腾云

—云计算网络技术揭秘

徐立冰

201208

目　录

第一篇　缘起篇

第 1 章　云计算的兴起 11
1.1　云计算的身世 11
1.1.1　70 年前的畅想 11
1.1.2　世纪之交的互联网盛宴 12
1.1.3　世纪初的爆发 14
1.1.4　百花齐放的年代 15
1.2　云计算的 DNA 16
1.3　云计算的基本元素 17
1.3.1　自助服务 17
1.3.2　通过网络分发服务 18
1.3.3　资源池化 19
1.3.4　资源的灵活调度 19
1.3.5　可衡量的服务 20
1.4　IaaS/PaaS/SaaS，它们都是什么？ 21
1.4.1　基础架构即服务/IaaS 21
1.4.2　平台即服务/PaaS 22
1.4.3　软件即服务/SaaS 22
1.5　私有云/社区云/公有云/混合云 22
1.5.1　私有云 22
1.5.2　公有云 23
1.5.3　社区云 23
1.5.4　混合云 23
1.6　云计算的独有优势 24
1.6.1　降低成本 24
1.6.2　扩展性 25
1.6.3　高可靠性 25
1.6.4　远程访问 26
1.6.5　模块化 26
1.6.6　高等级服务 27
第 2 章　云与网的关系 ..... **Error! Bookmark not defined.**
2.1　外延与内涵—云计算网络分类 ..... **Error! Bookmark not defined.**
2.2　外延—可靠、安全、灵活的网络通道 ..... **Error! Bookmark not defined.**
2.2.1　可靠的网络 ..... **Error! Bookmark not defined.**
2.2.2　安全的网络 ..... **Error! Bookmark not defined.**
2.2.3　灵活的网络 ..... **Error! Bookmark not defined.**
2.3　内涵—下一代数据中心网络平台 ..... **Error! Bookmark not defined.**
2.4　支持云网络的新型网络设备 ..... **Error! Bookmark not defined.**

第 3 章 安全的网络通道（1）—网络准入 ..... Error! Bookmark not defined.
3.1 为什么安全是云计算的基础？ ..... Error! Bookmark not defined.
3.2 云计算安全的发展现状 ..... Error! Bookmark not defined.
3.3 网络在云计算安全防护中扮演的角色 ..... Error! Bookmark not defined.
3.4 企业网络准入技术分类 ..... Error! Bookmark not defined.
3.5 二层准入 ..... Error! Bookmark not defined.
3.6 三层准入 ..... Error! Bookmark not defined.
3.7 客户端方式 ..... Error! Bookmark not defined.
3.8 二层准入 VS. 三层准入 VS. 客户端 ..... Error! Bookmark not defined.
3.8.1 二层准入的特点 ..... Error! Bookmark not defined.
3.8.2 三层准入的特点 ..... Error! Bookmark not defined.
3.8.3 客户端方式的特点 ..... Error! Bookmark not defined.
3.9 最终用户需要什么样的方案？ ..... Error! Bookmark not defined.
3.10 IT 部门需要什么样的方案？ ..... Error! Bookmark not defined.
3.11 什么是完美的产品？ ..... Error! Bookmark not defined.
3.12 虚拟桌面的机会 ..... Error! Bookmark not defined.
第 4 章 安全的网络通道（2）—网络加密 ..... Error! Bookmark not defined.
4.1 通过 VPN 保证云计算的数据安全 ..... Error! Bookmark not defined.
4.2 VPN 技术选择—SSL 胜出 ..... Error! Bookmark not defined.
4.3 让 SSL 胜出的独门绝技 ..... Error! Bookmark not defined.
4.4 SSL 的技术实现 ..... Error! Bookmark not defined.
4.4.1 SSL 握手协议 ..... Error! Bookmark not defined.
4.4.2 SSL 记录协议 ..... Error! Bookmark not defined.
4.4.3 SSL 记录头的格式 ..... Error! Bookmark not defined.
4.4.4 SSL 记录记录数据的格式 ..... Error! Bookmark not defined.
4.5 几种 SSL VPN 类型 ..... Error! Bookmark not defined.
4.5.1 零客户端模式 ..... Error! Bookmark not defined.
4.5.2 瘦客户端模式 ..... Error! Bookmark not defined.
4.5.3 隧道模式 ..... Error! Bookmark not defined.
4.6 SSL 的后续发展—DTLS/TLS ..... Error! Bookmark not defined.
第 5 章 可靠的网络通道 ..... Error! Bookmark not defined.
5.1 云服务的用户体验与网络服务质量 ..... Error! Bookmark not defined.
5.2 云计算流量的分类 ..... Error! Bookmark not defined.
5.3 针对云计算的网络服务质量设计 ..... Error! Bookmark not defined.
5.3.1 流量识别 ..... Error! Bookmark not defined.
5.3.2 流量标记 ..... Error! Bookmark not defined.
5.3.3 流量处理 ..... Error! Bookmark not defined.
第 6 章 灵活的网络通道 28
6.1 移动性是云计算网络的基本特征 28
6.2 现有解决方案 1—DNS 重定向 29
6.3 现有解决方案 2—健康路由注入 30
6.4 LISP 是对现有方案的改进 31
6.5 LISP 的核心思想－－Map-and-encap 32
6.6 LISP 的基本架构 33
6.7 LISP 的新包头 34
6.8 通过 LISP-ALT 实现可扩展网络 35

