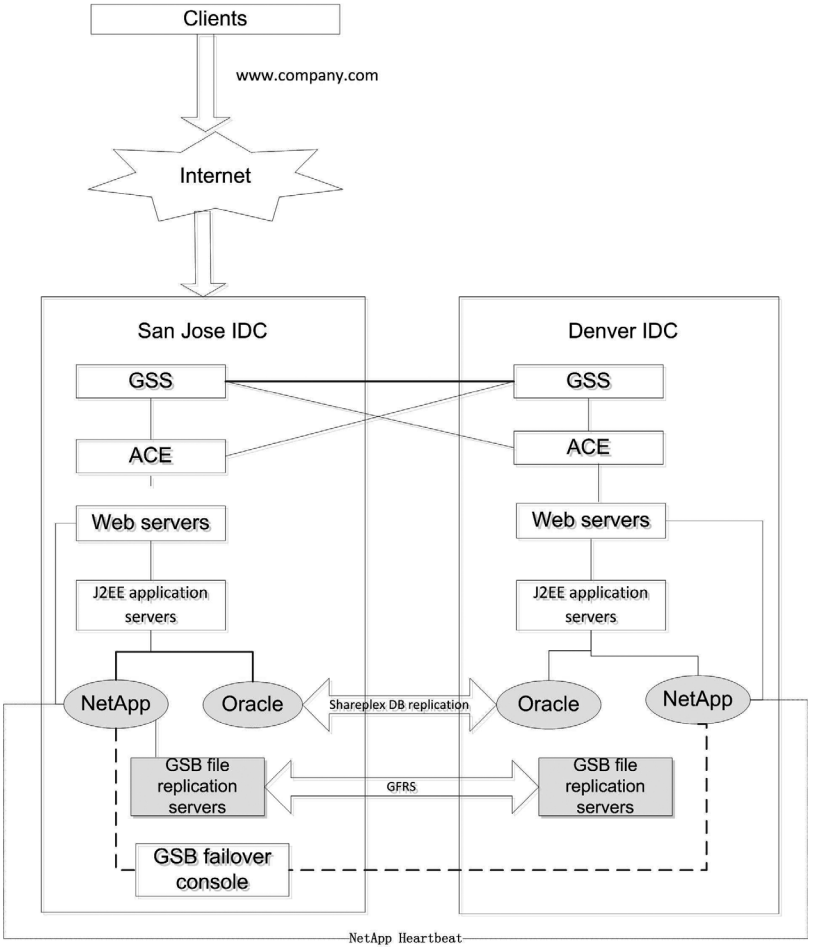


图28-7　WebEx GSB主-备模式示意图

同时 WebEx 有 40 多个群组，为了保证服务转移操作的方便性，开发了一个 WEB GUI 供管理员方便地进行服务转移和查询状态。

28.5.3　WebEx GSB的设计挑战和要点

GSB 所服务的是 WebEx 的实时网络会议系统，对系统的实时性要求很高，所以整个系统的技术难点在于 ：

（1）如何克服网络系统的滞后性（比如 DNS TTL 的问题）。
（2）如何克服网络传输的迟滞性（network latency，从而保证数据的快速复制和同步。
（3）如何保证在网络异常发生后数据的一致性。

为了克服这些问题，GSB 采用了以下的设计和技术手段 ：

（1）采用 Netscalar 的 GSLB 技术来实现 VIP 的快速切换。
（2）采用多线程 / 多进程技术来提高数据的扫描和传输速度。
（3）和 WebEx Meeting 应用开发团队合作，从应用逻辑的角度来保证数据的一致性。

经过若干年的实践，时间证明了这些是经济有效的手段。

1. 服务转移的设计

在 WebEx GSB 中，有多种服务，像传统的网络会议系统，服务转移时只要关闭一个 VIP 就可以实现。对于一个 cluster，只要在主备服务的 VIP 之间配置 GSLB，规定主服务“活”时，域名服务器永远返回主 VIP ；主服务“死”时，就返回备份服务的 VIP。

另外，在更复杂的 Web2.0 系统中，一个站点内包括多个子服务，它们协同起来才能提供一个完整的服务。考虑到数据间的耦合，以及跨数据中心访问的效率问题，WebEx 的做法是，把所有的子服务按照数据耦合的程度，分成若干个服务组，每个组内的所有子服务一起进行服务转移，也就是说，只要其中的一个服务坏掉，连同所有其他的子服务一起发生服务转移。在这样的云计算系统中，负载平衡器要实现一个逻辑与的功能，在思科 ACE 中，这可以通过设置 detect VIP 来实现，在 F5 中可以通过负载平衡器端的脚本语言 iRules 来实现。其工作原理如图 28-8 所示。

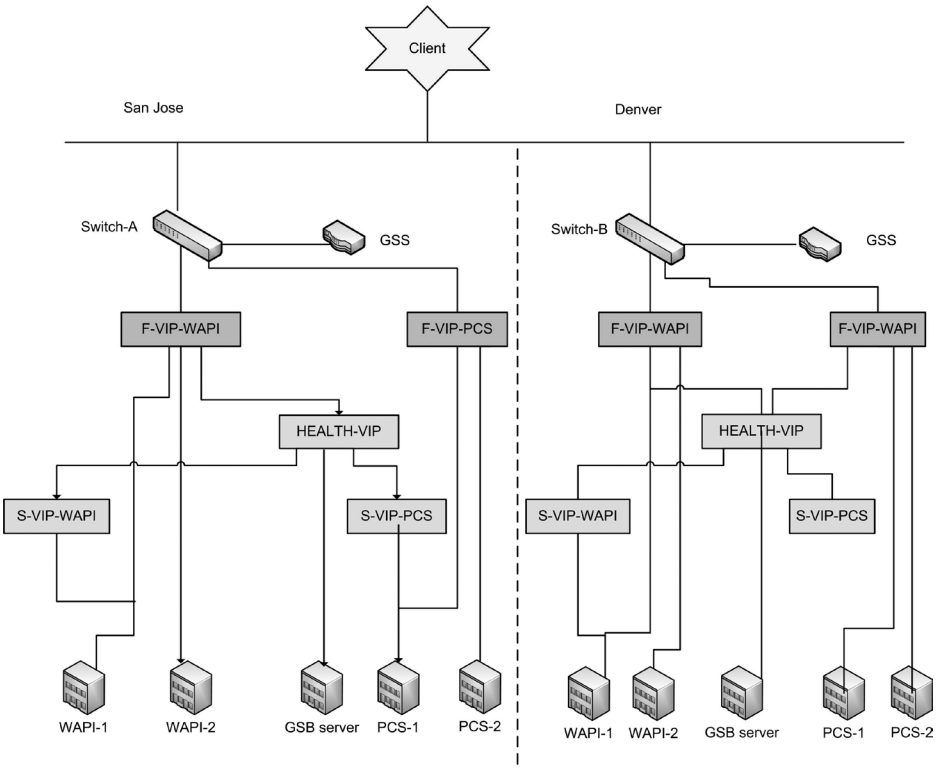


图28-8　负载平衡器服务转移逻辑与功能示意图

每个子服务有两个 VIP ：一个是静态 VIP（static VIP），另一个是浮动 VIP（floating VIP）。DNS name 指向浮动 VIP，静态 VIP 只有一个健康检查，指向每个服务器，只有所有的应用服务器全部死了，静态 VIP 才会死掉。浮动 VIP 有两个健康检查，一个指向每个服务器，另一个指向 GSB Failover Console（是一个控制台，其实就是一个 Apache Server 集群），两个健康检查之间的关系是逻辑与，也就是说，其中的任何一个返回失败的结果，浮动 VIP 就会死掉。

GSB Failover Console 是一个 HTTP Server 的集群，它仅仅提供一个手动服务转移的逻辑。当需要手动服务转移时，管理员只要改变一个标志文件的内容，就会影响这个健康检查的返回值，从而引起浮动 VIP 的失效。

2. 数据库复制的设计

WebEx 的大部分数据库都是 Oracle，所有的复制都采用 Quest Software 公司的 Shareplex，复制是实时双向的。数据库的复制设计主要考虑以下方面 ：

- 一对多或多对一地复制要避免级联复制，就是如果 A 和 B 之间有双向复制，C 和 B 之间也有实时双向的复制，但是要避免 A 和 C 之间的复制。这个可以通过用相同的 port 来避免（Shareplex 的一个特性）；
- 对一个表的部分数据进行复制，这个可以通过 Shareplex 的水平和竖直分割的特性来完成 ；
- 源和目标端的表结构不一致，这种情况主要发生在部署 database 的 patch 时，这个可以通过自己开发特殊的触发器来进行数据过滤，或者通过 Shareplex 的竖直分割的特性来完成 ；
- 冲突解决方案的设计，WebEx 的解决方案是通过时间戳来解决的。每一个要复制的表在设计时就一定要有个叫做 “ LastModifiedTime ” 的字段，最终如果数据发生冲突，就用新的记录覆盖旧的。具体的解决方案的设计要结合应用程序的业务逻辑要求来完成，不能一概而论。

3. 文件复制的设计

WebEx GSB 中对文件的复制也用服务的设计理念来模块化整个系统。结构如图 28 - 9 所示。

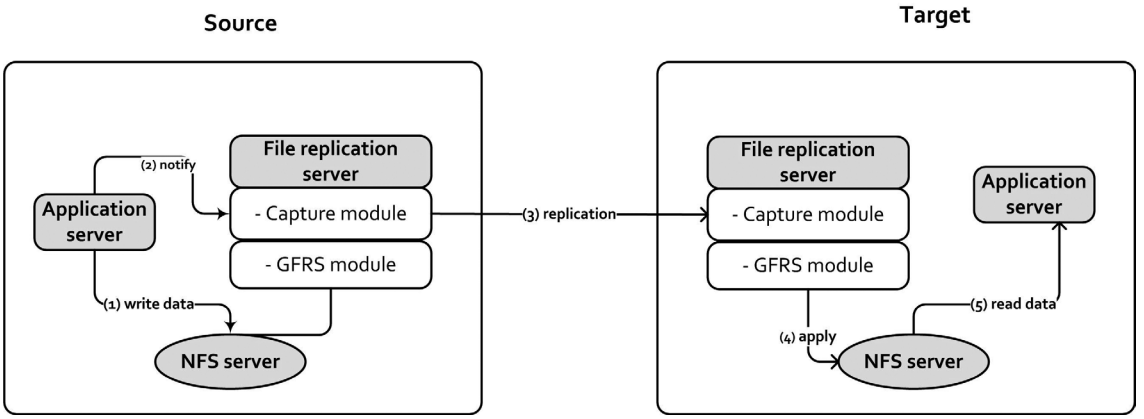


图28−9 WebEx GSB文件复制服务的设计

在源端和目标端，应用服务器和复制服务器都挂载到相同的 NFS 服务器，他们可以访问相同的数据。当源端的应用服务器产生新的文件或修改已有的文件后，通知复制服务器的 capture 模块，这个模块负责写 redo log 并且通知 gfrs 模块，然后 gfrs 模块根据文件路径从 NFS 服务器读文件并且通过 TCP 连接把文件传输到目标端，目标端的 gfrs 模块负责把接收到的文件写到 NFS 服务器上。这样当服务转移发生后，目标段的应用服务器就可以访问到相应的文件了。

4. 灾备触发的设计

灾备触发是基于整个系统的健康检查（health check）。 健康检查的设计主要考虑到自动服务转移和手动服务转移两个方面。对于简单 Web 服务系统，由于不存在多级的服务组成部分间的关联，一般就把自动服务转移和手动服务转移的健康检查合二为一，它的工作流程如图 28 - 10 所示。

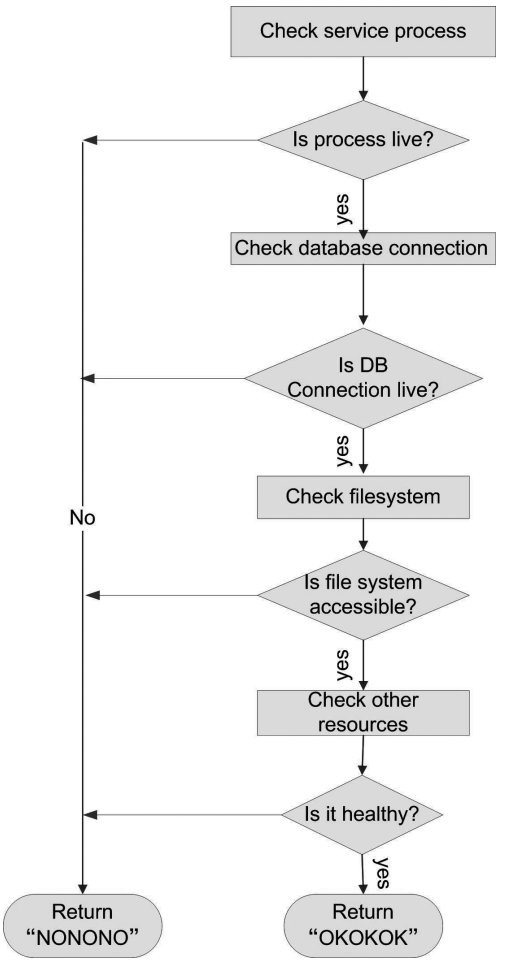


图28–10 服务健康检查方法

对于复杂的云计算系统，会有多个相对独立的子服务系统。在设计中，需要权衡考虑效率和灵活性。对于多个子服务系统的灾备触发设计可以有不同方案。

- 如果考虑到开发、部署的灵活性，可以把每一个子系统设计成独立的服务转移的系统，这样子系统间的耦合是通过应用程序的网络访问来进行（HTTP、SOAP 等）。这样每个子系统都可以单独进行服务转移，自由灵活，但缺点是子系统间的访问可能要跨区域，对于耦合相对紧密、访问频繁的子系统来说，效率会是个问题。
- 考虑到访问效率的问题，可以把耦合紧密的子系统绑定进行服务转移，这样可以保证他们永远在同一个数据中心，从而保证了访问的效率。但是要保证一个服务失败后，所有的服务一起进行服务转移，这需要在负载平衡器上进行一些特殊的配置。通常通过实现一个特殊的服务 ：GSLB Console 来完成。GSLB Console 的主要职责是 ：
- 负责监控同一组内的所有子服务的静态 VIP，并且实现逻辑与的关系 ；
- 提供特殊的手动服务转移的标志位，提供手动服务转移的接口 ；
- 提供一个同一的健康检查的 URL 给组内的所有子服务的浮动 VIP。

5. 控制台的设计

WebEx 总共有将近 100 个服务，实现一个集中的基于 Web GUI 的服务转移控制台（GSB Failover Console，GFC）是必要的。服务转移控制台的主要功能是 ：

- 提供统一的界面来浏览所有服务的状态 ；
- 提供统一的界面来简化手动服务转移的操作过程 ；
- 提供一个统一的监控平台，可以提供服务转移 mail 通知的功能，也可以对每一次的服务转移有一个详细的日志，提高系统的安全性。

28.5.4　项目回顾

这里总结一下 WebEx GSB 项目达成的目标 ：

（1）成功地减少了服务中断时间

在 WebEx 的历史上，曾经发生过整个城市光缆中断，整个数据中心的大型数据库（硬件系统故障）瘫痪，所有这些严重情况的发生，都没有造成 WebEx 网络会议系统的不可访问。原因是当这些事情发生时，系统自动发生服务转移，从而大大提高了客户的满意度。

（2）大大减少了产品维护升级对服务的影响

当需要对产品进行维护升级，对外的服务便被切换到另一个数据中心，然后进行维护和升级。之后，经过内部技术人员的测试后才能将服务切换回来，然后再进行另一个数据中心的产品维护升级。

（3）同时提高了产品的负载能力

因为一个 GSB 的服务群要服务于多个主服务群，所以 GSB 群的容量和负载能力比任何一个主群都要大。在 WebEx 的历史上，有多次客户的超大型会议都是被有意切换到 GSB 群上的。这样超大型的服务的质量得到保证，同时也达到了资产的充分利用。

28.6　本章总结

本章着重分析了业务连续性的对云计算提供商的挑战，以及为了应对这些挑战，云计算提供商要从管理上和技术上要做的一系列的工作。其中，灾备方案是业务连续性的一个重要的组成部分。通过一些原理的阐述和案件的研究，在这个方面做了比较详细的讨论。

参考文献：

[1] 业务连续性管理（BCM），百度文库，2012 http://wenku.baidu.com/view/b431f0cf89eb172ded63b726.html

[2] 业务连续性规划（BCP），中启航国际教育学院，2010 http://wenku.baidu.com/view/22294941be1e650e52ea9929.html

[3] Gary Liu，业务持续管理（BCM），2010, http://wenku.baidu.com/view/ce2252da50e2524de5187e38.html

第29章　云计算的合规性

29.1　IT合规概述

29.1.1　什么是IT合规

作为一个组织的合规工作中的重要组成成分，IT 合规是指组织在内部的流程中，依据外部或者内部标准作业规范确定组织的业务需求以及 IT 操作规程满足监管机构法律法规、行业公约内部或者外部的流程，组织或者外部机构通过这些流程实现以下内容 ：

- 确定是否满足相关需求，即 确定组织的业务和 IT 需求是否是依据组织的业务目标、外部监管机构或者行业组织相关法律法规、内部的相关策略标准等因素驱动。由业务目标、法律法规、用户合同、组织内部策略和标准或者其他因素驱动 ；
- 将策略、程序、过程以及系统付诸实施，以满足监管需求的需要 ；
- 确定相关内部策略、流程和指引在组织内部的各个环节和部门得到正确贯彻和正确执行。

29.1.2　IT合规对云计算提供商的必要性

审计与合规功能在外部监管机构检查一个公司法律法规符合性和评价一个公司内部 IT 流程有效性中总是发挥重大的作用。

由于 SaaS、Iaas 和 Paas 环境的动态性，为了确保客户的相关权益，审计和合规在云计算过程中变得更加重要。作为服务的提供者，云计算提供商必须在 IT 合规方面进行有效地控制和落实，以保证服务的安全性，完整性等各项要求。很多的企业用户会要求云计算提供商提出合规相关的证明和证据。如果无法证明相关的可靠性和安全性，这些企业客户将无法向他们自己的客户、行业监管机构证明组织内部业务运营的准确性和安全性。

例如某个支付组织使用云计算 IaaS 提供商提供的主机空间，则支付组织需要证明其使用的主机空间符合相关的支付卡行业信息安全标准 PCI DSS 相关要求。如果某上市组织使用了云计算提供商的 ERP 系统作为组织内部的 ERP 服务提供者，则上市组织需证明其服务提供商的 ERP 资源是安全、可信赖并且是正确的。

综上所述，IT 合规在云计算提供商的必要性和价值体现在以下几个方面 ：

- 确保组织内部各项操作基于业务需求、符合组织内部规章制度和流程，并符合相关监管机构和行业特性的合规要求 ；
- 确保所提供的服务满足客户正常工作需求的同时，对于客户的利益未造成实际的以及潜在的伤害 ；
- 协助客户证明其所使用的外部 IT 服务资源的可用性、安全性、保密性 ；
- 向潜在客户证明组织本身提供服务的合规性、安全性，增强客户对于云计算的信心，为组织发展新业务提供必要的安全证明。

合规标准的核心要求就是需要企业证明组织的信息化建设安全性。

29.1.3　互联网线上服务提供商在合规中面临的挑战

对于依靠创新和产品差异化来强化竞争能力的互联网服务提供商来说，快速反应市场和快速满足客户需求是服务提供商成功的法宝。但是，而这一法宝对于服务提供商内部的研发，技术运营和安全的挑战非常的高。

