Broadview®
www.broadview.com.cn

做个火影般的架构师

ARCHITECTURE DECODING

架构解密

从分布式到微服务

第2版

微服务、云原生、Kubernetes、Service Mesh是分布式领域的热点技术，它们并不是凭空出现的，一定继承了某些“前辈”的优点。我们不仅要了解这些技术，还要深入理解其发展脉络、原理等，才能游刃有余地将其用于现有的项目开发或老系统改造中。

吴治辉 编著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

架构解密

从分布式到微服务

第2版

吴治辉　编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

微服务、云原生、Kubernetes、Service Mesh 是分布式领域的热点技术，它们并不是凭空出现的，一定继承了某些“前辈”的优点。我们不仅要了解这些技术，还要深入理解其发展脉络、原理等，才能游刃有余地将其用于现有的项目开发或老系统改造中。

本书总计 9 章。第 1 章讲解分布式的基础——网络，对国际互联网、NIO、AIO、网络传输中的对象序列化问题、HTTP 的前世今生、TCP/IP、从 CDN 到 SD-WAN 等知识进行深入讲解。第 2 章讲解分布式系统的经典理论，涉及分布式系统的设计理念、一致性原理；ZooKeeper 的使用场景；CAP 理论的前世今生；BASE 准则；分布式事务的原理。第 3 章从 RPC 开始，讲解分布式服务治理框架的起源与原理，并讲解 ZeroC Ice 的原理和微服务架构实战。第 4～6 章以专题形式讲解内存、分布式文件存储和分布式计算，对每个专题都讲解相关的重要理论、产品、开源项目及经验等。第 7 章深入讲解全文检索与消息队列中间件的原理及用法。第 8 章讲解以 Kubernetes 为代表的微服务架构解决了传统架构的哪些痛点；Service Mesh 解决了微服务架构的哪些问题，以及如何理解它的原理和核心内容。第 9 章分享作者的架构实践经验。

不论你是有十几年研发经验及架构经验的 IT 老手，还是刚入门系统架构的 IT 新手，本书都能对你理解分布式架构和微服务架构大有助益。

图书在版编目（CIP）数据

架构解密：从分布式到微服务 / 吴治辉编著. —2 版. —北京：电子工业出版社，2020.6
ISBN 978-7-121-38835-4

Ⅰ. ①架… Ⅱ. ①吴… Ⅲ. ①分布式计算机系统－架构 Ⅳ. ①TP338.8

中国版本图书馆 CIP 数据核字（2020）第 048254 号

责任编辑：张国霞
印　　刷：三河市君旺印务有限公司
装　　订：三河市君旺印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编 100036
开　　本：787×980　1/16　印张：20.5　字数：400 千字
版　　次：2017 年 6 月第 1 版
　　2020 年 6 月第 2 版
印　　次：2020 年 6 月第 1 次印刷
印　　数：5000 册　　定价：89.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819，faq@phei.com.cn。

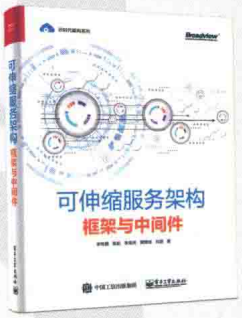
作者简介

吴治辉

HPE资深架构师，拥有近20年软件研发经验，专注于电信软件和云计算方面的软件研发，拥有丰富的大型项目架构设计经验，是业界少有的具备很强Coding能力的资深架构师，也是《ZeroC Ice权威指南》《Kubernetes权威指南》《区块链轻松上手》等书作者。

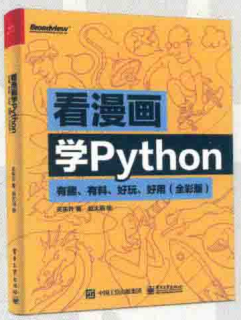
写书、投稿、市场宣传等事宜可联系虾米编辑
邮箱：zhanggx@phei.com.cn
微信：zgx228