6.9 一个 LISP 转发实例 35
6.10 LISP 的应用场景 36
6.10.1 IP 终端的灵活移动 36
6.10.2 IPv6-IPv4 混合部署 37
6.10.3 多租户 VPN 环境 39
6.11 我们真的需要 LISP 吗？ 39
第 7 章 TRILL/FabricPath 和 SPB—从“交换”到“路由” ..... **Error! Bookmark not defined.**
7.1 二层网络的困境 ..... **Error! Bookmark not defined.**
7.2 为什么传统二层网络不给力？ ..... **Error! Bookmark not defined.**
7.3 FabricPath 的目标 ..... **Error! Bookmark not defined.**
7.4 FabricPath 的实现：新的控制平面 ..... **Error! Bookmark not defined.**
7.4.1 新增一个二层帧头 ..... **Error! Bookmark not defined.**
7.4.2 增加一套简化的 IS-IS 路由协议 ..... **Error! Bookmark not defined.**
7.5 插播一个问题：为什么要新的地址空间？ ..... **Error! Bookmark not defined.**
7.6 FabricPath 的工作模式 ..... **Error! Bookmark not defined.**
7.7 第二个问题：现有技术不足够吗？ ..... **Error! Bookmark not defined.**
7.8 TRILL－－FabricPath 的公开标准 ..... **Error! Bookmark not defined.**
7.9 另一个 TRILL—SPB ..... **Error! Bookmark not defined.**
7.10 TRILL VS SPB ..... **Error! Bookmark not defined.**
第 8 章 利用以太传输存储流量—FCoE ..... **Error! Bookmark not defined.**
8.1 存储与网络的关系 ..... **Error! Bookmark not defined.**
8.2 传统存储网络面临的挑战 ..... **Error! Bookmark not defined.**
8.3 融合！FCoE 给出的解决方案 ..... **Error! Bookmark not defined.**
8.4 FCoE 的基本面 ..... **Error! Bookmark not defined.**
8.5 给以太网动手术—FCoE 的数据平面 ..... **Error! Bookmark not defined.**
8.5.1 PFC—不丢包以太网 ..... **Error! Bookmark not defined.**
8.5.2 ETS—灵活带宽调度 ..... **Error! Bookmark not defined.**
8.5.3 DCBX—与现有环境的兼容性 ..... **Error! Bookmark not defined.**
8.6 连接两个世界的 FIP—FCoE 的控制平面 ..... **Error! Bookmark not defined.**
8.6.1 FCoE VLAN 发现 ..... **Error! Bookmark not defined.**
8.6.2 FLOGI 注册和 FPMA ..... **Error! Bookmark not defined.**
8.7 典型的 FCoE 网络架构 ..... **Error! Bookmark not defined.**
8.8 FCoE 架构中的两种设备类型 ..... **Error! Bookmark not defined.**
8.8.1 ENode（FCoE Node） ..... **Error! Bookmark not defined.**
8.8.2 FCF（FCoE Forwarder） ..... **Error! Bookmark not defined.**
8.9 FCoE 的演化：Multi Hop FCoE ..... **Error! Bookmark not defined.**
8.9.1 纯以太网模式 ..... **Error! Bookmark not defined.**
8.9.2 FIP Snooping 模式 ..... **Error! Bookmark not defined.**
8.9.3 NPV 模式 ..... **Error! Bookmark not defined.**
8.9.4 VE_Port 互联模式 ..... **Error! Bookmark not defined.**
8.10 一个 FCoE 数据帧的转发过程 ..... **Error! Bookmark not defined.**
8.11 FCoE 的标准化与市场化进程 ..... **Error! Bookmark not defined.**
8.12 iSCSI 行不行？非 FCoE 不可吗？ ..... **Error! Bookmark not defined.**
第 9 章 连接虚拟机的交换机 ..... **Error! Bookmark not defined.**
9.1 为什么虚拟化数据中心需要一台新的交换机 ..... **Error! Bookmark not defined.**
9.2 仅仅在服务器内部实现简单交换是不够的 ..... **Error! Bookmark not defined.**