基于互联网提供线上服务的组织在这一方面要求比较传统的 IT 企业要求更多。而云计算提供商因服务的客户数比较传统互联网服务提供商更多，在可用性和安全性的要求方面比较传统的在线服务提供商更高。同时在线金融服务商在参考云计算提供商可用性要求同时，在服务准确性和完整性要求方面更加苛刻。而这导致在合规要求和压力而言，从低到高可以参考如下的表示 ：

一般的 IT 产品企业 → 服务提供商 → 云计算提供商 →在线金融服务商

在本章的研究中，会综合选取合规要求最高的在线金融服务商和云计算服务商来做详细的讨论，特别是在线金融服务商的实践。

根据实践的总结，服务提供商在满足安全合规的要求时面临着挑战主要是 ：

（1）合规带来的运营成本的增加

合规审计会带来对管理的规范要求。企业的运营成本会相应的增加。例如为了满足银联账户信息安全合规要求，企业需要将 POS 机具原本 HTTP 访问的 Web 程序修改为 HTTPS，这直接导致企业 POS 机具设备成本提高。同时为了合规而增加的流程，也是导致合规成本增加的最常见因素之一。

对于大部分的企业，运营成本的增加，是来自原有的运营体系中由于种种原因而存在大量的手工流程。由于这里流程的非自动化，在整个 IT 合规流程中，需要消耗大量的人力资源来满足合规的要求。这不仅增加成本，而且容易出错，可重复性差。例如为了符合 PCI 对于防火墙调整的变更管理需求，需要将每周进行的防火墙策略调整制定 EXCEL 电子表格和各种审批文档，并且要确保所有人员的一致执行。

运营成本增加的另一个原因是在自企业内部各个部分的边界问题。业务部门以及不同地区不一致的流程也将使得合规计划不成体系，从而导致如下所示的效率低下 ：

- 首先，整个机构中可能有多个部门对同样的应用控制进行测试和报告，这导致在审核准备上花费更多时间，并且内审人员需要花费更多时间来评判控制环境。审核时间的延长必然导致审核费用的增加。
- 其次，以不同的方式执行各项合规计划，这将使未在整体范围内协调一致的部门产生许多零碎的信息，这种效率低下将导致重复工作，从而导致多个部门对相同的 IT 环境控制进行测试和报告。例如某企业存在有一个业务系统，几个管理部门都具备后台管理功能和开发功能，由于平台的性质导致了需要审计团队对同一个平台进行多次测试和报告。

为了通过合规而投入的大量技术和 IT 资源会直接影响企业开展业务的能力。例如可能对技术部门原本应该上线的业务项目（如新的业务程序、新的合作伙伴接入、新的销售服务计划）造成延期 , 很多项目不得不延迟或者暂停。

（2）同时多项合规带来的运营管理复杂性增加

由于业务和监管的要求，企业可能要同时多项合规的要求。以国内在线的第三方支付机构为

例，可能需要同时遵从 PCI DSS、银联账户信息安全合规、ISO27001:2005 、SOX、SAS70、非金融机构支付服务管理办法等合规标准。

为了获取合规证明，组织需要努力满足各种规章要求。由此，企业要额外投入大量 IT 资源来证明本身能达到 IT 合规各项要求。例如为了满足《非金融机构支付服务管理办法实施细则》的第三十六条，非金融支付机构就需要建立必要的同机房数据备份设施和同城应用级备份设施，直接的后果是导致企业运营管理复杂性直线上升。

29.2　IT合规规划

29.2.1　IT合规整体框架

不同的合规有不同的侧重如行业标准、监管机构法律法规、用户特定的框架等。但在实际工作中审查这些需求点的时候，通常会发现他们之间存在相当多的交叉点。云计算提供商可通过整体性 IT 合规框架来处理这些需求。

云服务提供商在进行合规工作和实际作业过程中可以参考使用目前通用的合规标准和统一的策略方法，从而提高效率并满足合规性的要求。

例如在 ISO27001 中具备 13 个大类 39 个控制项 139 个控制点、PCI DSS v2.0 中存在六大类 12 个控制点，多于 220 个子要求。如果分别按照两个标准制定相关的管理制度和文档，组织将会制定大量重复性的规章制度。而如果组织定义统一的 IT 合规框架，使用通用的合规策略语言，将会大大降低组织内部在 IT 合规方面投入。图 29-1 是 IT 合规中常用到的标准。

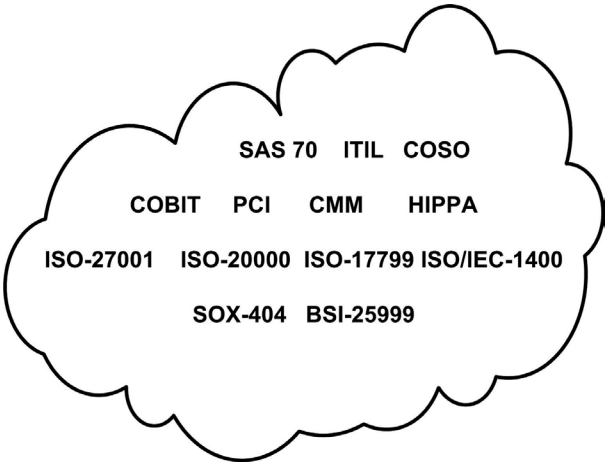


图29-1 IT合规整体框架

29.2.2　IT合规解决方案

在审计云计算对上述合规框架的合规性时，合规组织应首先需要明确两个方面 ：

- 组织内部审计部门对 IT 合规性工作的期望值是什么 ；
- 外部客户或者监管机构审计方对于合规性的期望值是什么。

云计算提供商如需进行 IT 合规工作，可参考下列步骤进行 ：

1. 明确需求

在组织进行 IT 合规工作中，明确来自两个方面的需求 ：组织内部的业务需求和目标客户的行业特性需求。

不同的目标客户群体拥有不同的行业特性，而行业特性不同将会导致潜在客户关注的焦点不同，例如支付行业关注的是持卡人的信息，而在医院的行业中更加关注的病人隐私信息安全。关注的焦点不一致将导致采取的合规标准侧重点不一致。

2. 确定合规的标准和规范，并和外部监管机构进行沟通

在明确了需求后，组织应确定要采取的合规技术标准和规范。为了明确监管机构的关注点和监管要求，组织要在进行合规前期邀请外部监管机构进行必要前期沟通。

在通常的合规项目中，如果前期未和监管机构沟通，往往会发现组织内部对合规标准的理解和监管机构的关注点有较大的差距，而这些差距将会导致组织在合规工作准备中出现部分偏差，需在后期进行额外投入资源进行补救工作。

3. 定义清晰合规的范围

在进行合规工作中，最重要的一点是明确定义企业合规的范围。

例如在支付卡信息安全标准的合规中，到底办公环境的物理安全是否属于合规范围之内，需要企业进行 PCI 标准详细的分析和理解，确定企业的办公环境是否数据涉及持卡人的数据环境。例如企业如果所有运维操作均在持卡人环境中进行，而后台持卡人业务清算和风险控制处理在办公环境中进行，则需将把办公环境纳入合规范围以内，若清算和风险控制处理均单独在持卡人环境中，并未涉及办公环境，则可将办公环境从持卡人合规范围内剔除出来，从而减少 PCI 的合规范围。

4. 梳理企业内部操作流程和确定差距

在组织进行合规操作中，建议服务提供商将组织内部的各项操作流程和环境与审计合规标准进行比对，梳理组织内部实际操作步骤和合规标准的差距。

在这个过程中，企业应将实际内部运营操作（而非文档中的操作规范）、以及将产生的日志和记录的部分与合规标准进行比对。因为在合规和审计工作中虽然看重组织的操作规范，但作为有经验的合规监管机构更看重的是组织日常操作过程中的记录和日志。

5. 依据差距确定整改计划及执行整改

组织应该根据组织内部自我评估的结果来设计合理的合规整改项目，包括条款清单，风险等级和项目优先级。通常在整改项目的实施的过程中，各涉及部门对整改项目的支持力度是项目成功的关键因素，同时也是项目风险最大的地方。

在国内发生过一个真实的案例，国内某电子商务组织中，在实际业务需求中存放了持卡人卡号的 CVV（Card Verification Value，信用卡校验值）信息，以方便持卡人进行二次支付时免输入持卡人相关信用卡账户信息。这虽然增加了客户支付的快捷性，但同时增加了客户信用卡泄露的风险。依照 PCI DSS 的相关要求，支付机构或者商户不得存放持卡人的 CVV 信息。组织在进行

PCI DSS 合规性工作，将此项目列入了整改清单，但因业务部门不同意进行此项整改，导致组织无法通过 PCI 主任审核员的现场审核检查工作，无法获得 PCI DSS 合规报告，导致整改项目最终失败。

6. 整改完毕后，依据合规标准再次进行合规

在安全整改完成后，组织可以向外部监管机构或者组织进行合规申请工作。同时，组织应该将重点转移到整改项目的各项实际运行，保证有实际运行记录来确定支持合规的审计。

29.3　IT合规实践

29.3.1　IT合规工作内容

IT 合规的内容中通常组要包含三个方面的内容 ：

（1）财务合规中的 IT 合规。
（2）IT 服务标准化的合规。
（3）信息安全管理体系。

行业性的合规性工作的框架和标准通常都是以这三方面为出发点，再基于行业特点或者关注重心的内容不同而有所区别。组织可按照这三个方面的安全框架组建组织 IT 合规工作内容。这三方面主要的合规标准对比简介如表 29 - 1。

表29-1　SAS70、ISO27001、ISO20000合规标准对比分析

考虑主题	SAS 70（Ⅱ型）	ISO27001认证	ISO20000认证
报告的预期对象	限于服务机构、服务机构用户及审计员。该报告不适合非用户群体	任何系统的利益相关方面，虽然机构通常公布其认证情况，但限制发布相关的支持细节	IT服务系统的使用者
预期目的	为客户审计员提供服务机构的控制信息，而这些控制信息可能影响客户机构的财务报表中的论断	为机构准备合理的信息安全管理体系提供证明	为机构准备合理的信息技术服务管理提供证明
主要内容范围	重点在于服务机构影响客户机构的财务报表的系统和控制	信息安全管理标准ISO27001中涉及的11个大类相关的内部控制	主要针对IT服务的交互和支持，强调运行维护阶段的内容
报告性质	在关于服务机构的控制是否称述得当、控制措施是否设计合理，以及控制措施的有效性这些方面提供保证	在机构准备合理的信息安全管理体系提供第三方保证	在机构准备合理的信息技术服务管理提供第三方证明
报告的可比性	没有用于每个项目的定制控制目标和标准	对组织内部信息安全管理使用统一的标准，但组织在适用性声明中选择性要求具体控制点适用与否	对组织提供的IT服务水平进行统一的评比
审计频率	通常每6个月或者12个月	认证有效期三年，每年进行监督审核	认证有效期三年，每年进行监督审核
业务连续性范围	不包括在内	作为信息安全管理体系标准控制目标组成部分包含在管理体系中	业务连续性只是ISO20000中的一个流程
认证办理机构	国际四大审计机构出具报告	建议寻找中国信息安全认证中心	建议寻找中国信息安全认证中心

29.3.2　IT合规的实践建议

IT 审计往往是公司上市财务审计的组成部分。如果组织准备在美国上市或者已经上市，上市法规会要求公司进行独立的外部审计，确保上市组织的内控措施是真实有效的，例如遵从 SOX404 审计，审计公司将会出具一份具有真实公正的财务报表的年度报告来证明组织的财务报表真实有效。

为了证明组织的内部控制措施有效性，外部审计人员将执行一些具体的审计测试，这些测试中，审计人员事实上是在测试系统中数据的有效性，为了进行这种价值型判断，审计人员会对 IT 控制进行检查，包括一般性控制检测和应用程序控制检测。

审计人员将在最基本的层次上进行检查，检查的内容和 ISO27001 非常接近，例如包括访问控制、变更控制、业务连续性检查，同时会有一些测试（比如 Linux 系统和 Windows 的操作系统层中的帐号控制和密码组成的策略），但是大多数的检查都是基于访谈和证据。

最后外部审计人员生成一份在事实准确性方面获得被审计方同意的管理建议书。管理建议书是纯信息性的（for information only），只有最严重的控制失败性问题才会进入到年度报告中。当然作为一个组织内部的信息化建设部门，通常而言这也是组织的被审计部门获得预算的好机会。

为了应对 IT 合规的挑战，组织应该尽可能减少信息的无序管理，同时实现审核流程和 IT 安全控制的自动化，以降低劳动力和咨询成本。组织还尝试提高内部审核的频率，以维持来之不易的合规成果。当然组织最终的目标都是以更低的成本满足合规的要求，使得组织可以将 IT 资源分配到可以更直接的创造收入的业务的开发也运营工作中。

由于大部分的法规都比较模糊，因此在确立 IT 要求指南并将这些要求对应到特定的组织策略和控制目标时，组织必须自己制定出相应的措施。下面针对某些具体的 IT 合规进行简单介绍，并对如何进行合规进行说明。

1. 如何通过财务审计中的 IT 审计

（1）确定 IT 审计的范围

作为财务审计组成部分的 IT 审计范围一般由财务报表中的重大财务账户来定义出被审计的商业流程和交易类别。作为支撑这些商业流程和交易类别的信息系统都属于 IT 审计的范围，包括 IT 基础设施服务和财务应用程序。

通常审计人员对这些范围都进行两部分审计测试 ：

- IT 一般性控制审计包括的内容有程序开发、程序变更、程序操作、访问控制、控制环境。
- 应用程序控制审计主要关注于应用程序和外部接口之间的输入控制、校验控制、系统接口、系统计算、权限控制等方面。

（2）IT 审计准备的材料

- 变更管理方面 ：变更的整个过程是否有人审批过程并被监控和测试，同时是否存在职责利益冲突 ；
- 访问控制方面 ：系统帐号是否有授权并权限合理，系统是否设置合理的安全参数，特权

帐号是否设置的合理，访问的过程是否有日志记录 ；

- IT 操作方面 ：各项业务数据是否定期备份和测试，自动化脚本是否有合适的授权，事故是否被定期统计和分析。常见的例子是，开发人员为了脚本的执行方便，常使用 root 的权限，造成授权过大。

2. 如何通过 SAS70 安全认证

所谓的 SAS70 即 Statement on Auditing Standard 70（审计标准公告第 70 号），是由美国职业会计师协会 AICPA（American Institute of Certified Public Accountants）制定的，针对服务机构的内部控制、安全保障、审计监督措施的审计标准。

SAS70 有两种类型 ：SAS70 类型 I 报告和更高级的 SAS70 类型 II 报告。公司通常先执行类型 I ，然后进入到类型 II 。

在类型 I 的报告中，审计人员将会 ：判断控制目前是否像组织本身所描述的那样已经生效 ；判断生效的控制是否满足已声明的控制目标。

在类型 II 的报告中包含类型 I 的各个部分，再加上一个关于控制都已经被测试过、并且能够提供合理保证的声明。报告中还应记录一段时间内控制措施的测试详情。

在 SAS70 的报告中重点对服务提供商所提供服务的业务流程、应用程序开发和维护、变更控制、逻辑访问安全、物理安全、IT 操作、业务连续性、网络和通信等方面进行控制串行测试。

通常一个组织如果有信心通过 SOX，那么他也应该有信心通过 SAS70，因为两者的本质是一样的。下面将 SAS70 项目实施的一般性步骤介绍如下 ：

第一阶段 ：项目启动

- 确定组织内部提供服务的组织结构 ；
- 与外部审计机构进行沟通，了解其关注点 ；
- 确定服务接受者财务报告的编制流程 ；
- 确定相关的业务流程及 IT 支撑系统 ；
- 确定与服务质量密切相关的业务流程。

（要注意 SAS70 审计可能不仅涉及到一个服务接受者，需要遵循的法规，可能也不止一个，在进行范围确定时要一并考虑）。

第二阶段 ：风险评估并识别控制活动

- 明确控制目标并采用成熟的控制框架，如 ISO 27001:2005、CoBIT、COSO 等 ；
- 识别业务流程及应用控制相关活动，和 IT 运行相关的人、技术、环境相关控制措施 ；
- 识别和标准有距离服务使用者存在的补偿性控制措施。

第三阶段 ：差距分析

针对控制目标和实际现状进行差距分析，确认重大缺失和缺陷。

第四阶段 ：体系设计和试运行

- 在差距分析的基础上，根据控制目标及风险分析的结果，对内控体系进行优化设计 ；

· 对新的内控体系进行人员培训并进行试运行 ；
· 在新的内控体系试运行期间，收集各相关部分的反馈并进行内控体系的完善。

第五阶段 ：内控措施的评估和测试

针对新的内控体系运行情况，进行内控体系的有效性测量，并进行内控体系的自我评估，确认控制体系满足相关控制目标要求。

第六阶段 ：持续改进

针对内控体系的运行情况，定期进行内控体系的风险评估工作，并根据有效性测量的结果实施持续改进计划。

3. 如何通过 PCI DSS 安全认证

PCI DSS（Payment Card Industry，Data Security Standard）中文名称是支付卡行业信息安全标准，是支付卡行业安全标准委员会 PCI SSC 负责在全球范围内推行的支付卡行业数据安全标准。目前最新版本为 PCI DSS v2.0 。

支付卡行业要求在 2012 年内支付行业都应该采用 PCI DSS v2.0 版本进行支付行业数据安全标准认证。PCI 的数据安全标准旨在保证和进一步提高信用卡持有人的数据安全，该标准使得全球支付行业采取同样的数据安全标准。该标准以 PCI 数据安全标准中的 12 条基本要求作为基础。在这套安全测试工具中同时还加入了相关测试步骤。12 条 PCI 数据安全标准要求的概要描述如表 29 - 2 所示。

表29-2 PCI数据安全标准概要

控制目标	PCI DSS 需求
建立并维护安全对网络	安装并且维护防火墙以保护持卡人信息
	避免使用供应商提供的默认系统口令和其他安全参数
保护持卡人数据	保护存储的持卡人数据
	加密开放/公共网络上的持卡人数据传输
维护一个弱点管理程序	使用定期升级的防病毒软件或计算机程序
	开发并维护安全的系统和应用
实施强有力的安全访问控制措施	根据业务需要限制对持卡人数据的访问
	为每一个具有计算机访问权限的用户分配唯一的ID
	限制对于持卡人数据的物理访问
定期监视并测试网络	追踪并监控对网络资源和持卡人数据的所有访问
	定期测试安全系统和流程
维护一个信息安全策略	维护一个信息安全策略

PCI DSS 安全要求适用于所有系统组件。系统组件可以是持卡人数据环境中或与之相连的任何网络组件、服务器和应用等。持卡人数据环境是处理持卡人数据或敏感认证信息的网络部分。网络组件包括（但不限于）防火墙、交换机、路由器、无线接入点、网络设备和其他安全设备等。服务器类型包含但不仅限于 ：Web、数据库、认证、邮件、代理服务器、网络时间协议（NTP）和域名服务（DNS）等服务器。应用是指所有外部购买的和定制的应用，包括内部和外部（如 Internet）应用。