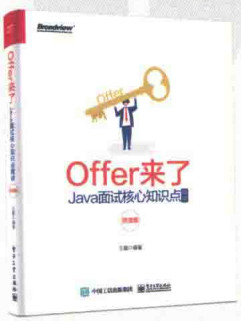
好书分享


可伸缩服务架构
框架与中间件

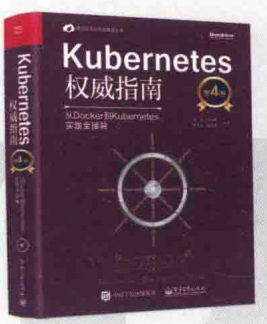

分布式服务架构
原理、设计与实战

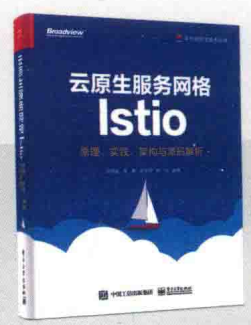

互联网轻量级
SSM框架解密

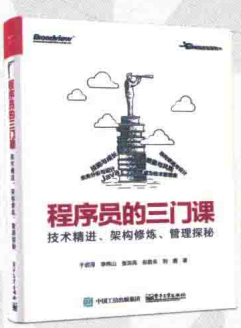

看漫画
学Python
有趣、有料、好玩、好用（全彩版）

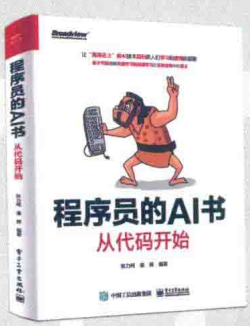

Offer来了
Java面试核心知识点


Offer来了
Java面试核心知识点


Kubernetes
权威指南


云原生服务网格
Istio


程序员的三门课
技术精进、架构修炼、管理探秘


程序员的AI书
从代码开始

目录

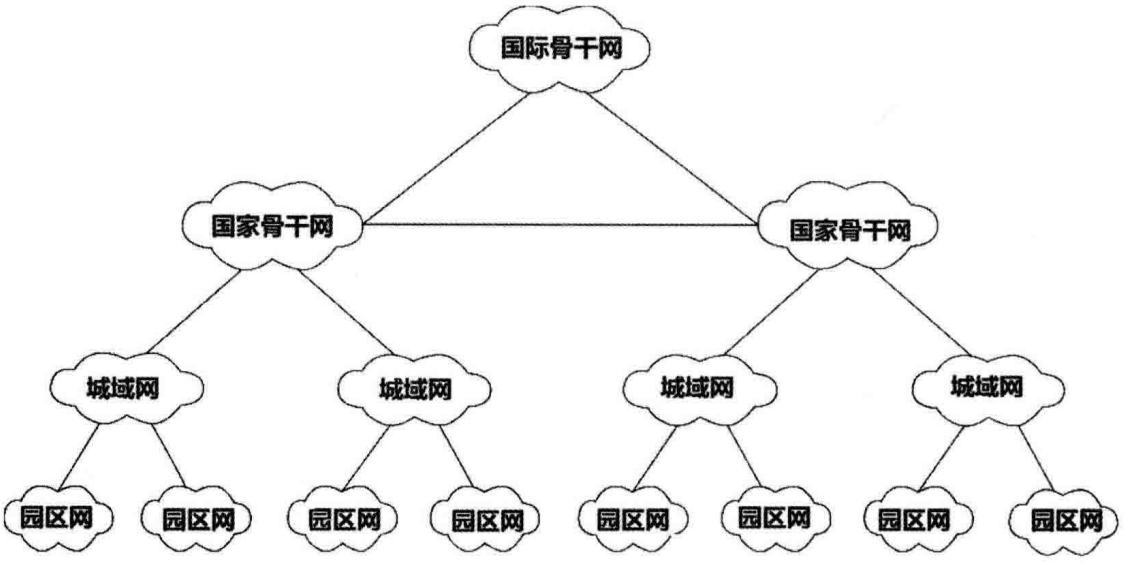
第 1 章

深入理解网络

网络之于分布式系统，就好像双翼之于飞鸟。随着虚拟化、云计算和大数据的不断发展，深入理解网络变得越来越重要。