9.2.1 软件 VEB ..... Error! Bookmark not defined.
9.2.2 硬件 VEB ..... Error! Bookmark not defined.
9.3 VN-Tag—为虚拟机打上网络标签 ..... Error! Bookmark not defined.
9.4 一个 VN-tag 交换实例 ..... Error! Bookmark not defined.
9.5 基于 VN-Tag 的新一代网络设备 ..... Error! Bookmark not defined.
9.5.1 VN-Tag 网卡 ..... Error! Bookmark not defined.
9.5.2 VN-Tag 交换机 ..... Error! Bookmark not defined.
9.5.3 操作系统支持 ..... Error! Bookmark not defined.
9.6 VN-Tag 之外的选择—VEPA ..... Error! Bookmark not defined.
9.6.1 标准版 VEPA ..... Error! Bookmark not defined.
9.6.2 增强版 VEPA ..... Error! Bookmark not defined.
9.7 VEPA 交换机扫描 ..... Error! Bookmark not defined.
9.7.1 HP 5900 ..... Error! Bookmark not defined.
9.7.2 Juniper QFabric ..... Error! Bookmark not defined.
9.7.3 Juniper EX4500&EX8200 ..... Error! Bookmark not defined.
9.7.4 Extreme Summit X670 ..... Error! Bookmark not defined.
9.8 VN-Tag VS VEPA ..... Error! Bookmark not defined.
第 10 章 虚拟化的最后一公里－－虚拟化网卡 ..... Error! Bookmark not defined.
10.1 补齐虚拟化的最后一公里 ..... Error! Bookmark not defined.
10.2 什么是虚拟化网卡？ ..... Error! Bookmark not defined.
10.2.1 什么是虚拟接入 ..... Error! Bookmark not defined.
10.2.2 什么是虚拟通道 ..... Error! Bookmark not defined.
10.3 利用 SR-IOV 实现虚拟化网卡 ..... Error! Bookmark not defined.
10.4 SR-IOV 的实践者－－Palo ..... Error! Bookmark not defined.
10.5 将 SR-IOV 带入现实的辅助技术 ..... Error! Bookmark not defined.
10.6 更加彻底的虚拟化－－MR-IOV ..... Error! Bookmark not defined.
10.7 后面的故事 ..... Error! Bookmark not defined.
第 11 章 数据中心互联设计—更广泛的二层网络 ..... Error! Bookmark not defined.
11.1 数据中心二层互联的需求 ..... Error! Bookmark not defined.
11.2 通过 VPLS 实现互联 ..... Error! Bookmark not defined.
11.3 一个 VPLS 转发实例 ..... Error! Bookmark not defined.
11.4 VPLS 的限制 ..... Error! Bookmark not defined.
11.4.1 缺乏对局域网的优化 ..... Error! Bookmark not defined.
11.4.2 依赖运营商资源 ..... Error! Bookmark not defined.
11.4.3 配置复杂 ..... Error! Bookmark not defined.
11.5 通过 OTV 实现互联 ..... Error! Bookmark not defined.
11.5.1 OTV 的数据平面 ..... Error! Bookmark not defined.
11.5.2 OTV 的控制平面 ..... Error! Bookmark not defined.
11.6 OTV 对二层协议的优化 ..... Error! Bookmark not defined.
11.7 OTV 对三层网关的优化 ..... Error! Bookmark not defined.
11.8 OTV 环境下的多接入和流量负载均衡 ..... Error! Bookmark not defined.
11.9 LISP VS VPLS ..... Error! Bookmark not defined.
11.10 LISP 与 OTV 的关系 ..... Error! Bookmark not defined.
第 12 章 自定义网络—OpenFlow&SDN ..... Error! Bookmark not defined.
12.1 通过软件定义网络—SDN ..... Error! Bookmark not defined.
12.2 实验室中走出的 OpenFlow ..... Error! Bookmark not defined.