如果组织需要通过 PCI DSS 认证，那么组织需要准备好下列材料 ：

（1）最新的网络结构图 ：

根据 PCI DSS 要求，应实施隔离保护持卡人数据，要求包括 ：

- 将需要外部访问的 Web 服务器和邮件服务器等放到隔离区（DMZ）中 ；
- 数据库服务器放到内网，只允许必要的访问通过 ；
- 不允许内网服务器通过互联网访问 DMZ 区 ；
- 实施状态检查，动态包过滤 ；
- 使用 NAT 等技术实施 IP 伪装。

（2）针对防火墙和路由器等网络设备的安全策略 [1] ：

- 配置的测试和审批 ；
- 描述网络组件的组、角色和职责等 ；
- 开启的服务、协议和端口清单和说明（并简要阐明为什么必须开启该项）；
- 要求每半年审核防火墙和路由器规则。

（3）针对（访问支付环境的）个人计算机的安全策略和流程 ：

- 安装个人防火墙、防病毒等防护软件，并定期更新补丁和库信息 ；
- 所有非 console 的管理访问需要加密，如 SSH、VPN 或 SSL/TLS。

（4）针对服务器的安全策略和流程 ：

- 采用最新的补丁程序，并进行脆弱性管理 ；
- 每个服务器只实现一个主要功能，如 Web 服务器、应用服务器、数据库服务器 ；
- 去除所有不必要的服务和协议 ；
- 去除所有不必要的功能，如脚本、驱动、特性、子系统、文件系统和不必要的服务 ；
- 安装防病毒和恶意软件防护软件，并确保采用最新的补丁和病毒库。

（5）支付环境获得和分配正确网络时钟的过程。

（6）内外部网络脆弱性扫描和渗透测试流程和记录（记录包括 ASV 和渗透测试报告）。

（7）部署入侵检测 / 入侵防护系统。

（8）使用文件完整性监控软件，警告对重要系统文件、配置文件或者内容文件的未经授权的改动，并每周进行文件比对。

（9）（包括系统和网络设备的）安全日志审核的安全策略和流程，以及安全策略和审计日志保留策略。

（10）公司保留和处理数据的策略和流程，包括卡信息的保护、屏蔽显示、纸质和电子媒介保存等。请注意 PCI DSS 只允许存储主账号 PAN、有效期、持卡人姓名、服务代码（存储时必须采用保护机制）；不允许存储完整的磁条数据、CAV2/CVC2/CVV2/CID、Pin/Pin block，即使加密存储也不允许。

（11）流程化的介质销毁策略。

（12）文档化描述用于保护存储数据的密码系统，包括厂商、密码系统类型，以及加密算法。

（13）详细说明密钥管理过程和流程 ：

- 涉及密钥包括使用的密码算法如 3DES 的密钥，管理员密钥等 ；

- 需要包括密钥生成、安全分发、安全存储、周期性变更（PCI 要求至少一年）、维护旧的和不安全密钥列表且保证不再使用这些、分离 Key 的双重控制、禁止未授权的密钥置换、密钥管理员职责并由其签署（表明其理解并接受这些职责）。

（14）策略阐明所有默认设置必须经过改变，比如密码、SNMP 字符、去除不必要帐户。

（15）书面的软件开发过程，确保其基于产业标准，并且在整个生命周期中关注安全问题。包括但不限于 ：

- 生产环境和开发测试环境必须分离 ；
- 生产数据不能用于测试和开发 ；
- 对于 Web 应用的审核应参考业界安全代码指导，如 OWASP ；
- 对于 Web 应用，应具有策略应对已知的威胁和脆弱性，可采用如下方法的任何一个 ：
  - 周期性审核（至少一年一次）Web 应用的脆弱性 ；
  - 安装 Web 应用防火墙。

（16）书面安全开发策略，确保要求代码审核。

（17）安全访问控制 ：

- 针对系统用户的访问控制 ；
- 针对系统组件运维人员的访问控制，如 Web server、数据库、网络设备等 ；
- 对于远程方式访问的管理员和员工，应具有双因素验证机制 ；
- 对于管理员和员工的访问密码和 ID 每个人必须唯一 ；至少 90 天修改密码 ；密码长度至少 7 个字符 ；密码必须包括字母和数字 ；不能使用上 4 次的密码 ；
- 物理访问控制。如果是采用主机托管机房，请提供 SLA ；并查看是否达到 PCI DSS 附录 A 对于主机提供商的要求。

（18）风险评估策略和流程（至少每年执行一次风险评估）。

（19）风险评估报告，信息资产清单（包括物理资产、信息资产和服务类资产等），特别地应整理出支付环境相关的允许安装的软硬件列表。

（20）人力资源方面 ：

- 员工入职和离职流程（包括背景调查，以及 NDA 签署等）；
- 员工 NDA 记录（所有具有权限访问支付系统数据的员工）；
- 背景调查记录（所有具有权限访问支付系统数据的员工）；
- 培训和意识教育流程及其相关记录 ；
- 职责角色分配记录（如研发人员、运维人员、测试人员、客服人员、源代码审核者等）。

（21）所合作的服务提供商和商户列表，并监控他们的 PCI DSS 符合性状态。

（22）应急响应和处理流程（PCI DSS 要求 12.9 是基线要求）。

（23）支付业务描述（要提供英文，将会呈现在证书报告中）。

4. 建立信息安全管理体系 ISO27001

首先说明信息安全管理体系 ISO27001 的认证不是一个强制性的合规认证，所有的组织如果想要通过这个认证，从动力而言应该是组织内部本身的期望。

ISO27001:2005 是全球范围内被认可的有关信息安全管理的国际标准。最初源于英国标准 BS7799，经过十年的不断改版，终于在 2005 年被国际标准化组织（ISO）转化为正式的国际标准，于 2005 年 10 月 15 日发布为 ISO/IEC 27001:2005。该标准可用于组织的信息安全管理体系的建立和实施，保障组织的信息安全，采用 PDCA（Plan - Do - Check - Action）过程方法，基于风险评估的风险管理理念，全面系统地持续改进组织的安全管理。 其正式名称为 ：《ISO/IEC 27001:2005 信息技术 - 安全技术 - 信息安全管理体系 - 要求》。

比如，某个在线金融公司在建立信息安全管理系统（ISMS: Information Security Management System）体系前，各项合规性认证工作需要建立许多具有重复维度的信息安全标准内容。在建立 ISMS 体系以后，由于 ISMS 体系覆盖了组织安全、人力安全、资产安全、技术安全、研发和运维安全、合规性管理、事件管理、业务连续性等内容，从合规性角度而言，组织只需针对其他各项安全标准进行部分内容的细化，从而使得合规性的工作内容大大降低。

由于信息安全管理体系建设可在互联网中找到大量相关文章，所以本文不再做更详细的介绍。在国内如果组织期望通过 ISO27001:2005，那么组织需要按照下列步骤建立信息安全管理体系 ：

（1）定义信息安全管理体系 ISMS（Information Security Management System）的范围和边界，形成 ISMS 的范围文件 ；

（2）定义 ISMS 方针（包括建立风险评价的准则等）并形成 ISMS 方针文件 ；

（3）定义组织的风险评估方法 ；

（4）识别要保护的信息资产的风险，包括识别 ：

①资产及其责任人 ；

②资产所面临的威胁 ；

③组织的脆弱点 ；

④资产保密性、完整性和可用性的丧失造成的影响。

（5）分析和评价安全风险，形成《风险评估报告》文件，包括要保护的信息资产清单 ；

（6）识别和评价风险处理的可选措施，形成《风险处理计划》文件 ；

（7）根据风险处理计划，选择风险处理控制目标和控制措施，形成相关的文件 ；

（8）管理者正式批准所有残余风险 ；

（9）管理者授权 ISMS 的实施和运行 ；

（10）准备适用性声明。

在 ISMS 体系建立完毕，并试运行一段时间后（建议半年），国内的组织可以和中国信息安全认证中心联系进行认证工作。

5. 应对合规审计人员的技巧

在各项外部合规性现场检查的过程中，组织内部的负责人员在对外开展接待工作时应该讲究一些技巧性。下面是其在应对外部合规人员审计过程中的部分建议 ：

- 安排内部人员全程接待。外部合规人员的所有工作应该在内部接待人员的陪同下进行，尽量安排对组织内部实际情况和业务熟悉的人员进行接待 ；
- 有一说一。对于审计人员的提问，内部人员应按照工作职责所知原则回答问题，同时回

答问题范围因只针对和本职相关的问题进行沟通 ；

- 积极参与到合规检查工作中去。提前和检查人员沟通相关合规检查范围和工作计划，知道检查人员的期望，确保有足够的时间进行准备 ；
- 主导和帮助合规审计人员。为审计人员安排预约的会见人员和会见时间，从而避免合规审计人员的盲目需求造成对组织内部人员的干扰 ；
- 审计证据。永远不要为审计人员制造审计证据 ；
- 保留复制。所有提供给合规检查人员的文档和证据都应该保存一份 ；
- 合理的办公区域。给外部合规检查人员一个单独的空间，尽量不要和内部员工在一起 ；
- 不人为设置障碍。在现场合规检查过程中，对于应该向合规检查人员开放的地方，应及时合理的开放。对于不方便开放的区域应及时和检查人员说明情况 ；
- 检查合规报告草案。组织内部项目负责人应对合规检查草案中的错误内容或者持有异议的内容提出建议和想法。

29.4　合规工作中的难点和解决思路

在合规工作中，各个组织因组织信息化建设水平和管理水平的不一致，在组织合规工作中遇到的难点各不相同。这里是在实践中碰到的一些问题和解决思路，以供参考。

29.4.1　公司的战略与支持

（1）公司的决策机制

一个 IT 合规项目的成功首先要确定其是否与公司的业务战略契合。公司业务战略的需求是一个 IT 合规项目是否需要的根本动因。

如果一个项目和组织的业务战略不契合，项目本身的存在性都不合理，那么这个项目肯定是一个失败的项目。而判断 IT 合规项目是否符合组织的业务战略目标，这需要合适的决策人员来进行判断。

解决思路 ：

IT 合规项目一定要依靠组织内控部门或者内部合规部门来推动，同时将业务部门的负责人或者代表纳入项目中。只有业务部门明确了合规的意义，业务部门才会积极参与合规整改相关内容。所以只有将业务部门纳入参与到合规项目的决策机制中，业务部门才会以部门内部项目的形式参与 IT 合规项目。

（2）组织没有足够的资源支持项目

合规项目的成功，很大的程度在于依据合规的目标进行组织内部资源的整合、整改。而如果 IT 项目没有足够的人力资源、组织资源来进行项目支持，只是 IT 部门在 IT 层面进行推动的话，项目往往会在中途停滞。特别是对于业务部门需要配合的地方，通常情况下 IT 部门很难推动业务部门的整改。如果牵涉到技术整改的话，还往往需要资金资源进行保障，来确保技术整改的落地，而如果从公司管理决策层面不进行明确和支持的话，通常 IT 部门的资金需求往往很难保障。

解决思路：

在项目实施的初期，一定要将项目实施的资源需求明确，同时在项目启动之前将预算提交。因为不能保障项目过程中会发生什么事情，如果不提前预计相关费用，项目过程中发生的相关费用将无法保证。

29.4.2　IT管理

（1）组织 IT 规范性制度不齐全

在互联网组织中，特别是在创业型组织中，通常都存在重操作轻规范的缺点，组织相关人员通常都依靠运维人员的个人经验或者组织内部的人员默契进行相关规定的执行和操作。这样做的好处在组织小的时候可以方便地实施项目，而这也往往是在 IT 合规中最容易出现的问题。

解决思路：

所有的合规工作中，外部检查单位最看重的，也是第一眼的印象的，就是组织的规章制度。所以在合规工作中，组织的规章制度一定要齐全，但组织不要为解决合规而制造不切合实际的大而全的规章制度。因为通常这些制度文档都无法在组织内部落地。组织在制作内部规范的时候，要积极依照组织内部实际运行情况撰写组织规章制度，同时参考合规标准的内容，将合规的内容和组织运作实际结合。

例如关于密码有效期的问题，组织可以通过自动化程序实现账户相关的所有安全问题，那么组织在制度中就只需要说明通过系统某某程序进行控制就可以了。

（2）IT 项目过程内部参与人员无 IT 技术背景

许多组织在实际运作过程中，部分业务系统的需求提出、采购、建设过程中往往由业务部门自行主导。例如财务部门提出人力资源管理系统，在项目建设的整个过程中由财务部门自行组建项目团队及实施。这种模式的好处在于产品的各项业务功能均由使用部门自行确认，业务功能模块和实际组织运作的贴合度高。但从进行组织内部 IT 合规的角度，如果前期项目建设过程中没有内部合规部门的参与，项目建设中往往缺少很多合规所需要的支撑性的证明材料，或者部分功能无法满足合规的要求，而这些功能模块往往不是业务部门关注的重点。同时项目建设证明性材料也不是业务部门关注的重点。

解决思路：

在业务部门主导的项目中，系统运维部门或者内部控制部门一定要在项目的推广前期主动介入，了解相关情况和提出各项需求。以方便系统的建设过程中满足相关需求，因为项目功能的变更计划提得越早，项目的改动成本也就越少。与其在项目结束后提出需求导致使用和实施部门的挫折，还不如及早主动介入。

29.4.3　技术运营团队的工作

在服务提供商的组织功能中，技术运营（Technical Operations）是至关重要的。他们负责

服务的 7*24 的生产线运营，包括平台构建、升级、日常运营等。

在这个功能上，常见的问题是 ：

（1）研发人员即运维人员

更快地将产品导入市场、更快地响应客户需求，这是许多组织制胜的法宝。而为了达到这个目标，创业初期，组织的研发人员往往就是生产线的运维人员，或是技术运营人员。研发人员可以在生产环境直接进行代码的修改。这从程序开发维护和生产线运营的角度来看，绝对是致命的打击。

举个真实发生的案例，某个组织的技术运营人员某日在进行日常操作的过程中偶然发现系统自动化任务中发现一个定时执行任务的批处理任务。而这个处理任务不在其定时任务记录清单中。于是排查分析自动批处理任务，最后发现这个脚本是研发人员为了在家处理故障，将系统故障信息自动发送到其个人邮箱，而这个研发人员却早已离职。

解决思路 ：

无论什么时候研发人员都应该禁止在生产环境上进行直接操作，即使是为了快速解决问题也不可以。所有的生产线变更和事故处理都应该由技术运营工程师进行操作，并且对所有操作应进行记录。记录的方式有很多种，例如部署运维堡垒机、Linux 下将 histrory 的日志定向到日志服务器中、运维人员使用的操作终端强制记录操作命令等方式。

（2）审批缺少运营自动化平台

前段时间对国内部分市场占有率比较高的四家电子商务平台组织进行信息安全合规的检查，发现拜访的组织均没有组织内部的运营自动化平台，如变更管理系统。内部的审批依靠口头或者 email 来往进行确认。这在进行合规性一般性控制测试工作中，属于审批无法确认的缺失，合规审核机构可以据此判断组织的内部一般性控制措施失败。

解决思路 ：

作为临时的方案，在缺失自动化审批平台的时候，组织应尽量明确一种标准的审批模式进行审批确认，例如邮件或者纸资材料。所有的审批记录应进行保存和维护。如果组织使用邮件进行审批确认，建议组织可在邮件系统中开通审计邮件，将审计邮件账户添加到工作邮件组中。这样可确保所有发往工作邮件组的邮件被记录。但这样也会导致一个新的控制需求，对于所有审批邮件需存档备份，并定期进行恢复性测试。

作为长期方案，组织或公司内部一定要建立运营自动化平台来涵盖变更管理，事故管理等重要的服务运营管理流程。

29.5　案例研究：在线支付服务的合规性

X 公司是国内最大的在线的第三方支付（业内称为 ：非金融支付，是指银行金融机构之外的支付服务）企业之一，包括提供外币卡在线支付服务。在运营中，因涉及持卡相关信息的处理，企业所服务的国外商户非常关注企业对于持卡人信息的安全保护措施。企业因此企业需要证明持

卡人信息在企业运营系统中的状态和存放是安全可控的，为此企业在 2009 年决定启动 PCI DSS 认证，以证明企业的信息安全处理合规性。

这个案例介绍了四年中，企业在合规认证上所做的实际的具体的工作。在案例的讨论中，可以看到企业在不同阶段所遇到的问题，挑战，以及整改方案。同时，也可以看到，在这四年中，企业的安全管理的成熟度也在不断的提高。

29.5.1　第一年信息安全整改实践

1. 企业信息安全状态

在 2009 年，企业首次邀请 PCI 的质量体系评审机构（QSA，Quality System Auditing）进行符合型审核。之前，企业进行了必要的自我差距分析，根据自我分析结果，确定发企业信息安全管理工作属于信息安全建设的初级阶段。

（注 ：信息安全建设初级阶段具备如下特征 ：所有信息安全动作由员工个体进行基于自我意思驱动，组织不具备必要的信息安全组织结构和规划，整体信息安全管理杂乱无章。具体的运营管理成熟度。详细请见技术运营部分的第 10 章 “ 生产运营综述 ”）

在自查过程中发现了如下的问题 ：

- 企业尚未建立信息安全管理制度，开发人员对于持卡人信息的保护只是基于自我安全开发意识的驱动，对信用卡敏感信息进行了简单的 DES 加密。
- 企业已经建立了简单代码开发过程管理制度，但为了快速的响应生产事故处理速度，企业研发人员负责生产环境的代码上线和日常维护。
- 企业办公网络和生产网络未有效隔离，企业处理持卡人信息所在的办公网络环境和生产环境进行了简单的安全隔离，但和企业内部的各部门通信未进行有效地防护隔离。
- 企业在进行信息安全建设过程中仅基于企业信息安全工程师的个人意见进行信息安全规划和建设，从未进行过信息安全整体风险评估工作。

2. 信息安全整改内容

为了通过首次 PCI QSA 审核，公司成立了专项合规整改项目组，项目组负责人为业务部门负责人（注 ：此项目驱动力是来至业务端客户的合规需求），项目组成员包涵系统运维部信息安全经理、产品经理、产品研发经理、HR 部门人事专员。

项目以专项合规整改项目方式，针对所暴露出来和各项合规标准差距进行整改。整改措施分为三个部分，组织结构的调整、参考 PCI DSS 标准进行信息安全技术差距的整改、规章制度的修订和发布。这三部分的主要工作内容为 ：

- 依据 PCI DSS 规范要求制定并发布各类信息安全规则制度。
- 内部组织调整，调整内容为 ：以研发经理为目标责任人，强制要求在一个月之类讲系统运维和研发文档标准化，将产品系统图文档化为起点，要求在研发在旁的情况下，由系统运维部进行的上线。
- 人力资源部门负责对所有涉及持卡的信息处理的员工进行背景调查、签订保密协议、进行信息安全意识培训等员工入职前信息安全工作。

经过为期三个月的专项整改，企业针对 PCI 所需遵循的各项安全要求建立了企业的基本信息安全管理体系，并依据 PCI 合规标准对技术差距点进行了改进。

3. 信息安全整改结果

经过三个月的信息安全整改后，企业通过了 PCI 安全合规首次现场检查。

29.5.2　第二年信息安全整改实践

1. 企业信息安全状态

2010 年，为了确保企业通过 PCI 的年度再审核，企业由系统运维部信息安全经理（注 ：企业已经有了合规管理部，但因为涉及信息技术，所以管理层决定由参与了上一年度 PCI 认证的信息安全经理负责相关的合规工作）担任 PCI 合规项目负责人，继续负责 PCI 年度审核工作。外部 QSA 审核人员进场后的前三个月，项目负责人对项目体系运转情况进行了再次复查。