1.1 从国际互联网开始

几台计算机接在一台交换机上，相互能直接发送信息、传输数据，这样的网络叫作局域网，办公室和家庭里的网络就是典型的局域网。局域网需要上外网的时候，需要电信服务提供商（ISP）提供上网服务，将局域网对接到更大的网络——城域网。连接几个城域网的网络叫作国家骨干网，连接全球骨干网的网络叫作国际骨干网。如下所示是国际互联网的一个示意图，国际互联网是一个分层汇聚网络，位于顶端的是国际骨干网，负责连接国家骨干网，在一些国家之间还有直达通道。在国家内部通常有一个全国性的高速国家骨干网，这个骨干网只能在某些点对接国际骨干网。国家骨干网负责将分布在各个城市里的城域网连接起来，每个城域网则负责将本区域众多园区网接入，这些园区网可以是省内某些高新产业园的网络、一些大的 IT 公司的网络等。

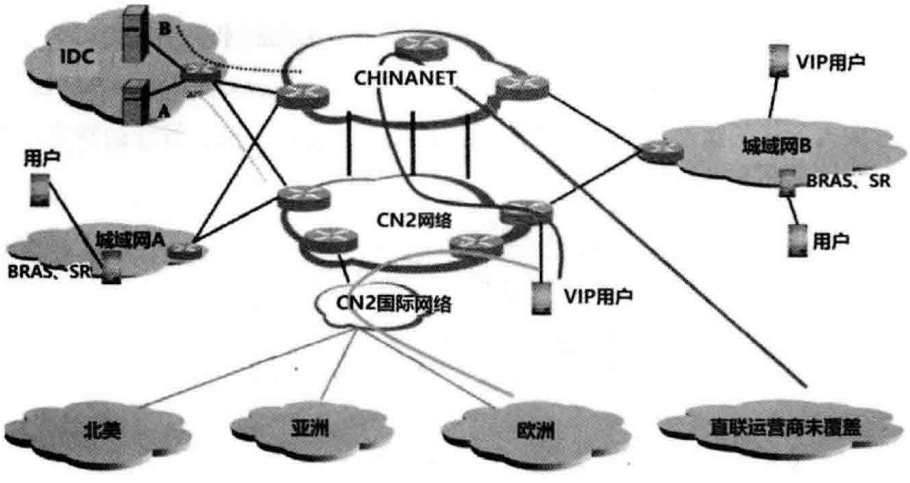


首先说说位于顶端的国际骨干网。为了将地球上的各个大洲互联，人们建设了很多海底光缆，可以说海底光缆构成了国际骨干网的骨架。美国是国际互联网的中心，它周边有丰富的海底光缆，直达各个大洲。中国大陆地区的海底光缆连接点有三个：青岛、上海和汕头，总共有 6 条光缆通向全球。