12.3 OpenFlow 的系统模型.....Error! Bookmark not defined.
12.4 OpenFlow 交换机基本组成.....Error! Bookmark not defined.
12.5 两种 OpenFlow 交换机.....Error! Bookmark not defined.
12.5.1 OpenFlow 专用交换机.....Error! Bookmark not defined.
12.5.2 OpenFlow 兼容型交换机.....Error! Bookmark not defined.
12.6 OpenFlow 中央控制器.....Error! Bookmark not defined.
12.6.1 控制器的主动工作模式.....Error! Bookmark not defined.
12.6.2 控制器的被动工作模式.....Error! Bookmark not defined.
12.7 一个 OpenFlow 实例.....Error! Bookmark not defined.
12.8 什么是 OF-Config？.....Error! Bookmark not defined.
12.8.1 OF-Config 解决的问题.....Error! Bookmark not defined.
12.8.2 OF-Config 的功能描述.....Error! Bookmark not defined.
12.9 OpenFlow 的近亲.....Error! Bookmark not defined.
12.9.1 分布式转发模块化交换机.....Error! Bookmark not defined.
12.9.2 远端板卡.....Error! Bookmark not defined.
12.9.3 Nexus 1000v.....Error! Bookmark not defined.
12.9.4 Open vSwitch.....Error! Bookmark not defined.
12.9.5 EEM.....Error! Bookmark not defined.
12.10 Google 的 OpenFlow 实践.....Error! Bookmark not defined.
12.11 网络厂家的 SDN 战略.....Error! Bookmark not defined.
12.11.1 NEC 的 OpenFlow 战略.....Error! Bookmark not defined.
12.11.2 HP 的 OpenFlow 战略.....Error! Bookmark not defined.
12.11.3 Juniper 的 OpenFlow 战略.....Error! Bookmark not defined.
12.11.4 Nicira 的 OpenFlow 战略.....Error! Bookmark not defined.
12.11.5 Cisco 的 OpenFlow 战略.....Error! Bookmark not defined.
12.12 SDN/OpenFlow 的前景.....Error! Bookmark not defined.
第 13 章 更大的云——VXLAN.....Error! Bookmark not defined.
13.1 VXLAN 要解决的问题.....Error! Bookmark not defined.
13.2 VXLAN 的新头部.....Error! Bookmark not defined.
13.2.1 VXLAN 包头.....Error! Bookmark not defined.
13.2.2 UDP 头部.....Error! Bookmark not defined.
13.2.3 外层三层包头.....Error! Bookmark not defined.
13.2.4 外层二层帧头.....Error! Bookmark not defined.
13.3 VXLAN 的数据平面—隧道机制.....Error! Bookmark not defined.
13.3.1 隧道机制减小对现网的改动.....Error! Bookmark not defined.
13.3.2 隧道机制对快速变更的支持.....Error! Bookmark not defined.
13.4 VXLAN 的控制平面—改进的二层协议.....Error! Bookmark not defined.
13.5 纯 VXLAN 部署场景.....Error! Bookmark not defined.
13.6 VXLAN 与非 VXLAN 混合部署.....Error! Bookmark not defined.
13.7 一个 VXLAN 转发实例.....Error! Bookmark not defined.
13.7.1 第一阶段——ARP 请求.....Error! Bookmark not defined.
13.7.2 第二阶段——数据传输.....Error! Bookmark not defined.
13.8 VXLAN、OTV、LISP，他们都有什么关系？.....Error! Bookmark not defined.
13.9 Microsoft 的算盘—NVGRE.....Error! Bookmark not defined.
第 14 章 桌面虚拟化网络漫谈.....Error! Bookmark not defined.
14.1 桌面虚拟化的前身.....Error! Bookmark not defined.