经过内部的自我差距分析，确认企业已经建立了完整的信息安全管理体系，但体系在自我运转一年后，尚属于信息安全建设可重复但直观阶段。

（注 ：信息安全建设可重复但直观初级阶段的特征为 ：企业管理层已经认识到信息安全的必要性，并从组织结构中进行了简单资源保证，但信息安全管理制度在企业内部尚未得到有效的裁剪，存在不合理的地方 ；内部信息安全流程已经由承担相同任务的不同人遵守相似的程序阶段 ；内部员工对于信息安全的要求没有系统性正规的培训和标准的交流 ；信息安全责任留给个人，并高度依靠个人个人的知识，容易出现错误的阶段）。

在自我内部差距分析中发现如下几个现状问题存在 ：

- 前一年设置的信息安全制度因属于为通过认证而进行认证的，部分信息安全制度因造成员工长时间反弹力度较大而未继续推行，例如员工电脑终端管理规范，因员工抱怨过大，而停止运行。
- 因企业研发人员流动率高，新入职开发人员对于公司安全代码开发规范不熟悉、不了解，部分代码未遵照安全代码开发规范。

企业按照 PCI 流程进行相关安全运维后，发现运维工作安全比重工作量增加，而企业只有一名信息安全工程师职务，导致大量信息安全工作没有得到及时处理。部分技术检查和审核工作流于形式。

2. 信息安全整改内容

- 对相关信息安全管理制度进行落地裁剪，确保满足合规需要的情况下，确保相关信息安全管理制度和规范的可执行化。
- 根据企业研发现状，调整企业研发流程，将企业代码安全扫描工作由系统运维部进行接管，从而强制在系统上线前研发代码的安全问题在系统上线前及时得到解决。
- 增加了一名信息安全工程师，将部分日常信息安全工作移交运维同事完成，而信息安全管理员重点执行对相关操作和日志行为的安全审计。

3. 信息安全整改结果

经过两个月的信息安全整改后，企业通过了 PCI 安全合规首次现场复审。

29.5.3　第三年信息安全整改实践

1. 企业信息安全状态

历经两年 PCI 信息安全认证后，企业初步建立了信息安全管理体系并进行了落地化的裁剪。信息安全意识在企业文化中已经初步具备。在 2011 年内部风险评估和自我差距分析过程中，初步得出了企业信息安全管理工作已经进入可定义和流程阶段。

（注 ：信息安全管理可定义和流程阶段的主要特征为 ：信息安全相关流程和规章制度已经标准化和记录，并通过培训和沟通。按照要求 , 这些流程在遵循，然而，流程的遵守会出现部分偏差。流程本身并不复杂，但却是现行体系的正规化的重要一步。）

通过核查，信息安全技术的落地工作中还存在不少需要进行改进的工作。下面是 2011 年年度信息安全风险评估中发现的问题的简单描述 ：

- 信息安全运维相关工作因为人工干预过多，存在部分技术落地较差的状态，举例如下 ：信息安全日志的定期审核，在以往的信息安全日志审核，需要信息安全人员人工登录信息系统中执行，随着企业运维机器数量增加，需要查看机器的日志数量增加，以往使用的日志完整性校验软件 Tripwire 在性能方面存在较大的问题而无法承受。导致日志审核无法做到自动化分析和自动化告警。
- 研发安全规范在企业内部推进中，由于安全管理小组隶属于系统运维部，较难推进相关研发部门进行开发软件标准化整改。信息安全部分工作在日常运营中出现因人为错误而停止。
- 随着企业业务的快速发展，特别是非金融支付牌照企业在信息安全方面涉及的部门和人员增加以后，信息安全在各部门的沟通和落地沟通成本增加。

2. 信息安全整改内容

基于内部风险评估分析结果，企业公司系统运维部信息安全经理和公司风险管理部进行了沟通，决定执行下列步骤 ：

- 以信息安全风险为度量单位，设置必要的信息安全风险管理组，由其负责公司层面的信息安全整体策略，系统运维部负责相关政策的落地和执行。在公司组织结构中，单独成立内审部，由内部审计团队单独实施 IT 内部审计，并对公司风险管理委员会汇报相关审计结论。
- 信息安全运维工作实施自动化策略，将日志收集、告警使用 OSSEC 工具进行替代，并将相关告警集成到企业自行开发的安全运维平台，实现生产操作监控、日志安全自动化告警和自动化分析。实施安全运维操作审批工具化，生产运维操作必须进过审批流程进行操作，并在审批流程中设置必要的安全审批环境，所有变更实现自动化记录。
- 将企业安全开发要求落地到研发部门绩效考核中，系统运维部只针对在提交到生产环境中的代码安全进行安全审核，发现的安全隐患数量和各部门绩效进行直接挂钩。

3 . 2011 年信息安全整改结果

企业通过了年度 PCI 符合性检查。这次审查通过比前两次快很多。

29.5.4　第四年信息安全整改实践

1. 企业信息安全状态

在经历了连续三年的 PCI 信息安全合规之后，也就是 2012 年，企业进入了信息安全管理建设第四阶段管理和可衡量阶段（在 2011 年企业经历的非金融支付合规认证过程和结果证明了企业信息安全管理体系的初步建立）。

2. 信息安全整改内容

在企业 2012 年信息安全工作中，信息安全的重点在于以风险评估为指导，基于企业信息安全风险评估结果对企业信息安全工作重点调整，将企业信息安全风险和企业业务风险进行结合。使得企业信息安全重心更加偏向于业务过程安全风险，为企业业务拓展提供保证。同时在信息安全管理体系的落地过程中，更加关注信息安全正常在实现信息安全技术的自动化。

在组织结构中彻底将代码安全的工作职责移交到产品研发部门，实现需求设计中的研发安全考虑、代码开发过程中安全代码功能开发、测试环节中安全隐患的排查。最终在运维环节实现产品完整生命周期的安全。

3. 2012 年信息安全整改结果

企业快速通过了年度 PCI 符合性检查。

29.6　本章总结

1. 在合规中的要点

信息安全合规的主要工作在于依据合规标准的内容进行企业组织结构调整、信息安全管理体系的裁剪和落地、技术差距的整改分析。这三者的关系是并行和缺一不可。

而企业在合规建设中首要工作是取得企业管理层的支持，并具备明确的组织结构进行保证，而负责信息安全合规的相关团队在工作技能方面应该涵盖具备对相关信息安全标准的掌握，如果企业不能理解相关信息安全标准要求，很容易造成对其理解偏离。

在合规项目组组建完毕并获得企业管理层资源的保障后，企业最好进行一次内部自我差距分析，根据分析结果进行相关差距整改，同时在差距整改过程中，若设计到信息安全管理制度的指定，企业不应追求大而全的信息安全管理体系文档，而应注重信息安全管理规范的落地。在进行技术差距的整改中，应尽量通过自动化信息工具和流程来实现技术差距的整改，以方便企业在第二年度审核中保留相关的系统运行证据。

企业进行现场合规检查工作前，应举行召集合规审核机构进行审核前沟通，明确现场审核工作的重点和相关接待要求，并将审核计划详细信息和被审计各部门提前沟通，确保审计的顺利进行。

在企业进行现场合规检查中，企业应尽量安排对企业各部门熟悉且了解信息安全合规标准的

员工进行陪同，并由其作为合规接待对口人进行合规证据递交的接口人。

在合规检查完毕后，企业应和审核机构一起沟通明确合规检查相关结论，并对其中的异议地方当场提出，并根据讨论结果进行调整。

2. 在合规中的难点

（1）合规要求和人员意识的冲突

信息安全合规工作中，做为项目负责人，工作的内容主要是和人打交道 ：和企业内部沟通，协调外部审计机构。在和企业内部沟通中，经常会遇到员工如下的观点 ：合规工作和我的部门职责无关，合规工作就是找麻烦的事情。而在和合规检查机构沟通中，经常会面临 ：对不起，请参考合规标准，具体的检查清单和评判标准不能提前给到你。

这些沟通的困难实质在于，被审核部门以及审核部门的部分人员没有将利益点放在一起。而做为项目负责人需要将双方的利益交际点明确出来，并取得对方的共鸣。

（2）合规整改需求和资源保证的冲突

企业进行任何项目工作，获得足够的资源保障是很困难的，如何在资源保证不足的情况下获得企业管理部门的支持和理解是合规工作成功的关键点，而资源保证的困难实质在于企业管理层是否对合规工作真正足够理解和支持。

（3）合规建设和持续管理的冲突

合规工作，在首次作为项目的方式启动，在项目的初期企业各部门通常都保持了足够的重视，管理层经常进行对进度表示关注，各部门对项目过程中的难点进行积极推进支持，对合规工作而导致的不便利性表示容忍和接受，而合规工作通常都是一个持续进行并逐步优化的过程，而企业各部门往往由于考核指标的变化，容易出现对各项规则制度和要求产生反抗和抵触心理。这使得企业合规工作无法检查和一贯性。而这个困难点的根本原因在于企业是否愿意将合规要求真正的落地到企业日常运营过程中，并对其运营的质量进行监督和优化。

3. 进一步的建议

在信息化建设过程中，企业通常以项目的方式进行“ 运动化 ”建设，例如办公自动化的建设、企业客户关系的管理。企业合规工作也容易陷入运动战中。而合规工作往往需要持久进行优化，所以合规落地化的首要工作是将合规工作的各项要求落实到企业各部门的日常考核中。只有落实到企业日常部门考核中，企业合规的各项要求才会不反弹、不走偏。其次合规工作的重要意义在企业的内部宣传也非常的重要，合规工作难推进的原因很大一部分在于企业员工和管理层不了解、不重视。

在实践中，有过这样一个合规项目，由于合规工作的意义得到企业管理层的足够重视，内部员工通过合规标准宣讲，知悉项目的意义和部门责任，在短短的两周内，就完成了合规的各项准备和现场检查工作。

总结下来，信息安全合规如果要想做好，关键是 ：短期的重要工作是宣传要得力，长期的重

要工作是将合规工作要求落实到企业管理部门的日常考核中。

参考文献：

[1] PCI Security Standard Council, 支付卡行业（PCI）数据安全标准，PCI，2008，http://wenku.baidu.com/view/f1062dd276a20029bd642d82.html

第五部分　组织能力
Organization Capability

第30章　组织能力

30.1　组织能力

30.1.1　企业成功的关键

企业定义了明确的战略方向，只是企业在竞争中致胜的第一步，企业要取得最终的成功不仅要有正确的战略，更重要是要建立能够将确定的战略得以实施的组织能力。这就是在企业管理中经常强调的 ：

企业成功＝正确的战略　X 组织能力

这两个因素之间是相乘关系，而不是相加关系，任何一项不行，企业就无法获得成功。如果企业只有正确的战略，却没有合适的组织能力，即使商机出现，也无法把握。

建立实施战略的组织能力往往比确定企业战略更艰难，这就是为什么有些企业确定了正确的战略方向但依然不能致胜。对一个具有一定规模的企业来讲，战略调整有时要经历几个月，而组织能力的建立和调整往往需要几年。

因此，企业取得成功的保障就是企业能够围绕其确定的战略进行组织能力建设。

30.1.2　组织能力的定义和建设

“ 组织能力（organizational capability）指的不是个人能力，而是一个团队（不管是 10 人、100 人或是 100 万人）所发挥的整体战斗力，是一个团队（或组织）竞争力的 DNA，是一个团队在某些方面能够明显超越竞争对手、为客户创造价值的能力。”[1]

组织能力是企业竞争力的 DNA，它有几个特质 ：

- 它是独特的，每一家企业都有与其战略相适应的特定的组织能力。
- 不同的组织能力，都将局限或强化企业在不同层面的表现。
- 组织能力是源生于企业内部的。
- 虽然战略和组织能力在企业的持续成功中同等重要，但在现实中，组织能力在影响企业成功方面往往起到更为关键的作用。
- 组织能力的建设周期长，难度大，常常成为遏制企业发展的主要瓶颈。
- 战略很容易被模仿，而组织能力难以在短期内模仿。

无论是制定正确的战略，还是打造合适的组织能力，关键在于最高领导人和领导团队的能力、判断和坚持。

组织能力来自三个部分[1] ：

- 员工能力（Employee Competency）
- 员工思维（Employee Mindset）

· 员工治理（Employee Governance）

组织能力的建设 ：

- 战略是通过高层领导团队，花费相对短的时间就可以制定的。
- 组织能力是通过全体员工投入、耗费较长的时间才能打造的。
- 任何变革措施，如果没有公司的最高领导层的支持和推动，仅仅依靠人力资源部门是很难取得实质性成就，因此，组织能力常常成为遏制企业发展的主要瓶颈。
- 组织能力的建设需要各级管理者的积极投入，担负起培养员工的责任。

组织能力建设的第一步是要对企业的组织能力进行分析。按照杨国安教授提出的“ 组织能力三角形 ”模型，组织能力的建设与保证有三大支柱 ：员工能力，员工思维模式，员工治理方式，如图 30 - 1 所示 ：

图30−1　杨国安教授的组织能力模型[1]

根据这个模型，建构或改造企业的组织能力，需要同时在三个方面着手 ：

· 员工能力 ：员工“ 会不会 ”？
  · 员工是否具备应有的知识、技能和特质来实施公司的战略。企业要从人员的招聘、选拔，以及训练、能力提升方面着手，来建立员工能力，以解决员工“ 会不会 ”的问题。
· 员工思维模式 ：员工“ 愿不愿意 ”？
  · 员工思维模式也称员工的心态，指员工是否有意愿、有发自内心的动力去实施企业的战略。企业要从企业的文化、绩效考核体系、态度的引导等方面着手，以解决员工“ 愿不愿意 ”的问题。
· 员工的治理方式 ：员工“ 能不能 ”？
  · 指是否有使员工得以充分发挥能力的管理机制。企业要从组织结构、工作流程、规章制度以及信息沟通方面着手，以解决组织“ 能不能 ”让员工发挥作用的问题。

建构企业的组织能力，要靠在上述三方面的相互配合、缺一不可，少了任何一个方面，都将会使所有的努力功败垂成。

30.1.3　我们的写作

本章的写作，会基于杨国安教授的组织能力三角型理论，围绕云计算企业的服务运营展开。这是以卓越的服务能力为中心，具体讨论员工的能力、思维和治理各个方面的具体要求。后面的几节，包括案例研究，都是围绕这个重心和按照这样的思路进行讨论的。例如，在员工的治理上，会集中讨论服务的组织架构和流程（如图 30-2 所示）。

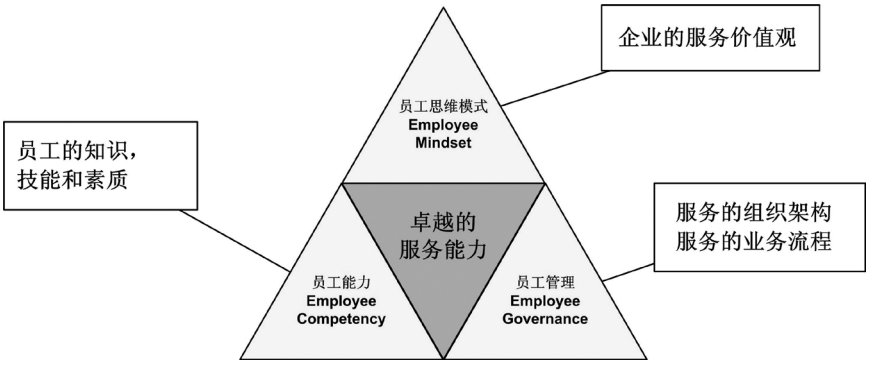


图30-2 服务运营的组织能力要求

我们写作的视角，是讨论一个传统的 IT 企业，如软件公司，向云计算服务提供商转型中所遇到的组织能力上的挑战及其构建的实践探讨。选择这个视角的原因是，随着 IT 行业从硬件、到软件、再到服务的发展，云计算服务提供商大部分脱胎于软件企业，如 Salesforce 和 Siebel 来自 Oracle。同时，云计算服务公司的商务团队和技术团队 也都大部分来自软件企业。

30.2　云计算服务公司面临的挑战

30.2.1　云计算服务对组织能力的挑战

组织能力建设的目的是要使团队能够有效地执行公司的策略。因此，公司策略的确定，就定义了组织能力建设的挑战。

对于一个云计算服务提供商而言，其挑战主要是来自于两个重要的公司策略 ：服务运营和服务创新。虽然这两个策略都非常的重要，但是在某一个阶段，一般只能有一个策略是重点。

- 在公司的服务产品推向市场，向客户提供 7x24 的服务的时候，公司的重点是服务运营，目的是建立卓越的服务能力。
- 当公司成长到一定的阶段，需要不断推出新的服务的时候，公司需要的是服务创新的策略，相应的组织能力也需要有不同的要求。

这两个策略和相应的组织能力的要求，是随着公司的不同阶段而不同的。它们之间的交替，伴随着公司成长，如图 30-3 所示。

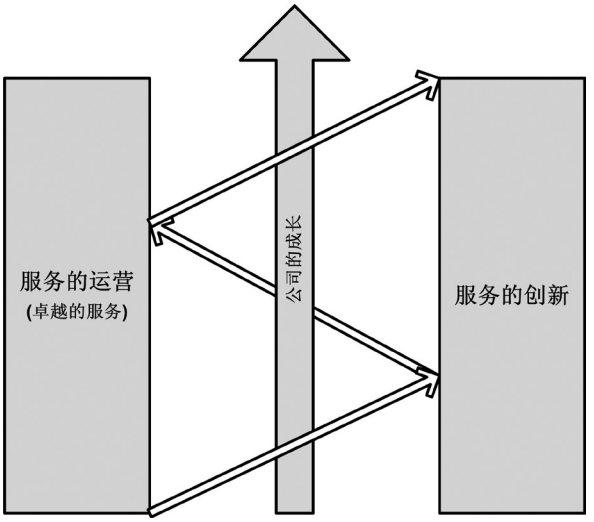


图30-3 云服务公司的不同阶段的策略

不同阶段的公司组织能力建设，将围绕着各个阶段的相应的策略重心而进行。

服务运营与服务创新的组织能力有很大的不同，它们各自的都是非常大的讨论题目。我们在这个章节中会重点围绕服务运营的组织能力建设展开讨论。在本书的后续版本中，我们会加入以服务创新为重点的组织能力建设讨论。

30.2.2　卓越的服务能力

云计算服务企业的运营能力集中体现在它卓越的服务能力上。卓越服务能力在组织能力三角型上主要体现在如下方面 ：

员工能力 ：

· 明确服务运营各岗位对人才知识结构和能力结构的要求，选择合适的人才。
· 持续的培训以培养和提高员工的能力。

员工思维 ：

• 从领导层到全体员工对 “ 客户至上 ” 的价值观的理解和与之相配套的激励机制。

员工治理 ：

• 建立合理的组织结构，调整组织边界，促进服务运营的高效。
· 定义明确的服务与运营管理制度和流程，形成组织自我改进的机制。

卓越的服务能力的要求所带来的挑战对整个云计算服务公司的组织结构和团队都有很大的影响，这样的服务能力的要求落实到各个部门时，其相应的要求是有不同的。各个部门要根据自己的职能做出相应的思考，规划和执行。