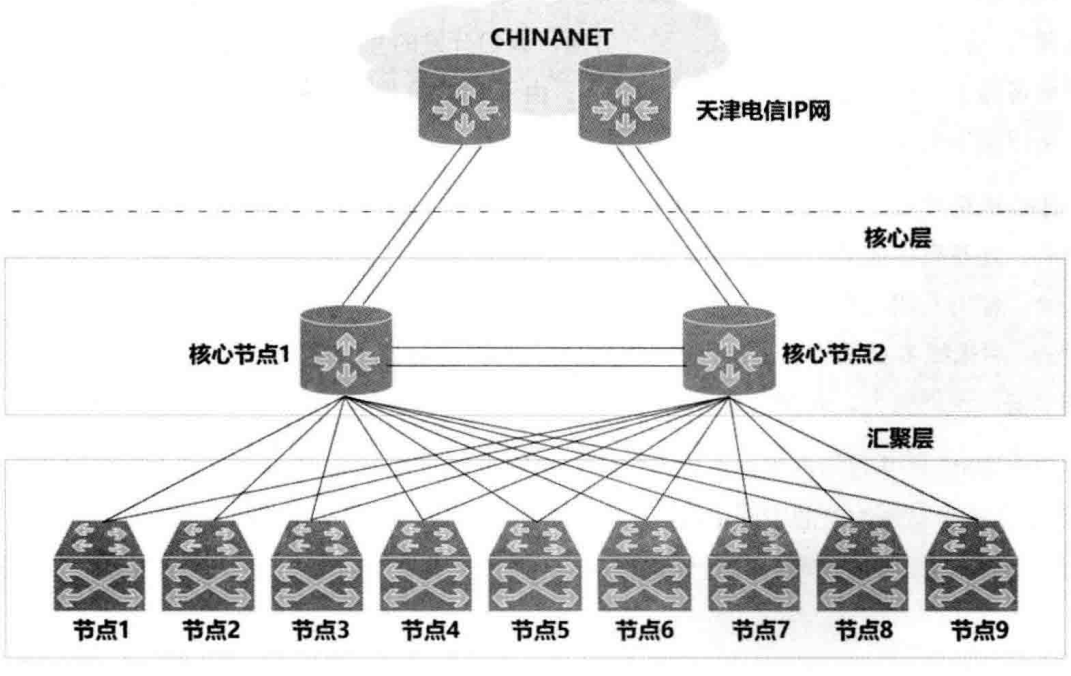
接下来说说国家骨干网。以中国为例，中国的全国性骨干网是 CHINANET，它是前邮电部经营管理的网络，1995 年年初与国际互联网连通并向社会提供服务。类似于国际互联网的架构，CHINANET 也是分层网络，由骨干网和汇接层两部分组成：骨干网是其主要信息通路，由直辖市和各省会城市的网络节点构成；汇接层则用来连接各省（区）的城域网。

CHINANET 由 8 个核心节点组成，这 8 个核心节点分别是北京、上海、广州、沈阳、南京、武汉、成都、西安。毫无悬念，“北上广”3 个节点成为超级节点，也是 CHINANET 的 3 个国际出口，在这 3 个超级节点之间形成大三角电路，其他 5 个普通节点则与每个超级节点互联。

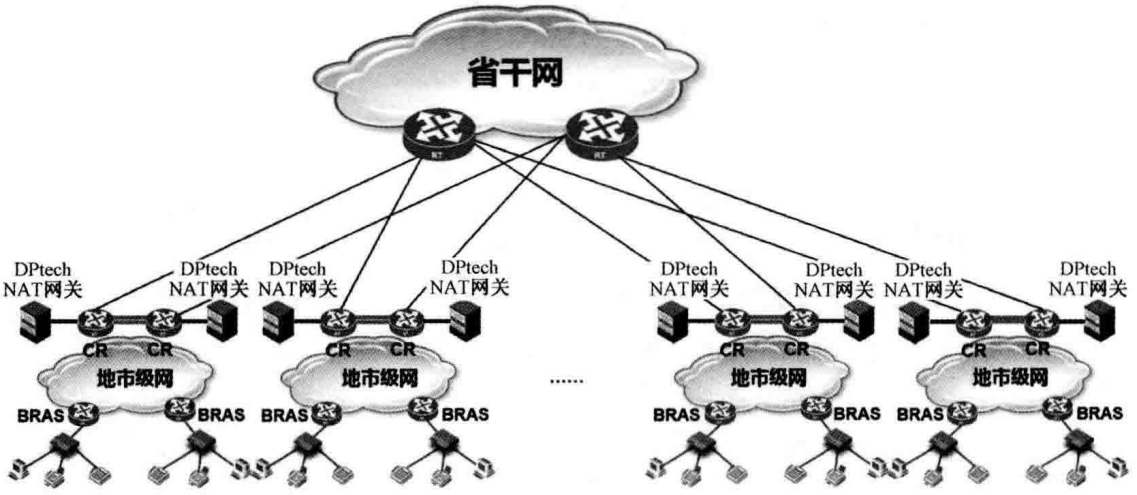
为了保证国内企业用户访问国外网站的速度和带宽，中国电信后来启动了另外一个精品网络，即 CN2 网络。从下图中可以看到 CN2 网络专门增加了海外直连线路，以保证海外站点的带宽和品质。