14.2 虚拟桌面的诞生 ..... Error! Bookmark not defined.
14.3 虚拟桌面是怎样工作的 ..... Error! Bookmark not defined.
14.3.1 集中托管方式 ..... Error! Bookmark not defined.
14.3.2 远程同步方式 ..... Error! Bookmark not defined.
14.4 虚拟桌面的客户端类型 ..... Error! Bookmark not defined.
14.4.1 零客户端虚拟桌面 ..... Error! Bookmark not defined.
14.4.2 瘦客户端虚拟桌面 ..... Error! Bookmark not defined.
14.4.3 胖客户端虚拟桌面 ..... Error! Bookmark not defined.
14.5 一个典型的虚拟桌面后台架构 ..... Error! Bookmark not defined.
14.6 决定虚拟桌面的成败—用网络替代 VGA 线缆 ..... Error! Bookmark not defined.
14.7 虚拟桌面的核心网络技术—网络显示协议 ..... Error! Bookmark not defined.
14.8 网络显示协议三大要素 ..... Error! Bookmark not defined.
14.8.1 网络资源 ..... Error! Bookmark not defined.
14.8.2 用户体验 ..... Error! Bookmark not defined.
14.8.3 CPU 占用率 ..... Error! Bookmark not defined.
14.9 显示协议—兵家必争之地 ..... Error! Bookmark not defined.
14.10 老牌显示协议—RDP ..... Error! Bookmark not defined.
14.11 显示协议的王者—HDX/ICA ..... Error! Bookmark not defined.
14.12 后起之秀—PCoIP ..... Error! Bookmark not defined.
14.13 HDX VS RDP VS PCoIP 谁主沉浮 ..... Error! Bookmark not defined.
第 15 章 Big Data 网络设计要点 ..... Error! Bookmark not defined.
15.1 Big Data 的产生 ..... Error! Bookmark not defined.
15.2 全新的 Big Data ..... Error! Bookmark not defined.
15.3 MapReduce 的原理 ..... Error! Bookmark not defined.
15.4 MapReduce 的业务流程 ..... Error! Bookmark not defined.
15.5 写入数据过程中的网络流量模型 ..... Error! Bookmark not defined.
15.6 MapReduce 算法过程中的网络流量模型 ..... Error! Bookmark not defined.
15.6.1 Map 过程 ..... Error! Bookmark not defined.
15.6.2 Shuffle 过程 ..... Error! Bookmark not defined.
15.6.3 Reduce 过程 ..... Error! Bookmark not defined.
15.6.4 OutPut 过程 ..... Error! Bookmark not defined.
15.7 读取数据过程中的网络流量模型 ..... Error! Bookmark not defined.
15.8 MapReduce 网络模型综述 ..... Error! Bookmark not defined.
第 16 章 怎样将服务器接入网络 ..... Error! Bookmark not defined.
16.1 ToR 和 EoR ..... Error! Bookmark not defined.
16.2 从增加一台服务器到增加一个机柜的服务器 ..... Error! Bookmark not defined.
16.3 鱼与熊掌不可兼得？ ..... Error! Bookmark not defined.
16.4 Cisco 的提案—FEX 远端板卡 ..... Error! Bookmark not defined.
16.5 Juniper 的尝试—QFabric ..... Error! Bookmark not defined.
第 17 章 VOQ 解密 ..... Error! Bookmark not defined.
17.1 头端阻塞是实现 DCE 交换机的障碍 ..... Error! Bookmark not defined.
17.2 利用 VOQ 防止头端阻塞 ..... Error! Bookmark not defined.
17.3 针对组播的 VOQ 设计 ..... Error! Bookmark not defined.
17.4 VOQ 的产业化发展 ..... Error! Bookmark not defined.
第 18 章 刀片服务器网络 ..... Error! Bookmark not defined.
18.1 刀片服务器渊源 ..... Error! Bookmark not defined.