图 30-4 是卓越的服务能力落实到客户服务部门和工程技术部门的情境。这两个部门实际上是在云计算服务公司中面临挑战最大的部门。

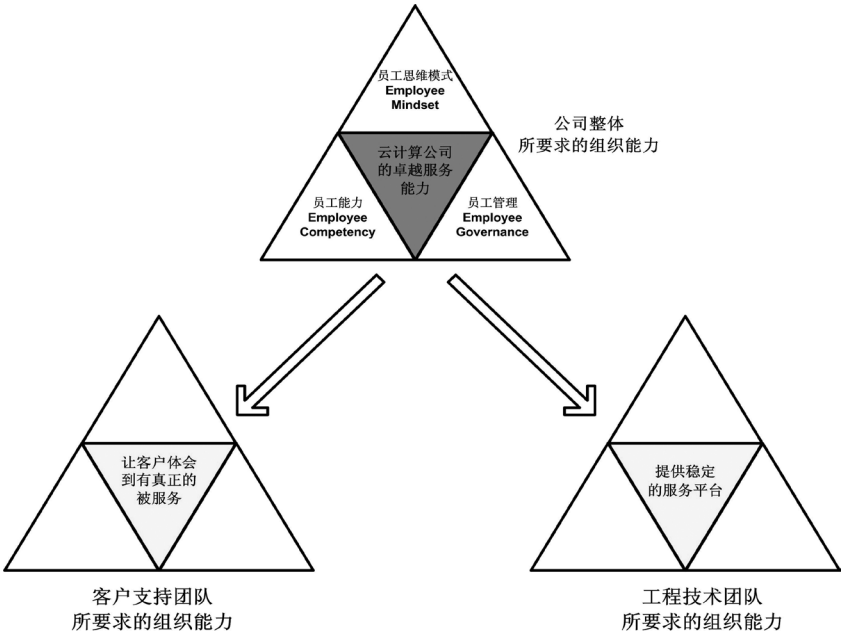


图30-4　卓越的服务能力对客服团队和技术团队的要求

对于工程技术部门而言，服务平台可用度是最重要的服务运营指标，其重要程度远远高于传统的软件产品公司。

对于客户服务部门而言，客户满意度是最关键的服务运营指标，这包括客户问题的响应速度、客户问题的有效解决周期等。

本章后面的章节将做详细的讨论。

30.3　员工的思维方式

员工思维模式（mindset）指的是员工在日常工作中所真正关心的、追求的、重视的事情。企业要打造组织能力，实现战略目标，不仅需要员工具备任职能力，他们还必须有朝公司的方向去努力的意愿，这一点非常重要，因为这决定了员工的思维模式，影响着他们每天大大小小的决策和做事方式。

30.3.1　公司价值观的建立：怎样确定价值观的内容

确定价值观的内容是建立公司价值观的基础。需要有如下几个步骤 [1]

第一步：确定期望的员工思维模式

根据公司的策略，确定需要的思维模式

第二步：分析现有员工的思维模式

找到现有员工的思维模式和期望的思维模式之间的差距以及症结所在。

第三步：制定思维模式变革措施

在找出员工思维模式的差距以及症结后，企业可以根据自身的情况，采取措施重塑思维模式。这个阶段所需时间最长。以下简单介绍员工思维模式转型的三种方式：

- 由上而下 ：这种方式的变革力量来源是依靠高层管理者通过个人言行、决策、制度等多方面的推动。
- 由外而内 ：这种方式的变革力量来源是依靠外部客户和竞争对手。
- 由下而上 ：这种方式的变革力量来源是依靠一线员工的主动参与和推动。

在上述步骤的实践中有几个注意点 ：

- 价值观的定义 ：这是公司最高管理层的责任，管理层一定要想明确了目标以后，才可以做出定义。
- 价值观的内容 ：要有重点，容易理解和记忆。比如，是否可以用一句话或几个词来讲明确公司或部门的价值观吗？
- 价值观的宣导 ：要不断在各种场合、利用各种手段对企业全体员工进行价值观的灌输，价值观要落实到企业的日常工作中，要与具体工作行为相联系，管理层要以身作则。

30.3.2　价值观的落地：团队的接受

改变人的思维方式是很具有挑战性的工作。在实际的操作中，需要拉（PULL）和推（PUSH）的协调效应，如图 30-5 所示。

图30-5　改变人的思维方式在实际的操作中需要拉和推的协调效应

（1）拉（PULL）的效应

拉（PULL）的效应主要来自两个方面 ：

- 一是公司的氛围，也就是“ 耳濡目染 ”；
- 二是高管的以身作则，也就是所说的“ 行胜于言 ”。

（2）推（PUSH）的效应

把体现核心价值观的行为或结果列入绩效考核，并根据考核结果给予员工相应的奖惩，是推动员工改变意愿和行为的有效工具。

由于“ 客户至上 ”是最重要的核心价值观，与之相关的服务理念和衡量要直接反映在服务相关团队的绩效考核之中，如平衡计分卡或 KPI 指标。这样员工才会努力实现公司期望他们达到的

目标。

在云服务公司中，以与工程相关的产品研发部门和技术运营部门为例，在他们的考核中，服务和质量的相关因素将被置于相应团队的平衡计分卡中。以下是相关的内容 ：

- 服务可用性 ：这是直接影响客户的体验。
- 成本效率 ：数据中心、网络、系统、服务平台及日常运营的成本将直接影响公司的盈利能力。
- 团队的技能 ：学习能力与知识水平。
  - 内部学习 ：学习过程的强化及技能改进 ；
  - 外部学习 ：从提供其他服务公司获得的知识，以加快前沿知识和技能的获得 ；
  - 团队成员的交流能力。

附录 30-1 和附录 30-2 是一个云服务公司的产品研发部门平衡计分卡和技术运营部门平衡计分卡。在表中，服务可用度被直接用为衡量的指标之一。

30.4　员工的能力

员工能力是要解决员工“ 会不会 ”的问题。

公司全体员工（包括中高层管理团队）必须具备能够实施企业战略、打造所需组织能力的知识、技能和素质。如何获得员工能力？企业需要回答以下两个具体问题 ：

（1）需要什么样的人才 ：

- 要打造所需的组织能力，公司需要怎样的人才？组织中的各个岗位必须具备什么任职条件、能力和特质？

（2）怎样构建这样的人才团队 ：

- 公司目前各种岗位上的员工是否满足要求？主要差距在哪里？
- 如何弥补这些差距？如何引进、培养、留住合适的人才和淘汰不合适的员工？

对于第一个问题，是各个部门在公司的战略明确后要自己深化的。也就是按照公司的云服务运营的目标，结合自己部门的需求，确定各个岗位的能力要求。比如 ：

- 市场人员 ：需要具有在互联网和移动互联网上进行有效市场活动的能力。
- 销售人员 ：对于 2B 类业务，除了具备大客户销售能力，是否还需要建立有效的中小企业销售能力？对于 2C 类业务，需要什么样的网络推销能力？
- 客户服务人员 ：呼叫中心需要有什么样的一线客户服务技能？客服经理需要什么样的客户问题受理能力和主动服务技能？
- 技术团队 ：需要有能够设计和维护 7x24 的生产型的服务（production service）并保证服务的高可用性（service availability）的能力。

在这些基础之上，各个部门对员工更进一步的能力的要求是不一样的。比如 ：

- 技术运营要求员工同时有技术和管理的综合素质（见本章实践研究（1））。

· 客户服务团队不仅要有客户问题受理能力的需求，还要有主动式客户服务服务的技能，和服务流程建立和改进的能力需求（见本章实践研究（2））。

在后面的案例研究中，我们会针对客户服务团队和技术运营团队的能力要求和构建做进一步的讨论。

在本节中，我们从一个整体云服务公司的角度讨论怎样构建团队。从实践的角度出发，我们选择了两个方面展开；(1）学习型的组织和（2）有效的培训。这是针对云计算服务是一个新兴的行业的特点。

30.4.1　建立学习型的组织

（1）学习型组织（learning organization）的要求

学习型组织是指一个能够帮助其团队成员学习和不断变革和提高的组织。学习型组织的发展是现代组织所面临的压力的结果，其目的使这些组织能够在商业环境中保持竞争力。

学习型组织有五个主要特点，系统思考（systems thinking），自我超越（personal mastery），心智模式（mental model），共同愿景（shared vision）和团队的学习（team learning）。学习型组织的概念是由彼得·圣吉（Peter Senge）和他的同事在 1994 提出的。

学习型组织不存在单一的模型，它是关于组织的概念和团队成员作用的一种态度或理念，是用一种新的思维方式对组织的思考。在学习型组织中，每个人都要参与识别和解决问题，使组织能够进行不断的尝试，改善和提高它的能力。学习型组织的基本价值在于解决问题，与之相对的传统组织设计的着眼点是效率。在学习型组织内，员工参加问题的识别，这意味着要懂得顾客的需要。员工还要解决问题，这意味着要以一种独特的方式将一切综合起来考虑以满足顾客的需要。组织因此通过确定新的需要并满足这些需要来提高其价值。它常常是通过新的观念和信息而不是物质的产品来实现价值的提高。[2]

我们在云服务的组织能力建立的讨论中强调学习型的组织是因为：

· 云服务产业是一个竞争激烈的行业，需要企业的不断的自我提高。
· 云服务产业是一个相对比较新的行业，可以借鉴的外部经验少，需要公司从服务到技术上快速的自我探讨和学习。

比如说：

• 在公司的愿景（shared vision）上，整个公司团队都应该是服务导向（service oriented）。
· 在系统思考（system thinking）上，培养综观全局的思考能力。以技术开发为例，要从开发的理念上，将服务的可管理性、高可用性等从一开始就设计到服务平台中。
· 在团队学习（team learning）上，团队要主动的（proactive）和有效（effective）的从问题中学习和积累经验。

具体到生产运营的技术管理中，组织遇到的挑战是：

• 7x24 生产运营上出现的问题不可能事先都能够罗列出来的，一定会有新的问题不时出现。快速和有效的处理新问题的能力，是要来自于团队很快的学习和提高。
· 在管理流程上，不管流程定义的多么有效，也不可能覆盖所有的问题。这是需要团队的

能力来处理。团队的管理能力提高也是来自学习。

（2）学习型的组织的建立

这里以云服务公司的技术运营团队的实践为例来说明如何建立学习型组织。

- 学习规则（1）：在哪里学习（where to learn）
  - 团队成员中的相互的学习 ：每个人都有自己的长处，经验和教训。
  - 从我们的决策中学习，无论这些决策是对的还是错的 ：要善于反思。
  - 从我们的生产运营中的事故中学习，从现有的事故中，特别是从以往的事故中学习教训。
- 学习规则（2）：怎样学习（ how to learn）
  - 建立 “ 对事不对人 ” 的工作环境（be issues focused, not people focused）
  - 透明的工作环境（transparent working environment）
    - 所有的问题都对整个团队公开，让所有的人都有相同的理解（all the people on the same page），作为有效的讨论的基础。
  - 事故分析会议（postmortems meetings）
    - 事故分析会议是技术运营的一个很重要的会议。对于每个生产线的故障，团队会检讨故障为什么发生，如何发生，如何解决，处理过程，以及相应的技术和流程的改进。这样的会议目的是让相关工程师从故障中总结问题，找到根源和进行改进，从而并防止它在未来发生。
- 改进规则 ：
  - 改进（improvement）是学习活动要达到的目的。改进的目标要具体化，才能具有实施性。” 1/2 时间 ” 规则（The“ 1/2 time ”rule）就是一个简单例子。
  - “ 1/2 时间 ” 规则是在生产线管理中的实践。它的目标规则如下。
    - 对每次的生产线故障时间进行衡量。改进的目标是 ：下一次服务的故障时间应该是目前这次的一半或更少的时间。
    - 这种规则具有一个非常明确的和可衡量的目标。这使这个规则的实际操作性很强。

其他的一些团队能力改进方法包括 ：

- 轮岗制度 ：做技术和管理能力上的不同角色的学习。
- 生产线的值班制度 ：定期的轮班（on - call）来处理生产线问题，以提高 7x24 生产线管理的感知和处理能力。

30.4.2　有效的培训体系

有效的培训体系来自几个基本点 ：

- 了解自己的差距 ；

- 制定针对性的改善措施 ；
- 有力的执行 ；
- 持续的改进。

（1）了解企业自身人才结构的短板

每个企业都有自己的培训体系。要使培训体系能够充分发挥效益，企业首先要了解自身体系目前在培养人才中的短板。表 30-1 是杨国安教授的一个自我评估体系来帮助企业了解自己目前的状况。[1]

表30–1 人才培养体系的自我评估

人才发展的架构模块	当前有效性（1–5分）
• 基础：有明确而透明的人才识别标准和流程 • 培训项目：设计完善的培训项目组合，针对不同的专业职能和不同的级别人才提供培训课程 • 课堂之外的“课堂”：通过工作委派和特殊项目为人才提供锻炼机会，以提高其技能和知识 • 高层领导的承诺和参与：高层领导者通过即时辅导、传授和榜样作用，亲自参与和推动人才培育	

其中自我评估的分级是 ：1 分为最低分，5 分为最高分。

（2）有的放矢：制定改善措施

针对发现的短板，要制定出相应的培训计划。同时要投入充足资源开发多种学习手段进行实施，如在线学习、课堂培训、经验分享、行动学习、校外的专业培训等方式。

这里是一家国内的最大的财务软件公司的培训计划 ：

① 经理人领导力提高
   a. 导师制。
   b. 轮岗。
   c. 商学院深造。
   d. 定期提交工作总结和改进措施。
   e. 每半年的 360 度评估。

② 强化行业高级人才招聘
   a. 行业领军人才，行业技术专家。
   b. 鼓励内部员工成长为行业专家等。

③ 其他（正式建立自己的企业大学）
   a. 全面地、系统地加强员工培训体系。
   b. 系统地优化课程体系。
   c. 系统地优化任职资格体系及员工认证工作。
   d. 强化本年度新入职员工的培训工作。

（3）有力的执行

打造员工能力不是某个人或者某一个部门的职责，是高层主管、中基层主管和人力资源部门的共同使命。人力资源设计架构，各层主管贯彻执行，才能逐步提升员工能力，提升组

织能力。

这里有几个实践中的关键点 ：

- 最有力的执行是来自高层的管理人员。
  - 领导以身作则、教学相长。 好领导应该是好老师。
- 制定和传达培训目的与期望
  - 让团队成员了解目的所在。
- 培训和实践有效结合
- 选择最佳的培训时间
  - 最佳时间是在员工就任新岗位之前。

（4）持之以恒 ：可量化的评估体系

如同服务质量的改进原理一样，人才培养发展体系也需要定期评估及相应的持续改进。

这里是一些评估指标（针对主管 / 经理的衡量指标）为例 ：

- 根据直接部属反馈主管人员的管理能力。
- 根据员工问卷调查，评估主管领导力的有效性。
- 能否培养本地人才接替外派人员。
- 输送人才的数量。

30.5　员工治理

员工治理也非常重要，如果缺乏关键的管理资源和制度支持，员工即使有能力有意愿，也无法充分施展才华，不能为公司做出最大的贡献，公司战略也难以实施。关键的管理资源和制度支持主要包括三个方面 ：

- 权责 ：企业应该授予员工一定的权责。权责多少要视企业的发展阶段和强调的组织能力而定。
- 信息 ：公司要给员工提供及时、有用的信息，让员工能够做出正确的决策、采取正确的行动。
- 流程 ：设定流程体系来支持员工高效率和高质量地完成任务。

在这里介绍云服务公司的整体架构，并对工程技术团队和客户服务团队的架构做详细的讨论。

30.5.1　整体结构设计

组织架构的设计的原则是 ：

- 组织架构的设计很多时候要看企业的发展阶段和经营战略。
- 当组织的规模、管理的复杂度、外在经营环境或战略方向发生改变的时候，组织结构也要随之而变。

· 另外，因为组织如何分工和整合会影响企业内部员工将精力和注意力放在哪里，因此组织设计必须要与企业希望的组织能力和战略重点紧密关联。

以职能分工是一个典型的结构，如图 30-6 所示 ：

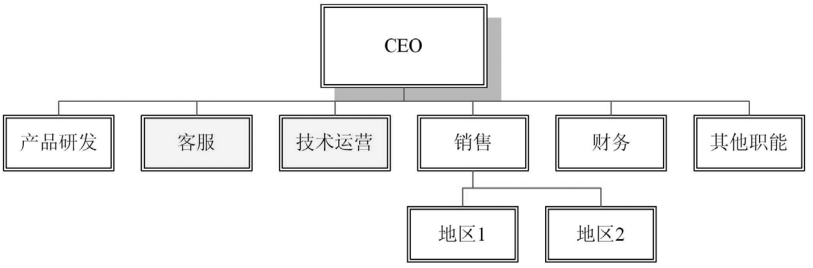


图30-6　以职能分工的组织结构

云服务供应商的职责是要提供 7x24 生产线服务给用户。在此目标下，工程技术部门和客户服务部门的结构将与传统的软件产品公司的不同。在这样的公司战略要求下，各个部门的职能都会有非常大的变化。以工程技术部门为例（如图 30-7 所示）：

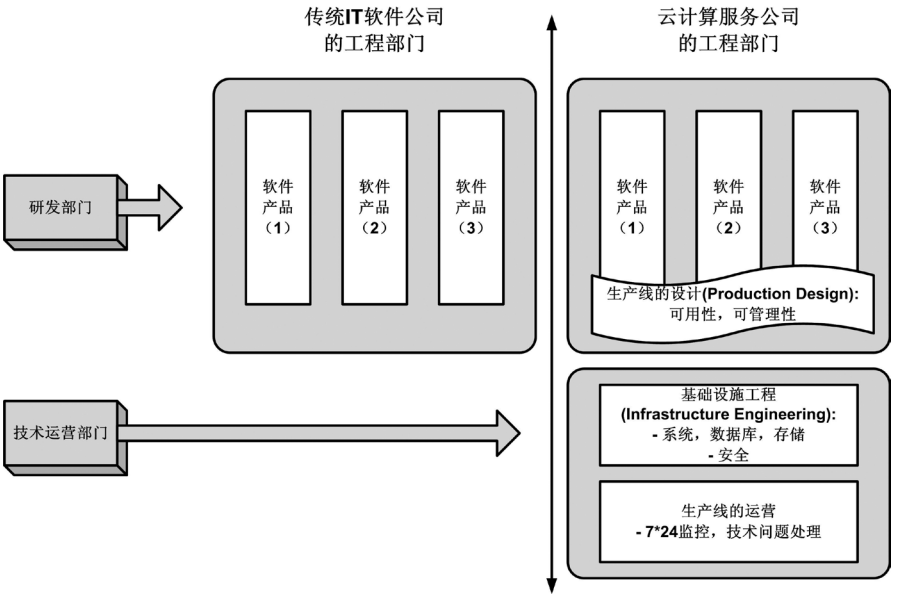


图30-7　云服务与IT公司的工程部门职能的比较

在这个架构下，几个重要部门的职能划分是 ：

（1）产品研发部（Product Engineering）

产品研发部（PE）部负责建设在线服务的应用的开发如 CRM、ERP、会议等，与传统软件公司的产品工程团队类似，产品研发部（PE）组织基本如下 ：

· 技术架构团队
· 开发工程团队 ：服务平台的设计与开发
· QA 团队

其中，开发工程团队可以按照服务平台的层次和技能做进一步的划分。如，

- WEB 开发团队 :负责设计和开发业务逻辑及应用，团队成员以 Java/PHP 等工程师为主。
- 平台开发团队 :负责公用服务的平台开发，团队成员以 C/C++ 等工程师为主。

（2）技术运营工程部（Technical Operation Engineering）

对于一个互联网或云服务提供商，基础设施工程和 7x24 运营管理是必须的。这些功能会在一个新部门来实现 :技术运营工程部（Technical Operation Engineering）。