再说说 CHINANET 的“汇接层”网络，这一层由 54 个汇接节点组成，除甘肃、山西、新疆、宁夏、贵州、青海、西藏、内蒙古等 8 个省份的单汇接节点外，每个省都有两个骨干网汇接节点，在第一和第二出入口节点之间通过一条省内中继连接。各省份双方向上连接，分别连接到一个超级核心节点和一个普通核心节点。如下所示是天津电信城域网汇聚接入 CHINANET 的网络示意图。



最后说说“城域网”。城域网也是一个分层汇聚网络，主要用来提供宽带接入服务，因此存在一种特殊设备，即宽带远程接入服务器 BRAS（Broadband Remote Access Server），它是用来完成各种宽带接入方式的宽带网络用户的接入、认证、计费、控制、管理的网络设备，是宽带网络可运营、可管理的基石。



我们如果在家上网，则通常会通过当地某个宽带运营商的网络接入城域网，最终实现“全球互联”，这是我们都熟悉的方式。另一方面，我们开发的互联网应用被部署到 IDC 机房里的某个服务器上，从而完成应用互联网的接入。由于 IDC 机房与我们所开发的分布式应用密切相关，所以接下来我们一起了解 IDC 机房的相关知识。

IDC 机房又被称为互联网数据中心（Internet Data Center）或者数据中心，不仅是数据存储的中心，还是数据流通的中心。IDC 机房是标准化的电信专业级机房，为企业、政府提供服务器托管、租用及相关增值等方面的全方位服务。一开始，IDC 机房主要是电信、联通等运营商建设的，后来很多企业也有了自己的 IDC 机房，BAT 都自建了 IDC 机房，比如腾讯先后自建了深圳宝安、深圳腾大、天津三个 IDC 机房。

由于 2002 年 5 月国内电信业大重组，原中国电信北方 10 个省份正式划入中国网通集团，南方 21 个省份重组为新的中国电信。这将把中国的互联网一分为二，导致互联互不通，使国内的 IDC 机房往往具备双线接入这一奇特特性。