18.2 刀片服务器同传统 ToR 接入的区别 ..... Error! Bookmark not defined.
18.3 把握刀片服务器的网络设计 ..... Error! Bookmark not defined.
18.3.1 直通模块 ..... Error! Bookmark not defined.
18.3.2 交换模块 ..... Error! Bookmark not defined.
18.3.3 集中接入模式 ..... Error! Bookmark not defined.
第 19 章 千兆不够，要万兆！ ..... Error! Bookmark not defined.
19.1 千兆到万兆的质变 ..... Error! Bookmark not defined.
19.1.1 万兆网络是 FCoE 的基础 ..... Error! Bookmark not defined.
19.1.2 更高的传输效率 ..... Error! Bookmark not defined.
19.1.3 助推虚拟化 ..... Error! Bookmark not defined.
19.2 万兆以太网标准现状 ..... Error! Bookmark not defined.
19.3 盘点万兆以太网交换机 ..... Error! Bookmark not defined.
19.3.1 Cisco Catalyst 6500 ..... Error! Bookmark not defined.
19.3.2 Cisco Nexus 7000 ..... Error! Bookmark not defined.
19.3.3 H3C 12500 ..... Error! Bookmark not defined.
19.3.4 H3C 10500 ..... Error! Bookmark not defined.
19.3.5 Juniper QFabric ..... Error! Bookmark not defined.
19.3.6 华为 CloudEngine 12800 ..... Error! Bookmark not defined.
19.3.7 DELL Force10 E1200i ..... Error! Bookmark not defined.
19.3.8 Brocade BigIron RX ..... Error! Bookmark not defined.
19.3.9 Extreme X8 ..... Error! Bookmark not defined.
19.3.10 Arista 7500 ..... Error! Bookmark not defined.
19.3.11 AVAYA 8800 ..... Error! Bookmark not defined.
19.3.12 Alcatel-Lucent OmniSwitch 10K ..... Error! Bookmark not defined.
19.3.13 锐捷 RG-S12000 ..... Error! Bookmark not defined.

前言

我真正系统接触网络技术是从一本名叫《CCNA 学习指南》（Richard Deal 著）的书开始的，此书并不是思科的官方教材，在亚马逊网上书店的 CCNA 相关搜索结果中甚至排不进前三页，之后我又阅读了被誉为“网络圣经”的《TCP/IP 详解卷 1:协议》。直到多年以后，我才意识到自己当初是个幸运儿，这两本书帮助我快速建立起数据网络的理论框架，花最少的时间渡过了学习一门新知识最开始那段迷茫、苦闷的阶段。《CCNA 学习指南》虽然名不见经传，但它同《TCP/IP 详解卷 1:协议》一起向读者诠释了一本优秀技术书籍应该具备的素质，即用最浅显的语言将复杂理论背后的逻辑直白地剖析出来，就好象杨振宁给大学生教授基础物理一样，通过生动、形象的语言将所有的现象最后都归结为几条基本的物理规律。只要掌握了这些基础规律，就能直接抓住事物的本质，以后即使碰到没见过的情况，也能快速分析出个大概，达到“授之以渔”的目的。