一般的技术运营工程部的团队结构是这样的 :

- 基础设施工程团队（Infrastructure Engineering）
  设计和实施云服务应用所需要的基础设施，如系统、网络、数据库、电话和服务监控系统的设计，集成，测试和实施。
- 服务质量管理（Service Quality Management）
  负责服务质量管理，包括生产线服务质量和流程质量的分析和改进。
- 安全工程与规范（Security Engineering and Compliance）
  负责服务的技术安全与安全管理。
- 7x24 生产经营和支持（Data Center Operation 或 Network Operations Center）
  负责 7x24 的生产线服务和数据中心监控与支持。

以上这些职能是一般的产品研发团队不覆盖的。详细的请见本章的实践讨论（1）。

（3）客户服务部门（Customers Services）

客户服务部门的目标是为客户提供服务支持，一方面确保客户在遇到问题的情况下能立即通过邮件、电话或者自助提交问题等方式获取相应的服务支持，以保证客户能顺畅的使用服务 ;另一方面进行客户管理，为客户主动提供增值服务，通过云服务为客户创造更高的商业价值。

云服务企业的客户服务要设立专门的机构进行保证，这通常以客户服务部、客户服务中心等形式存在。客户服务部门一般可以按产品类别划分团队或者按照服务职能划分团队。一般客户服务部门可以划分为两大类 :

第一类是面向客户问题受理的客户服务（customers support）部门，如客户服务呼叫中心，由一线服务受理团队和二线支持团队构成，一线受理团队直接受理来自热线电话、邮件、网络等各渠道的应用咨询和问题投诉，能够及时进行日常应用咨询和常规问题处理，在日常应用和常规问题之外的服务需求，需要转到二线支持团队进行进一步的诊断、分析和处理。

第二类是客户管理（customers management）团队，如大客户服务经理团队、中小客户服务经理团队、个人客户服务团队等，主要是对客户的应用状态进行监控，开展提高客户应用效果、为客户创造更多价值的增值服务。

云计算与服务公司最重要的一个特点是业务跨跃时空覆盖广泛的地理范围，客户的应用可能是在全国甚至全球范围内的，因此服务能力也要能够覆盖到所有用户应用区域。按照商业规范的 SLA 服务承诺，企业不仅要提供基于互联网的远程服务，也要对客户的一定级别的问题提供现场服务，这就要求公司的服务具备区域覆盖能力（如表 30-2 所示）。

表30-2 跨区域服务对顾客服务能力的要求分析

	可能存在的主要问题	应采取的管理措施、行动
服务能力的区域覆盖不足对员工治理提出的要求	公司在业务所覆盖的区域没有服务人员，服务差旅成本高。 服务人员不具备多语种的客户沟通能力。 远端服务人员的知识和能力不足	建立地区二线服务支持中心，招聘本地支持人员，服务辐射周边区域客户。 建立客服培训体系，通过服务知识库体系有效支持当地支持人员的服务能力。 与具备区域服务能力的合作伙伴建立服务外包合作

30.5.2　服务体系的架构

在互联网服务供应商的公司，提供服务运营的团队一般是四层结构（如图 30-8 所示）。

（1）第一层

客服支持团队（Call Center），直接面对客户的。

（2）第二层

- 资深客服支持工程师团队。
- 服务开通团队。
- 客户关怀团队及客户经理（Customers Care and Customers Account Management）。
- 培训团队，质量管理团队（包括呼叫中心管理，邮件系统管理，内部系统管理，知识库管理等）。

（3）第三层

技术运营工程团队，包括基础工程与数据中心的监控与管理。

（4）第四层产品研发部门

主要负责软件应用的平台的开发。

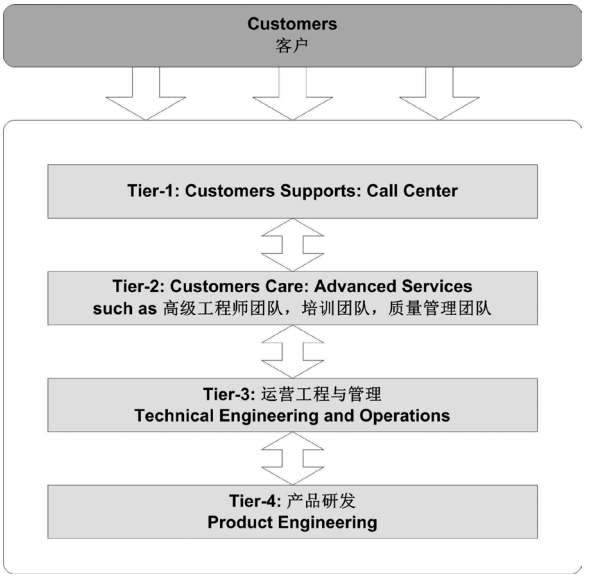


图30-8 服务运营的四层结构

与组织结构相要匹配的事管理流程。客户问题处理流程的设计要满足几个基本因素 ：问题处理的分级和处理问题的升级。具体见附录 30-3 ：问题处理流程的设计。

30.5.3　组织边界的管理：边界的弱化与增强的平衡

在讨论一个公司的结构时，必然会引起的一个管理问题的讨论 ：组织边界的管理。这是因为，公司内部的分工带来的职能（responsibility）与责任（ownership）的不同，必然会带来组织的边界。我们这里简单讨论一下云服务公司内部的组织边界的管理，这直接涉及到云服务的质量管理。

（1）组织边界

组织边界的传统定义是 ：任何使组织内外人员在工作方法、资源、想法和信息上无法顺畅整合的隔阂和障碍就是组织边界。

组织边界的分类为 ：垂直边界、水平边界和外部边界 ：

- 垂直边界 ：不同层级和等级间的边界。
- 水平边界 ：横向部门间的边界。
- 外部边界 ：公司与外部利益相关者的边界。

很多的文章在讨论设立无边界组织（boundaryless organization），其目的是要减少这些边界带来的隔阂和障碍，更有效地整合工作、人员、想法和信息流动。

组织边界常常是被认为是负面的，被与障碍（barrier）和低效（low efficiency）相联系。但在实际上，存在一定的边界是必要的，因为组织内部边界可以确保不同层级和部门的职责明确、专注和专业分工。

而所谓的真正的无边界组织，指的不是要拆除所有边界，而是要从公司打造组织能力，实施战略出发，减少不必要的边界，确保整个组织赢而不是单一某个部门或层级的赢。

（2）组织边界的加强

在云服务行业中，服务质量是关键。这里的增加边界的目的是加强质量管理。所提到的边界是横向部门间的边界，如 ：

- 将 QA 部门与研发部门分开 , 以提高软件发布的质量。
- 将技术运营部门与研发部门分开，以增强生产线的独立管理。

我们这里对技术运营部门与研发部门的关系做些讨论。

云服务行业中，一般是把这两个部门分开。其目的是要在这他们之间建立一种客户关系。这种关系在服务平台的开发流程上就是产品的交付关系（hand-over）。

- 一方面，技术运营部门是产品研发部的客户。可以认为技术运营部门只是一个购买了研发部的软件产品的客户。因此，产品研发部提供的软件产品质量和管理需要达到技术运营部门的所要求 SLA 的水平。
- 另一方面，产品研发部可以认为外包其产品的生产运营管理给技术运营部门。如果技术运营部门无法满足 SLA 要求的水平，产品研发部可以更换到另一个服务提供商。

这两个部门要有十分紧密的合作。技术运营部门需要提早介入软件研发的流程，通常是在软件架构设计的时候就应当参与讨论和审核工作。在设计中，技术运营部门应当把服务平台的可运营性、可靠性、成本、安全等要素带入到设计中。

当软件开发的同时，技术运营部门会同各个专项的研发部门，制定详细的基础架构方案、规划容量、软件上线方案、以及服务支持方案等，以达到最终的产品上线。

技术运营部门与研发部门的分开也是实践中得到的经验。在早期的云服务公司中，技术运营的职能隶属于研发部门，由一位研发背景的高管统一管理研发、QA 和技术运营者三大块功能。常常出现的状况是 ：为了满足商务的要求，产品要求尽快上线，结果就是导致了为了赶时间，QA 和技术运营的要求被降低（be compromised），最后直接导致了产品上线后的服务平台的质量的重大问题。

（3）组织边界的改善

组织边界的改善主要是指减少不必要的垂直边界，或指不同层级和等级间的边界，以提高工作的效率。

这种改善可以通过运用权责、信息、能力和激励这 4 个杠杆，激发企业员工的主人翁精神，让他们的思想和行为都能与公司战略更加协调一致。

这四个杠杆具体而言是 ：[1]

- 公司是否赋予员工足够的权责？
- 有没有提供他们及时有用的信息支持他们工作？
- 有没有投入充分的时间和资源，确保员工具备所需的专业和管理技能？
- 有没有提供适当的物质和精神奖励以激励他们积极参与和投入？

30.6　案例研究（1）：云服务的竞争力 – 高效的技术运营团队

30.6.1　背景介绍

与传统的 IT 公司不一样，云服务公司需要提供的是 7x24 的高可用的服务。相应的，构建一个高效的技术运营团队能力来支持和管理这样生产型的服务，是云服务提供商的关键技术竞争力。

技术运营团队（Technical Operations）和一般的 IT 公司中的研发团队或 IT 团队有着完全不同的目标和定位。

运营工程部主要将研发部门设计的软件平台通过生产线的设计和部署使之变成真正面向互联网用户的服务平台。在服务平台上进入生产运营（production service）阶段后，团队要管理并支持 7x24 的运营，这包括服务的高可用性，安全性，容量管理等。

构建技术运营团队的挑战在于 ：

- 服务的技术运营是一个新的领域，有着技术、管理和经验上的综合要求。

· 技术运营人才的稀缺。

本节从思维方式，团队能力和团队治理的三个方面来探讨如果建立一个高效的技术运营团队。

30.6.2　思维方式：技术运营的管理思想

主要的技术运营的管理理念是几个方面 ：

（1）一半技术，一半管理。

- 在管理上，以 ITIL 管理框架为基础。
- 在技术上，以平台的高可用性和可管理性为基础。

（2）以服务的恢复为第一优先，而不是以找到问题的根源为第一优先。

- 工程技术人员的思维方式是要找到问题的本质。但是，在运营中，客户的服务是第一位的。一旦出现问题，技术运营团队的首要目标是恢复服务，保证 SLA 所要求的服务可用度。

（3）在工程的活动中，如设计和认证过程中，要负向思维，或是说 :“ 默认不正确 ”。这是与研发团队的 “ 默认正确 ” 是相反的。

- 默认正确（Default Positive）
  对于研发团队，开发的目标是使一切功能能够工作（make it work）。他们会考虑他们的成果是正确或是可以工作的，除非证明是有问题。
- 默认不正确（Default Negative）
  对于技术运营团队，一切都需要验证通过才能认为正确。只有这样的思维方式，才会去检验各个关键点，保证服务平台的质量。

在团队的管理思想上，部门管理人员应该了解这两个部门的一些根本性的不同，以便更好地做团队建设。

对于产品研发部门，团队作风是 ：

- 创新导向 ：
  工程师需要是鼓励创造和创新。

对于技术运营部门，团队作风是 ：

- 纪律导向 ：
  在线服务是 7x24 运行的生产环境。管理这样的生产服务，一系列的规章流程必须被严格地遵守，如变更管理和事故管理。 纪律是非常的重要。

30.6.3　团队治理：团队的结构与责任

图 30 - 9 概括介绍了产品工程和运营部工程部的职能区分。

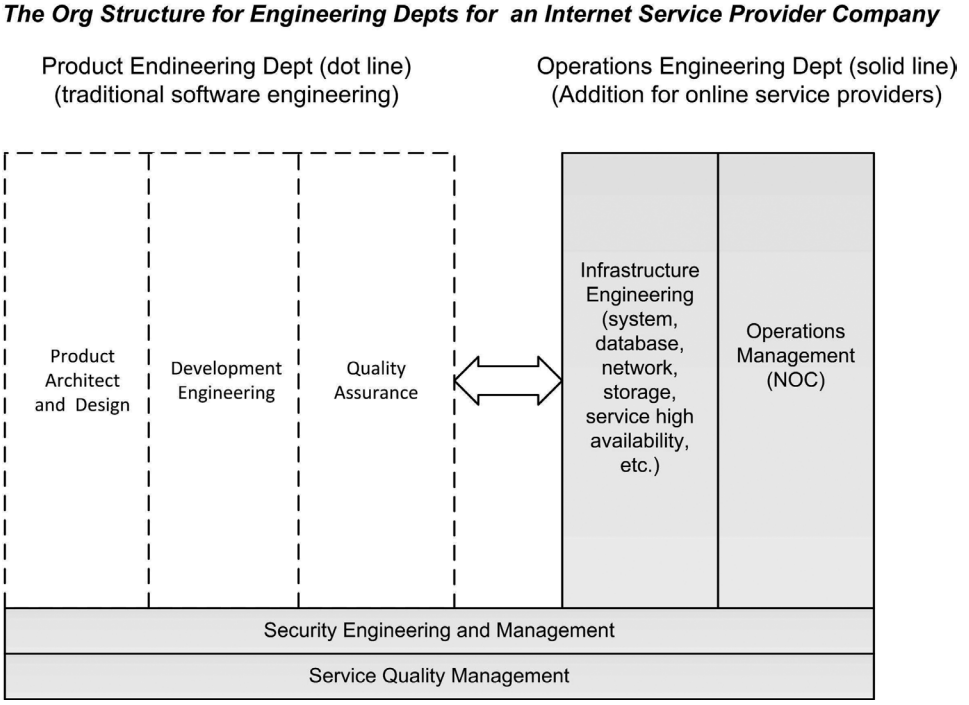


图30-9　云服务公司中产品工程和运营部工程部的职能区分

其中虚线部分是产品研发（Product Development Engineering）团队，实线部分是技术运营团队（Technical Operations）。

在这里，产品研发团队与通常的 IT 软件企业相同：主要有研发、质量保证（QA）、项目管理等功能团队组成。

技术运营部门是通常的 IT 公司所没有的。这里主要讨论技术运营部门的职能。

技术运营部门的职能有生产线的日常运营管理与基础架构工程。具体的职能如下：

（1）生产线的日常运营管理

- 7×24 数据中心管理（IDC）：
  - 负责各个数据中心布置和机房的建设、维护和管理。
- 网络中心监控（Network Operations Center）：
  - 负责各个数据中心从基础设施到应用服务的监控、报警和初步处理，满足 7×24 生产运营需要。
- 日常的运营管理，如生产线管理。
- 管理流程的制定和维护。

（2）基础架构工程（Infrastructure Engineering）

基础架构是指各个软件平台所需要的共用的基础平台，主要包括以下方面：

- 网络工程（Network Engineering）
  - 基础网络的架构、设计、实施和管理。

- 系统工程（System Engineering）
  - 系统设计 ：操作系统选择和规划、硬件平台的选型和规划、系统虚拟化等。
  - 存储 ：各层次的存储的设计，保证客户和系统数据的完整性、安全性和有效性。
- 数据工程（Data Engineering）
  - 数据库管理 :数据库的管理、设计和实施，如高可用数据库（HA database）的构建。
  - 数据复制（data replication）的设计与管理。

（3）应用服务支持

- 生产线上各个产品的日常运营管理和应急响应。
- 产品部署。

（4）技术运营工具开发

- 监控系统开发 ：实现应用层，或是用户体验层的监控。
- 运营管理平台的开发，如变更管理的平台。
- 运营自动化，通过工具实现部署的自动化和批量化。

（5）服务质量管理（Service Quality Management）

- 服务质量的定义、衡量、问题分析、持续改进的执行与跟踪。
- 运营管理流程与规范的定义与监督执行，如事故管理和变更管理。

（6）安全技术与管理

- 安全技术体系的设计、实施与管理。
- 安全策略和合规管理 ：进行管理规范化，以符合政府及行业的安全法规要求。

与这个组织结构相对应的，其他部门的职能也要有相应的调整，这里只是做简单的介绍 ：

- 产品管理团队（Product Management）：要管理来自市场上客户的需求以及来自生产运营的需求。
- 项目管理团队（Project Management）：从产品需求、架构、设计、开发到发布的各个阶段的管理，同时也要延伸到部署规划（deployment），BETA 和正式的生产线上线（production rollout）这些新的阶段的管理。

30.6.4　团队能力: 团队的培养

在我们讨论团队能力的怎样建设之前，我们首先看我们建设的目标，或是说技术运营团队所需要的能力是什么。

（1）技术运营团队所需要的能力

技术运营本身是一半的技术，一半的管理。因此，技术的培养与管理的培养是并重的。同时技术运营是经验积累型，就如医生的职业一样。技术、管理和经验这三个方面，构成了技术运营的知识结构（如图 30-10 所示）。

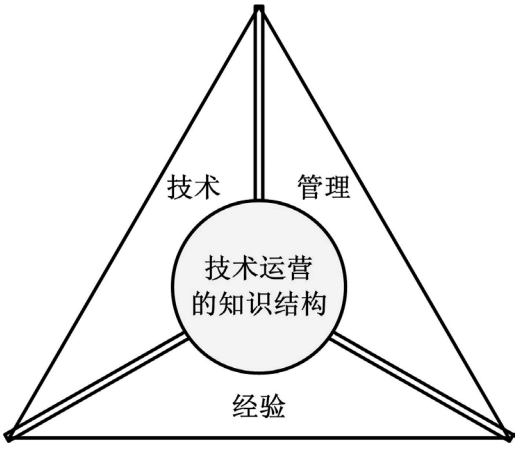


图30-10 技术运营的能力结构

这三个方面的具体要求是 ：

- 技术上的要求 ：技术运营要求的面很广，如网络、系统、数据库、存储等。技术运营工程师对其中的一到两个专项要有精深的了解，同时，对开发也要有一定的技能—这是因为云服务是以应用为中心的（application centric）。
- 管理上的要求 ：要对 ITIL 的管理思想有深刻的了解。在此基础上实施管理流程如变更管理和问题管理等，而不是为了流程而流程。
- 经验的积累 ：这里的经验包括两个方面 ：
  - 7x24 生产线问题的处理经验 ：比如怎样的快速恢复服务。
  - 建立有效管理流程的经验 ：比如怎样建立一个有效的变更管理流程。

（2）能力的建立

技术和管理的能力，可以通过几个通常的渠道进行，这包括 ：

- 公司内部
  - 内部的培训 ：各种技术讲座和讨论。
  - 内部的知识库的建立。
- 公司外部
  - 外部的培训 ：如专业机构的培训、厂商的培训等。
  - 外部咨询师的帮助。

然而，运营的经验积累，是无法完全从上面的通常的渠道完成的。经验的积累是有一个时间的过程。完全靠每个团队成员的逐步积累，从时间上和从生产线的稳定性来看都是无法满足的。因此，对于服务公司及其管理者而言，真正的挑战是如何在最短的时间里，让团队成员能够具有足够的生产线运营的经验。

这里是几个方式 ：

（1）从每个事故中学习教训

① 每次生产线的事故都是深刻的教训。每次的事故一定是有着技术上和管理上原因。因此，事故的分析要非常的细节和深入，要做到真正的反思和学习。

② 这里要特别强调的是，要从每个以前发生过的事故中学习，而不是要把所有的事故都经历一遍才能获得经验。中国的古话：“以史为鉴”（Learn from history），就是这个道理。在7x24 的不间断的服务运营中，这点至关重要。