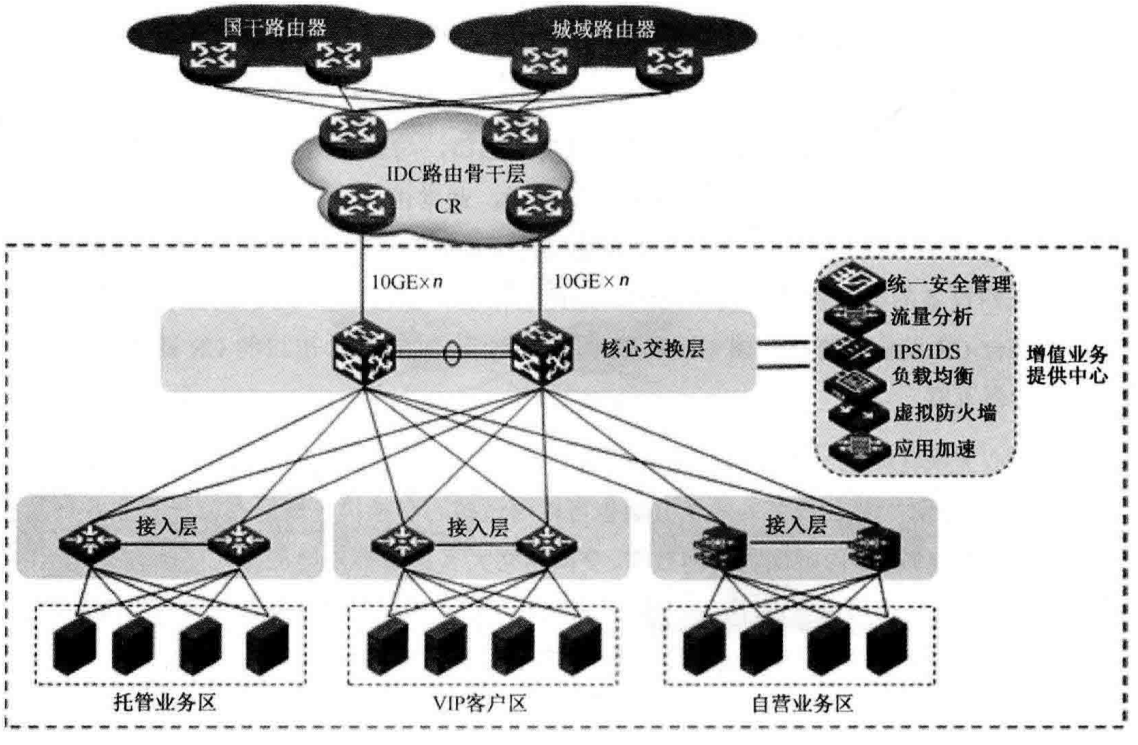
如下所示是 IDC 机房的网络架构图，一般分为出口路由区、核心交换区、接入网络区及增值业务区四个区域。

（1）出口路由区的主要功能是作为 IDC 机房的出口，与国干网（CHINANET）和本身的城域网互联，完成外部网络和 IDC 内网的三层互通，通常由两台 CR 路由器组成。某些大型省份通常会建设多个 IDC 机房，若每个 IDC 机房之间都与国干网和城域网互联，则会浪费国干网和城域网设备的端口资源和线路资源，因此通常会再建设一个 IDC 路由骨干层网络。骨干层中的两台 CR 路由器直接与国干网和城域网互联，各机房 IDC 出口的 CR 路由器则与骨干层出口路由器互联。

（2）核心交换区则由一组核心交换机组成，作为接入层与出口路由区的互联设备，起到汇聚流量的作用，同时 IDC 内部的流量互通也可以通过核心交换机完成。在云计算业务兴起后，为了扩大二层网络规模，同时提高内网效率，交换网络大多采用核心层加接入层的扁平化组网，不再设置汇聚层。为了实现高密接入，核心交换机通常采用数据中心级设备，具有高吞吐、大缓存等特点，同时通过 IRF2 网络虚拟化技术将多台核心交换机虚拟成一台，既提高接入密度，又方便管理。

（3）接入网络区下连物理服务器，上连核心交换机，主要部署千兆或万兆交换机。物理服务器数量多，且每台物理服务器均有多个端口，这就要求接入层交换机实现高密接入。当前在 IDC 网络中，接入交换机通常以 TOR 方式在每个机柜都部署两台，实现本机柜的服务器接入。接入交换机通常采用千兆下行（连接服务器）、万兆上行（连接核心交换机）的连接方式，并通过 IRF2 技术进行虚拟化部署。

（4）增值业务区部署与增值业务相关的设备，包括防火墙、IPS、负载均衡等设备。这些设备通常以旁挂核心交换机的方式进行设计，根据业务需求，在核心交换机上将流量引到增值业务区处理。对于云主机等业务，由于规划使用私网 IP 网段，因此必须使用防火墙实现 NAT 转换，该防火墙设备通常也以旁挂方式部署在核心交换机上。



在IDC机房里通常采用传统的VLAN技术实现租户网络的隔离，VLAN基于IEEE的802.1Q协议，在该协议的帧格式里面定义了 VLAN ID 的位数为 12 比特，因此最多只能支持 4094 个VLAN。而随着云数据中心的各种业务应用的规模落地，业务量不断增长，就可能需要成千上万个 VLAN，传统 VLAN 的数量不能满足云数据中心日后业务规模发展的需求。另外，在物理服务器被虚拟化后，云数据中心内部虚拟机的数量相比原有的物理机发生了数量级的增加，与之对应的虚拟机虚拟网卡的 MAC 地址数量也相应增加，这对云数据中心接入区网络的交换机地址容量能力产生了很大冲击。虚拟机数量很多时，会导致交换机的 MAC 地址表溢出，从而导致数据帧的丢弃或者产生大量的广播帧，严重影响网络的性能。最后，云数据中心的虚拟机通常需要在一定范围内迁移，在传统的 VLAN 网络下，虚拟机只能在二层网络下迁移，并且为了能够支持虚拟机的迁移，需要在二层网络中对VLAN进行预配置，这造成了VLAN配置混乱，影响了 VLAN 广播域的隔离，降低了网络的效率。VXLAN（Virtual eXtensible Local Area Network）虚拟扩展局域网是一种“VLAN 升级技术”，是一种大二层虚拟网络扩展的隧道封装技术，可以很好地解决上述问题，目前该技术已经成为各种规模化运营的云数据中心不可忽视的关键应用技术。

VXLAN 是 VMware、思科、Arista、Broadcom、Citrix 和 RedHat 共同提出的 IETF 草案，可以通过软件的方式来实现支持，其中最重要的开源软件交换机是 Open vSwitch，它也是虚拟化网络解决方案中最重要的一个开源软件，OpenStack、Docker 都可以用它实现虚拟网络。

1.2　NIO，一本难念的经

我们知道，分布式系统的基础是网络。因此，网络编程是分布式软件工程师和架构师的必备技能之一，而且随着当前大数据和实时计算技术的兴起，高性能 RPC 架构与网络编程技术再次成为焦点。不管是 RPC 领域的 ZeroC Ice、Thrift，还是经典分布式框架 Actor 模型中的 Akka，或者实时流领域的 Storm、Spark、Flink，又或者开源分布式数据库中的 Mycat、VoltDB，这些高大上产品的底层通信技术都采用了 NIO（非阻塞通信）通信技术。而 Java 领域里大名鼎鼎的 NIO 框架—— Netty，则被众多的开源项目或商业软件所采用。

相对于它的老前辈 BIO（阻塞通信）来说，NIO 模型非常复杂，以至于我们难以精通它，难以编写出一个没有缺陷、高效且适应各种意外情况的稳定的 NIO 通信模块。之所以会出现这样的问题，是因为 NIO 编程不是单纯的一个技术点，而是涵盖了一系列相关技术、专业知识、编程经验和编程技巧的复杂工程。

1.2.1　难懂的 ByteBuffer

Java NIO 抛弃了我们所熟悉的 Stream、byte[]等数据结构，设计了一个全新的数据结构——ByteBuffer，ByteBuffer 的主要使用场景是保存从 Socket 中读取的输入字节流并循环利用，以减少 GC 的压力。Java NIO 功能强大，但难以掌握。以经典的 Echo 服务器为例，其核心是读入客户端发来的数据，并且回写给客户端，这段代码用 ByteBuffer 来实现，大致就是下面的逻辑：

```
byteBuffer = ByteBuffer.allocate(N);
//读取数据，写入 byteBuffer
readableByteChannel.read(byteBuffer);
//读取 byteBuffer，写入 Channel
writableByteChannel.write(byteBuffer);
```

如果我们能马上发现在上述代码中存在一个严重缺陷且无法正常工作，那么说明我们的确