（2）事故中的决策

① 快速的决策是快速的服务恢复的基础，尤其是在大的事故中。

② 比如，如果一个在线会议系统（online meeting）上有 1,000 人在线，但是服务平台有故障，另外的 1,000 人无法入会。如果现在重新启动（reboot）平台，需要 15 分钟。重新启动平台之后，无法入会的问题可以解决，但是已有的 1,000 人在线会被中断会议。
这样的决策应该怎么做？实际上，这种决策应该是在事前就有备案或考虑，而不是等到有故障发生时再思考和决策，否则，即使是很有经验的技术运营管理者，也会难以平衡事故的影响和决断，从而耽误时间。

（3）有效的运维文档

① 文档可以很多，但是有效（efficient）的不多。有效的文档要讲清楚 3 点：文档的目的（what to do），解决方案的要点（key points to do）和方案的难点（difficult points to do）。

② SOP（Standard Operations Procedure）文档：快速服务维护，是要简洁有效的 SOP 文档。在 SOP 中，要明确什么问题要解决，有什么样的方案。这样，在出现事故时，可以做快速的参考和执行。

这些有效的文档，就是有效的经验的总结和积累。

30.7　案例研究（2）：构建服务导向的客户服务部门

30.7.1　服务组织的团队能力要求

提供服务的行业有很多。服务组织的能力要求可以分为基本要求和各自行业的要求。

（1）基本要求

基本要求是指整个服务行业的共同要求，如服务导向型的人才的要求，包括能够帮助企业为客户提供热情和真诚的关怀来及时的解决纠纷，提供服务，回答问题，并确保客户的满意度。下面列举了一些这样人员的特征，可以用来为团队招聘和培训时做参考。[3]

① 乐意助人的（helpful）：愿意主动的帮助别人。

② 为他人着想的（considerate）：如换位思考。

③ 合作性的（cooperative）：与团队成员协助和与客户互动。

④ 沟通（communicative）：如善于倾听客户的意见，有效的观察和表达能力等。

⑤ 解决问题（problem solver）：快速的定位问题，提出可行的方案。

⑥ 条理性（organized）：有能力判断和掌握任务的优先级，同时处理多个问题。

（2）行业要求

云服务企业是通过为客户提供云计算和应用服务来为客户创造价值的，因此赢得客户对服务的满意是企业服务运营的根本目标。为客户提供卓越的服务可以带来客户忠诚度，从而保证基于

服务的商业模式的稳定收入，和由于客户增加对服务的使用而产生的收入增长。

公司要为客户提供卓越的服务，要具有如下的服务能力 ：

- 被动式服务，即及时响应客户在应用中出现的投诉或问题，并有效地解决。
- 主动式服务，即采取主动的服务行动，避免或减少客户在服务中遇到问题，并发掘客户对服务的进一步需求，主动改进服务以满足客户的深层次需求，为客户创造更多的价值。被动式服务相当于是“ 救火 ”，而主动式服务相当于是“ 防火 ”。

云服务公司在初建时，由于要不断地处理来自客户各种各样的投诉，往往将服务的组织能力重点放在被动式服务上，整个服务体系以呼叫中心和技术支持为中心，而在主动式服务方面往往重视不够，从而出现疏于“ 防火 ”、疲于“ 救火 ”的服务运营状态。作为一个专业化的云服务企业，面对被动式服务和主动式服务，要做到双管齐下，齐头并进。

云服务企业的服务团队的核心思想是让客户有真正的被服务的感觉。这可以在三个服务方式上体现 ：被动式服务 / 主动式服务和服务的持续改进。每个服务方式都可以从员工思维 / 员工能力和员工治理三个维度来发展其相应的组织能力，如图 30 - 11 所示。

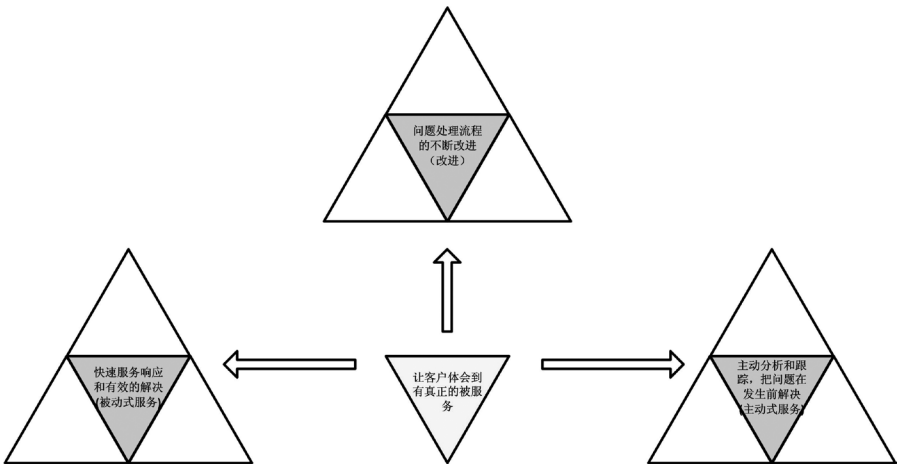


图30-11　服务团队的三种服务方式对组织能力的要求

下面将针对云服务行业对这列出的三个方面做具体的服务组织能力的构建分析。

30.7.2　被动式服务 (1) : 快速服务响应

快速服务响应是被动式服务的一个方面。它的实现来自两个方面 ：

- 客户问题的有效反馈 ；
- 问题处理的有效跟踪。

（1）客户问题的有效反馈

这是云服务企业获得客户对产品和服务反馈的主要途径，也是所有被动服务行为的起点，企业要建立规范化的、系统化的客户问题反馈途径，保证所有客户的问题都能够被倾听、记录。对客户的问题置之不理是对客户对大的伤害，甚至于超过产品缺陷对客户所造成的伤害，我们要给

予高度的重视。对组织能力的要求如 30-3 所示。

表30-3 客户问题有效反馈对组织能力的要求分析

客户问题有效反馈	可能存在的主要问题	应采取的管理措施、行动
员工能力	1．业务受理人员业务知识不足，不能准确理解和描述问题 2．客户问题被遗忘	1．受理人员业务培训与考核 2．问题受理话术设计 3．问题受理过程辅助信息系统
员工思维模式	1．业务受理人员服务态度不好 2．受理人员流动性大	1．客户第一的价值观宣导 2．明确的业务受理人员岗位职责和要求 3．KPI的衡量指标：客户问题响应时间 4．明确的业务受理人员职业生涯规划
员工治理方式	1．客户问题投诉渠道不通畅 2．客户问题责任人不明确 3．客户问题被遗漏	1．建立客户服务呼叫中心，提供服务热线、网站和邮箱。必要的话建立7X24小时受理能力 2．明确的客户服务经理职责 3．明确的客户问题责任人，第一受理人或客户经理 4．基于CRM的客户问题处理流程

（2）问题处理的有效跟踪

我们有时听到客户投诉“我们反映的问题怎么就不了了之了呢？”、“跟你们反馈很多次了，你们到底有没有人管”等等，说的就是这种问题跟踪能力欠缺的典型表现。问题有效处理对组织能力的要求如表 30-4 所示。

表30-4 问题处理的有效跟踪对组织能力的要求分析

问题处理的有效跟踪	可能存在的主要问题	应采取的管理措施、行动
员工能力	1．问题多，受理人员处理不过来 2．员工倒班、休假或变更，问题没有被及时跟踪 3．问题超出受理员工的能力	1．建立基于CRM的客户问题跟踪机制，保证客户问题得到及时有效跟踪 2．建立明确的问题升级处理机制，使不能在一线解决的问题及时传递到二线、三线支持人员
员工思维模式	1．客户问题不了了之 2．客户问题处理周期长	1．客户第一的价值观宣导 2．明确各服务范围的客户问题负责人 3．KPI的衡量指标： - 客户问题受理周期 - 客户问题解决满意度
员工治理方式	1．客户问题长期得不到解决 2．不能协调服务部门之外的资源解决问题	1．不同层次的周期性遗留客户问题评审 2．基于CRM的客户问题传递与跟踪机制 3．建立明确的问题升级处理机制，在相关部门设立用户问题协调人员，专门跟踪传递到本部门的问题处理进展

30.7.3　被动式服务（2）：客户问题的有效解决

能够获得用户问题的反馈只是被动服务的第一步，而问题能够得到及时有效的解决才是对客户的真正价值。

我们有时会听到客户讲，某家公司的服务人员态度不错，也挺主动热情，但就是解决不了问题。这些客户是有很大潜在流失风险的，他们流失的重要原因是“不解决问题”或“问题得不到

解决 ”。因此，被动式服务的根本目标是要及时有效地解决用户的问题。这来自三个方面 ：

- 团队问题处理能力
- 问题处理升级机制
- 问题处理中的沟通

下面对这些方面做详细的讨论。

（1）团队问题处理能力的建立和提高

问题处理能力的建立不是一次性的工作，因为随着客户对云服务的深入应用，会不断产生出各种各样的新问题，这就需要企业有一个问题处理能力的培养、积累和持续提高的组织能力。它对组织能力的要求如表 30-5 所示。

表30-5　团队问题处理能力的建立和提高对组织能力的要求分析

团队问题处理能力的建立和提高	可能存在的主要问题	应采取的管理措施、行动
员工能力	1．员工问题处理能力弱 2．知识太多，记不住，不好查	1．明确岗位技能要求，培训考核上岗 2．建立问题处理知识库，并提供问题解决过程辅助工具
员工思维模式	1．工作忙没有时间学 2．复杂问题还是留给高手解决吧	1．建立服务支持人员技术等级制，等级与岗位工资挂钩 2．知识库知识贡献奖励机制
员工治理方式	1．问题处理能力过于依赖于个人 2．这个问题他会，我不会 3．同一个问题，上次解决了，这次还处理不了	1．建立知识库系统 2．周期性知识共享与考核

（2）问题处理升级机制

对于按正常的处理流程不能不有效解决的客户问题，需要协调公司更多的资源研究解决，而要协调服务系统之外的资源，则需要按照一定的流程规范将问题升级到更高的管理层次，以便于引起重视，安排资源通过合作解决问题。这方面的能力的要求如表 30-6 所示。

表30-6　问题处理升级机制对组织能力的要求分析

问题处理升级机制	可能存在的主要问题	应采取的管理措施、行动
员工能力	无法判断问题是否要升级	明确问题升级机制，定义清晰的升级条件，如问题对客户应用的影响程度、客户问题关注层次、问题处理周期、问题无进展时间等
员工思维模式	1．问题解决不了怕暴露个人能力不足，不升级 2．还想再尝试努力寻找解决办法，造成问题拖延时间长	1．KPI：问题处理周期 2．不能及时升级的惩罚措 3．基于CRM的自动升级策略施
员工治理方式	1．不知道升级到什么部门、什么层次 2．不知道升级后的如何处理，	明确可操作的的升级规范，包括问题状态和需要达到的部门和层次，跟踪问题的处理过程等

（3）问题解决过程和结果沟通

客户有权力对于所投诉的问题的处理过程和结果希望有所了解，我们有义务给客户提供必要的信息，并且便于客户去获得相关的信息。相关的能力分析如表 30-7 所示。

表30-7　问题处理过程和结果沟通对组织能力的要求分析

问题解决过程和结果沟通	可能存在的主要问题	应采取的管理措施、行动
员工能力	服务人员与有投诉、不满意的客户进行沟通的能力弱	1．与客户，特别是不满意的客户，进行沟通的话术设计和技能培训 2．安抚客户的技能培训
员工思维模式	1．不能正确对待客户对自己团队甚至自己的投诉 2．对与不满意的用户进行沟通有畏难情绪	1．树立感谢客户投诉的价值观——客户的投诉是为企业提供改进的意见和建议，是在帮助企业 2．KPI：及时沟通完成率及客户满意度
员工治理方式	1．对问题处理过程不了解 2．沟通方式单一，且对个人能力的依赖性很大 3．客户对处理过程或结果依然不满意	1．基于CRM的问题处理与跟踪机制，并对需要沟通的环节进行提醒 2．建立用户问题处理状态的多种反馈机制，如网页查询，问题处理过程及结果报告模板 3．针对问题类型和对客户的影响程度，协调销售和客户经理参与反馈

30.7.4　主动式服务：有效的客户管理

主动式服务是要对公司的客户在其没有发生投诉或问题报告时主动开展的服务，它是以有效的客户管理为基础的。因此有效的客户管理是云服务企业的一个重要的服务组织能力。

（1）客户管理与服务团队建设

建立主动式服务客户的服务团队的要求如表 30-8 所示。

表30-8　建立主动式服务客户服务团队对组织能力的要求分析

客户管理与服务团队建设	可能存在的主要问题	应采取的管理措施、行动
员工能力	1．呼叫中心员工的知识结构和能力不适应给客户提供主动服务 2．客户经理不知道主动式服务如何开展	1．设立客户经理专门的岗位或团队，以明确主动服务的职责 2．客户经理的层次和能力要高于普通客服人员，最好从优秀的客服人员中选拔 3．建立客户服务工作规范，客户经理不仅靠主动积极的服务态度，更多的是按服务规范执行
员工思维模式	主动式服务的服务行为和效果不好评估	建立客户满意度调查机制，要确保能够听到客户客观而真实的声音
员工治理方式	1．员工忙于解决客户投诉，无暇顾及为主动服务 2．服务内容超出客户经理能力和资源范围	1．设立客户经理专门的岗位或团队，以明确主动服务的职责 2．建立客户服务工作规范和流程

（2）客户数据库管理

客户的状态信息来自多个渠道，被多个职能使用，要通过建立公司统一的客户数据库以保证客户信息的及时、准确和同步（如表 30-9 所示）。

表30-9　建立客户数据库的要求分析

客户数据库管理	可能存在的主要问题	应采取的管理措施、行动
员工能力	客户数据量太大，特别是动态信息，员工是不可能记忆如此大量的信息的	1．建立公司统一的客户数据库系统，并与CRM系统进行集成 2．配备数据库管理员对数据进行管理和优化 3．面向不同的业务职能设计差异化的用户界面和视图，并提供简单易用的用户操作界面
员工思维模式	客户数据管理枯燥，容易出错	
员工治理方式	客户数据逻辑关系复杂，面向不用的职能有许多不同的视图，很少有人能够全面掌握	

（3）客户满意度调查机制

对提供云计算和服务型的企业来说，客户满意是维持企业收入的关键。由于企业并没有直接自己投资建设基础设施和服务能力，仅仅是为所租用的服务付费，因此他们满意就会继续使用服务并付费，他们不满意就可以随时中止服务的使用并转向更好的服务商。企业要经常扪心自问，“客户对我们的服务满意吗？”。因此，企业一定要客观真实地了解客户对服务的满意度，并针对客户的反馈不断改进服务，提高客户的满意度，留住客户。相应的组织能力要求如表30-10所示。

表30-10　建立客户满意度调查机制的要求分析

客户满意度调查机制	可能存在的主要问题	应采取的管理措施、行动
员工能力	1．客服员工仅能通过客户的态度察觉表象层面的客户满意度，并不能真正了解客户的满意程度 2．客服人员缺乏能力从不满意的客户处系统了解满意度和改进点的能力	设计客户满意度调查问卷
员工思维模式	1．客服员工有自我肯定或拒绝他人否定的倾向 2．客户经理有时通过与客户接口人建立良好的个人关系，以期获得好的评价 3．客观反映客户满意度可能会得罪内部员工	1．客户满意度调查要坚持第三方的原则。基于用户问题解决的case级别的满意度调查要由非当事人做，最好由独立的部门或人员做；客户综合满意度调查要请第三方机构做 2．利用网络调查问卷
员工治理方式	1．客户的满意度往往不仅仅是对服务本身的，可能是对企业的综合反馈，如企业文化、产品、销售、技术等多方面 2．客户对满意度反馈率不高，嫌麻烦 3．满意度调查结果不了了之	1．第三方机构做用户满意度调查 2．利用网络调查问卷，问题要少，操作简便，几分钟完成 3．鼓励客户反馈的激励机制，如积分、返点、回扣等 4．客户满意度调查后续处理流程，包括统计分析，识别改进点，试点，服务规范更新等

（4）客户应用问题的主动发现

如果客户没有投诉，也不与我们联系，客户对我们的服务就没有问题吗？

我们经常听到的客户抱怨是，“我们不找你们投诉，你们从来也不与我们联系”，“希望你们帮助我们用好你们的产品和服务”，这些都是对我们主动服务的不满意。云服务企业要善于发现客户在应用中的问题，即使他们没有意识到问题或没有投诉，并通过改进服务为客户创造价值。相应的组织能力的要求分析如表30-11所示。

表30-11 主动发现客户问题的要求分析

客户应用问题的发现	可能存在的主要问题	应采取的管理措施、行动
员工能力	1．客户经理不懂客户的业务，不易发现客户在使用我们的产品或服务中存在什么问题 2．客户经理不知道何时客户有问题	1．客户经理按行业进行分工，客户经理要对所负责人行业有所了解和研究 2．建立客户周期回访机制，定期主动联系客户，沟通使用情况 3．建立各行业应用示范案例库，促进相近行业企业之间的分享和交流
员工思维模式	员工的惯性思维如下： 1．客户没有联系我们或投诉，说明客户没有问题，客户不喜欢经常被打扰 2．客户的问题可能是复杂的，我们外人怎么可能解决人家内部的问题 3．客户自己内部的问题不一定愿意与我们分享 4．解决用户的应用问题绩效无从衡量	1．建立为“客户创造价值”、“与客户共赢”价值观 2．制订KPI考核客户经理的周期回访率 3．对于客户经理主动发现客户问题并促进解决而产生的增值服务收入，可纳入绩效或提成
员工治理方式	客户问题可能涉及产品、研发、营销、服务、运营等多个环节，不是我们客户经理能够协调的	协调公司资源对重点客户或区域建立“应用推广日”，由各职能协同识别和解决用户问题

30.7.5 服务体系改进

（1）基于问题的服务体系改进

客户投诉问题的发生，有其偶然性，也有其必然性。偶然性体现在这个客户在这种情况下发生了这个问题给他带来了不便、困扰甚至事故，必然性体现在我们的服务体系可能存在有需要改进的地方，如果我们发现并采取了措施，可以避免或减少这个问题的发生。

我们有些企业经常会花重金请一些外部咨询机构来帮助诊断我们服务体系，给我们提出全套的改进方案，而我们的客户是给我们付服务费的同时还在通过投诉，免费帮助我们找到我们服务体系的问题，我们完全没有理由采取就事论事的态度将这个投诉平息的道理，相反，我们应该在帮助客户“灭火”后，要认真在我们的系统或服务系统中分析“起火”的原因，找出“防火”的方法。

企业就是要通过建立“基于问题的服务体系改进”能力，从用户的每个问题投诉中寻找改进方案，从而实现企业服务体系的自我改进和完善（如表 30 - 12 所示）。

表30-12 基于问题的服务体系改进的组织能力分析

基于问题的服务体系改进	可能存在的主要问题	应采取的管理措施、行动
员工能力	员工不具备从客户具体问题中发现体系问题的能力	1．在基于CRM的问题处理与跟踪机制中，在每个问题处理后加上一个“产品及服务改进措施”环节，要求客服员工及主管提出建议，并启动改进措施评估和实施流程 2．对客服人员进行培训，使他们建立对产品和服务体系的整体认识和理解
员工思维模式	客服人员对客户问题投诉的消极心态，希望尽快就事论事解决问题，不想将问题“扩大化”	1．进行“积极心态对待客户投诉”的价值观灌输 2．建立KPI激励客服人员从客户问题处理过程中提出服务改进建议

基于问题的服务体系改进	可能存在的主要问题	应采取的管理措施、行动
员工治理方式	1．问题涉及的面可能是多方面的，超出客服人员甚至主管的工作范围 2．客服人员无法及时了解到根据问题找出的服务改进措施，导致改进措施没有得到落实	1．将问题处理与跟踪机制中的“产品及服务改进措施”环节启动一个公司级的业务改进流程，按问题的类别或性质进行升级 2．改进措施的具体内容纳入客服日常的培训和共享机制

（2）服务项目设计与改进

通过主动与客户沟通发现的应用问题，有许多是其他企业也存在的问题，或可以提炼出共性的东西，要避免简单的就事论事，要善于系统化地改进产品、服务，从而提高企业的整体服务能力（如表 30-13 所示）。

表30-13 服务项目设计与改进的能力要求分析

服务项目设计与改进	可能存在的主要问题	应采取的管理措施、行动
员工能力	客户经理不知道哪些问题是共性的，只好就事论事处理了事	1．建立基于CRM的客户应用问题管理机制，在流程中，在特定的客户的问题解决后，要启动一个“服务改进措施”的任务 2．由公司的产品、运营、服务等部门成立联合小组定期对“服务改进措施”任务进行评估，制订工作计划
员工思维模式	客户经理是服务产品和规范的执行者，设计和改进服务产品或规范不是客户经理的职责	
员工治理方式	谁有能力进行服务问题的判断、设计和改进	

30.7.6　实践讨论小结

上面对客户服务团队建立以客户为中心，让客户能够体会到有真正的被服务的组织的能力构建做了详细分项。根据这些的分析，云服务企业在服务团队的组织能力建设方面要从“应采取的管理措施和行动”中整理出所有的项目的清单，针对清单中的内容，按重要性、工作量、难易程度等几个方面综合评估，排出优先级。针对每个项目，要指定负责人，制订工作计划，使这些服务组织能力的构建得以实施。

附录 30–1 ： 产品研发部门平衡计分卡

产品研发部门（Product Engineering Dept）平衡计分卡（季度）			
维度	目标与主题	核心衡量指标	目标值 季度
1. 财务层面	公司季度收入目标及支持	公司季度达成收入	
	公司季度毛利目标及支持	研发成本（如部门人员成本）与公司收入的比率	
2. 客户层面	生产系统稳定性	服务可用度（无故障运行时长/总时长）在SLA 要求的水平	
		重大事故发生次数	
		重大事故修复平均时长	
		产品上线成功率 （成功上线次数/总上线次数）	
	生产系统安全性	防钓鱼投诉及其他应用边界安全性	
		重大数据安全性事故，如重要数据丢失或泄密	
		安全认证	
	数据完整性与合规性	产品缺陷或事故引发数据差错导致报损次数	
	产品研发	各产品线的非标项目按时保质完成率	
	产品与业务容量规划	各产品线的业务容量分析与性能提升	
3. 内部运营	内部流程持续优化	研发项目工程化管理标准（Engineering Lifecyle Management）覆盖范围	
		内部流程有效性评审与改进次数	
		内部流程的文档化比例	
		内部网站对流程的覆盖比例	
	内部控制规范化	项目管理能力指标	
		环境一致化：生产系统、BETA、集成测试环境的高度统一	
4. 组织与人力	有效的员工绩效与激励机制	按要求及时完成员工绩效相关工作	
		有效控制关键员工离职率	
	借鉴外部经验，提升部门研发能力	与世界500强公司进行战略、技术、运营和管理等方面的交流	
	持续提升员工的专业知识与各项技能	专业技能培训次数	
		员工培训参与率	

附录 30–2 ： 技术运营部门平衡计分卡

技术运营部门（Technical Operation Engineering Dept）平衡计分卡（季度）			
维度	目标与主题	核心衡量指标	目标值 季度
1. 财务层面	公司季度收入目标及支持	公司季度达成收入	
	公司季度毛利目标及支持	技术运营成本（如部门人员成本，数据中心成本，生产线软件和硬件成本等）与公司收入的比率	
2. 客户层面	生产系统稳定性	服务可用度（无故障运行时长/总时长）在SLA 要求的水平	
		产品导致的重大事故发生次数	
		产品导致的重大事故修复平均时长	
	生产系统安全性	重大数据安全性事故，如重要数据丢失或泄密	
		安全认证	
	数据完整性与合规性	央行合规要求	
	系统与业务容量规划	生产系统容量分析与性能提升	
3. 内部运营	内部管理流程持续优化	研发项目工程化管理标准（Engineering Lifecyle Management）覆盖范围	
		内部流程有效性评审与改进次数	
		内部流程的文档化比例	
		部署自动化比例：代码管理、出包、部署、上线的脚本化覆盖范围	
	内部控制规范化	所有项目由技术运营部门整体管理，梳理各项目选型、评测、上线等流程	
	资产管理与使用效率提升	硬件设备的使用效率（如平均CPU占用率）	
4. 组织与人力	有效的员工绩效与激励机制	所有员工配备指导人，定时进行正式沟通	
		有效控制关键员工离职率	
	借鉴外部经验，提升部门技术能力	与世界500强公司进行战略、技术、运营和管理等方面的交流	
	持续提升员工的专业知识与各项技能	专业技能培训次数	
		员工培训参与率	

附录 30–3 ： 客户服务的保证——问题处理流程的设计

1. 问题提交流程

对客户的技术支持一般分为四层。其中的第一层和第二层一般属于客服支持部门。

第一层的技术支持工程师需要直接面对客户。这一层的技术支持人员通常需要有很强的沟通交流能力。并且他们有多种语言能力。他们需要解决绝大多数的客户问题（通常这一比例为 90%）。

第二层的是高级工程师，他们的职责是处理第一层技术支持工程师无法支持的问题或者通过紧急技术支持流程提交的问题。

第三层和第四层技术支持的职责属于技术运营和研发部门。在经过高级工程师的测试确认之后，无法解决的问题将会提交到运营或者开发的相应部门处理。

2. 技术支持的方式

技术支持一般有实时和非实时的支持方式，实时支持方式有电话和即时聊天支持，非实时的支持方式则有电子邮件和客户自助提交问题获得支持的方式。值得一提的是，大多数客户获得技术支持的渠道是通过网络。一般可以向客户提供一个免费的自助技术支持文档库，让客户通过自助查找的方式解决自己遇到的问题以解决技术支持部门的压力。

（1）电话支持

电话支持都是实时的，一般的 SLA 都是定义为客户的电话在某个段等待时间内能被接听。比如超过 85% 的电话能在 2 分钟等待之内被接听。电话支持由于其实时性，所以对支持工程师的要求是最严格的。

（2）电子邮件支持

由于工作时间排班的原因，为了保证支持的连续性，SLA 要求电子邮件支持一般在 24 小时时间内响应。除了一些特别紧急的问题或者需要与客户通话测试解决的问题，邮件支持不会转化为电话支持。

（3）客户自助提交问题获得支持

同电子邮件支持一样，这是一种非实时的支持方式。一般 SLA 也是 24 小时内响应。这种支持方式一般与内部管理系统有整合，属于代价最小的支持方式，也是技术支持部门需要尽力引导客户使用的方式。

（4）紧急问题

除了一般的技术支持渠道之外，技术支持部门也应有紧急问题处理的渠道。一般为客户经理或高级管理人员。在大客户或者其他客户的紧急技术问题而影响到合同继续执行时，向技术支持部门提交，此时会有资深的技术支持人员主动与客户联系处理问题。

以上梯队划分和支持方式并非绝对，在针对不同类型产品可能有所不同。比如针对用户群相对比较小、而技术要求很高的产品时，就有可能只有某些高级工程师提供电子邮件支持。

3. 客户问题的跟踪记录和管理

技术支持部门使用一个内部系统跟踪所有的客户问题。每个客户的问题，除了记录客户姓名、电话、电子邮件等联系方式之外，所有问题的描述，每个处理人员的处理步骤都会被记录。所有问题会以客户公司、问题所属产品类别被分类。系统也会提供报表功能以便各类管理人员查看相应报告。

4. 技术支持质量管理

对于技术支持团队的衡量主要分为两个方面。客户满意度调查和 SLA 的完成情况。

（1）客户满意度调查一般由第三方公司进行或者由技术支持部门中的独立职位人员进行。每个客户在提交问题的同时都会被要求提供有效的联系方式，在问题处理完之后，系统会自动（随机）发送客户满意度调查邮件，以收集客户的反馈。

（2）SLA 的完成情况可以分别由 呼叫中心系统和邮件或其他系统提供。电话收集客户反馈的方式成本比较高，一般很少使用。

对于让个人工作成果的衡量一般也分为两个部分，系统自动报告和质量检验员的检验。

（1）系统自动的报告包括，该工程师某个时间段内客户满意度结果，个人 SLA 的完成情况。对于呼叫中心的支持工程师（提供电话支持的工程师），还有某个时间段内的平均通话时长，一共完成的电话数量等的衡量。

（2）质量检验员会定期随机抽取所有支持工程师的电话录音，与客户的通讯邮件或者查看内部系统记录。以特定的模板对此进行评分，定期产生报告。

附录 30–4：组织能力的规划模板 [1]

经营环境	影响公司成败的战略趋势有哪些？ ・技术发展 ・客户和市场变化 ・竞争对手 ・法令改变 ・供应商 ・其他
战略方向	在这些战略趋势下如何取胜？ ・公司想在何处竞争？ 产品 地区市场 目标客户群 ・我们如何超越竞争对手？ 成本领先 技术领先 客户导向 服务 速度 质量 便利性 其他
组织能力	・我们需要何种组织能力？ 确定两三个关键的组织能力 如何衡量这些能力的成功与否
人力资源体系	・人力资源/管理体系如何设计？ 人员配置（招聘、调动、晋升、淘汰） 发展 评估 奖励 组织设计 信息传递

参考文献：

[1] 杨国安，“ 组织能力的 “ 杨三角 ” 企业持续成功的秘诀 ”,，机械工业出版社，2010 年，

[2] 学习型组织，智库百科，2013，http://wiki.mbalib.com/wiki

[3] Bettina Drew ，“ What Are the Characteristics of Customer Service Oriented Workers? ”，2013，eHow.com

作者后记

时间飞逝。冬去春来，现在又是一年的立春。

这本书终于进入最后的排版，即将付梓了。在这近 4 年写作中，发生了很多事情，如移动网络的广泛使用、大数据的出现等。而云计算的服务，也进入了更深的层次。

在这几年中，我们亲眼目睹了美国硅谷的云计算应用正在深刻地影响和改变着众多的行业。在这里，我们看到了云服务的开创者的继续发展，如 Salesforce，CISCO（WebEx），Google，同时我们也见到新兴的应用如 FaceBook 和 LinkedIn（社交），DropBox（存储），Workday（企业 HR 和财务），Zuora（计费）等。即使是对最传统的 IT 产业如电信服务，我们也感受到了云应用带来的冲击。比如 Twilio，它将传统电信运营商的服务转到云服务，包括语音、短信、电话号码等，使得获取电信的服务更加快速。这些日新月异的变化标志着两件事情 ：云服务的深化，和对最传统产业的颠覆。

云计算在中国的发展仍然滞后于硅谷，大部分还停留在 IaaS 阶段。但是，如今令人高兴地看到，我们已逐步抛弃那些浮夸的概念的讨论，越来越多的企业正在进入了云技术的实施和挖掘深层的应用服务。

硅谷是创新与实践的地方。与硅谷相比，我们希望国内有更好的投资环境来鼓励创新，和更多的实干家来开拓云计算的创业实践。

我们这本书致力于帮助云计算的践行者，希望我们的经验、分析和理论可以帮助中国的云计算公司的云服务理念得以真正的实现。

我也想借此机会衷心感谢我的家人、同事和朋友们的热情支持。感谢母校清华大学出版社的责任编辑的指导和编务的辛勤工作。这本书的写作是一个非常艰辛的过程。没有他们的关爱和鼓励，这本书是无法完成。

这里，我对我的父亲陈日浓先生，致以特别感谢。他是一位高级记者，致力于国家的新闻事业。他笔耕不辍，这对于我的写作有着非常积极的影响。在工程师队伍里，只有很少数的人愿意做文字工作，我大概就是这些少数人中之一。这就是我父亲的影响力的结果。

这本书是十多位作者的共同努力的结晶。在此，感谢他们的辛勤的付出和心血 ：李彦涛、刘国萍、杨力伟、朱少民、曹耀和、姜林、刘锦祥、刘芳、王之强、屠昶旸、杨国华、陈广顺、Jessica Hu、汪守金等。同时感谢很多同事在写作上的参与和帮助 ：严石，朱飞，蔡勇，项汀微，田国伟，翟喜贵，苏君福，湛勇，丁艺明等。感谢清华出版社的夏兆彦先生在这本书的写作进程中的大力推进。感谢创想空间商务通信服务公司（全时）的全力支持。

限于水平和时间，书中会有疏漏差误之处，欢迎大家不惜赐教批评指正。如果大家对我们的书有任何反馈和建议，请随时联系。我们的联系方式是 ：

电 子 邮 件 ：chris_cchen0@yahoo.com

微　　信 ：chris_cchen

微博（新浪）：chris_ 运营管理

书评（1）

沈剑

（N2 Studio和Yoyoor创始人,原美国WebEx副总裁）

三年前，当陈赤榕说要编一本关于“云计算”的书，我是既高兴又担心。

高兴的是，我们当年一起在 WebEx（美国网迅）公司建立起来的，能支撑数百万人同时在线网络会议的“实时云计算框架”终于有机会可以揭开面纱，让世人一睹其芳容了；担心的是，时过境迁，当初那些一次又一次的失败教训、一轮又一轮的实战经验，在几年之后，是否能够以全新的姿态来迎接移动互联网时代的挑战？

当我读完陈赤榕的《云计算服务——运营管理和技术架构》后，我是彻底放心和折服了！

书中不仅有 SaaS 和云计算体系的理论基础和前线第一手实践，还包括了以云计算服务为核心的商务运营、技术运营、业务架构、安全体系、组织能力等等必备的实战经验。很明显，这些章节不是学院派人士写的，因为我能看出，每一页都浸透着在实际运营操作中闯荡出来的一线管理者和架构师的血与泪，并通过几年的调整、优化、归纳和总结，最终升华成为这一系列的“云计算军规”。

希望这些“云计算军规”，再次能够帮助大批有志于打造新一代云计算服务平台的中国创业者和创新人才！

书评（2）

Eric S. Yuan,

（Zoom Video Communications创始人及CEO，原CISCO全球副总裁，原美国WebEx副总裁）

云计算始于美国硅谷。云计算早期的两个领头羊是 WebEx 和 Salesforce。1997 年由中国人朱敏在硅谷创建并于 2000 年在 NASDAQ 上市的 WebEx 揭开了云计算这一新型商业和计算模式的序幕。1999 年由美国人 Marc Benioff 在硅谷创建并于 2004 年在 NASDAQ 上市的 Salesfoce 正在把云计算发扬光大。

本书的主要作者陈赤榕亲身参与了 WebEx 全球的服务平台的技术架构和设计，为 WebEx 的成功作出了重大的贡献。过去几年陈赤榕回到中国加入了中国云计算的浪潮，其历时 3 年的杰作《云计算服务——运营管理和技术架构》详细阐述了如何实现云计算的服务。

随着云计算从 2009 年开始在全球快速演进，有很多关于云计算的书籍和文章出现，但本书从两方面的贡献超过了以往任何一本书：

第一点，本书以务实为基础。本书多位作者把多年业界实践经验融会贯通于本书，遵循实践和理论相结合的模式对云计算进行了深入细致的介绍，给读者提供了清晰的思路与脉络。

第二点，本书不仅从云服务客户的角度来看问题，更重要的是从云服务提供商的角度来谈“云落地”的问题，让读者真正的知道怎样地入手和实施云计算的战略，同时也揭开了云计算业界的面纱。

每次 IT 变革，都会带来行业中颠覆性的变化。云计算这一新的变革浪潮刚刚开始，如果你错过了 Internet 的浪潮，可千万别再错过了云计算的大潮。这本书就是你加入云计算大潮的门票。

书评（3）

钱中华

（复星文化产业集团，董事总经理、TMT联席总裁）

云计算是一个过热的话题了。但是，真正能够从产业革命历史的纵深里看懂其宏大叙事，又能从各个行业的应用与云计算的结合实务里探幽发微，很少有人能够胜任。本书做了难能可贵的探讨。

从 1770 年的第一次工业革命，到 1970 年开始的第五次信息技术革命，中间每一次的产业革命都把我们的经济形态带到了全新的阶段。而这种科技迭代进步、以及数字化、网络化的趋势，正在把我们带入第六次产业革命，这次革命给我们带来了全面的数据化、智能化，而实现这一切的基础就是云计算（Cloud Computing）。

以往关于云计算的书籍，主要讲的都是基础概念、架构、形态及应用模式，缺乏从实战落地的角度来深入下去，这也与云计算不算很长的发展历程有关。本书主要作者陈赤榕，从 WebEx 到全时（两家公司都是行业顶尖的基于云的服务商），都是云计算最佳的领导者和实践者。本书全方位地展示了云计算的技术架构、运营及服务管理、安全、合规、成本管理等各方面。细读本书，我们更能体会到陈赤榕所给我们带来的国际领先的概念诠释、应用和具体实战方法。

一个概念到商业模式成型，再到稳定运营、盈利，需要不断地实践和积累，而从陈赤榕的经历来看，显然他找到了捷径。这条捷径也将指引读者在自身的业务及商业运营上利用好云计算，这也是这本书给我们带来的独特的实战价值。

在当今这个智能化的时代，不管从政府管理、到企业运营、到各行各业最终的产品和服务提供，正如我们所见，云计算正用其魅力不断的渗透到所有的领域 ：私有云、公有云、IaaS、PaaS、SaaS……本书，将帮助我们抓住时代的机会！

书评（4）

路鹏 博士

（橡果国际总裁， 原阿里巴巴集团副总裁及淘宝网技术副总裁）

云计算在国内业界的讨论已经有几年了，现在终于进入了真正务实的阶段，或者云计算的实现阶段。

与国内的云计算讨论偏重概念或纯技术不同，作为云计算起源地的美国，云计算更多的是来自商业的应用与运营。早在 2000 年前后，Salesforce（SaaS），WebEx（SaaS），Amazon（IaaS），Google（PaaS）这些云计算的领先者，都是先经历了各自领域的实践，最后完成和推动了整个云计算产业的兴起。

云计算的核心价值在于云服务，真正的难点在于云服务的实现。陈赤榕这本书是国内第一本

系统地讨论如果实现云服务的书。

这本书与其他云计算书的最大不同在于具有特色 ：

（1）书的重心在于云计算怎样的实现或落地，而不是空讲概念 ；

（2）核心内容都是来自实践，以及基于实践上的总结 ；

（3）系统性很强 ：理论分析结合实践经验，框架和条理非常清晰。

此外，这本书是按照教材方式写作 ：重要的章节都有案例研究，将专题的要点和难点做了透彻的阐明。

本书主编及作者陈赤榕长期在美国硅谷，并在 2000 年初就进入了云计算的领域。来自陈赤榕和企业高管与架构师组成的作者群的长期的技术与管理的积累，使得这本书的实践性与理论性都很强。

云计算对国内 IT 业界真正的变革是来自云计算的实施，而不只是概念上的讨论。这本书将会极大地帮助和推动云计算的相关的业界实现这一变革。