可惜的是，我发现那样优秀的书籍真是可遇而不可求，大部分技术书籍满足于描述现象，止步于探究逻辑，在中文技术书籍的领域，这样的情况更加严重。作为对未知抱有渴求的年轻人，阅读了大量这样的技术文献后，脑袋里装满了各种规则和指标，对规则背后更深层的力量却仍然一无所知。这样的现象日复一日地在身边重复，直到有一天我突然问自己：“为什么不写一点自己喜欢看的东西呢?”。我开始在国内很有影响的技术网站“弯曲评论”发表了一系列以“拨云见日”为题目的技术文档，主题是剖析当下热门的新一代网络技术。没想到这些文章收到了热烈的反响，“弯曲评论”的站长陈怀临先生也破天荒地表达了赞许。这让我有了更多的干劲，于是便诞生了把我知道的东西都写出来的计划。

而定下“云计算网络”的主题也是一个有趣的过程。云计算这两年快速发展，其中的网络技术是非常重要的基础，而与之对应的中文资料却非常少，即使偶尔有一两篇文章，也难成系统，大家都在很辛苦地盲人摸象。客观地说，目前引领网络技术发展的仍然是聚集在硅谷的一群高科技公司，碰巧我的工作能够接触大量的前沿网络技术，同时又需要频繁地与用户交流他们的实际需求，这样的工作性质使得我可以以一个非常宽广且客观的视角来观察这个行业的技术发展趋势，说得通俗一点就是既有“望远镜”又有“放大镜”。手握这样的条件，我来撰写这个主题就具备了天生的优势，也许主观条件和客观条件同时就绪的人还真的不多，结果我的这本书就误打误撞成了国内第一本系统描述云计算相关网络技术的书籍。

然而在真正开始写作之后，我才发现好书奇货可居的原因。了解一个道理是一回事，能将它讲出来是另一回事，而要把它讲得漂亮并用书面语言形成于纸面上则要高上好几个层次。这需要作者自己首先理解到 120 分，然后才可能说出 80 分的道理。所以写作的过程对于我来说也是一个重新学习的过程，不断地质疑、推倒、重建，在这个学习地过程我再次深深地为那些经典的网络协议设计所折服，我在书中有意识地记录下这个过程，并急切地渴望通过文字把我对这些理论的理解同更多的人分享。

说回本书的内容，数据网络的真正爆发是最近十几年的事情。在这十几年中，网络技术几乎是以百米冲刺的速度在跑一场马拉松，各种理论和协议标准层出不穷，这说明当前的网络架构远未达完美，还有巨大的潜力可以挖掘。如果你在周五晚 7 点来到北京的国贸地区，最惹眼的不是繁华的城市霓虹，而是高架桥上一眼望不到边的蠕动着的汽车尾灯。成千上万的车辆拥堵在城市主干道上，已经成为阻碍城市运转效率的痼疾。数据网络至于方兴未艾的云计算来说，正好像城市交通之于和谐发展，地铁早高峰时段的“人肉罐头”一日不解决，广大上班族的心头就一日不能轻松。我们今天面对的云计算的巨大潜能，如何在网络技术上做好准备，迎接即将到来的浪潮之巅，是这个行业中每一个希望探得先机的从业者需要考虑和面对的问题。

这本书是我第一次持续这么长的写作计划，断断续续地拖了大半年。我必须感谢图灵公司的杨海玲和陈冰两位老师，没有他们的鼓励与帮助，我这个没有任何写作经验的菜鸟不可能完成这个庞大的计划；我要感谢陈漪，她在很多专业方面给了我非常多建设性的意见；我还要谢谢陈怀临老师开办的“弯曲评论”网站，这里是“拨云见日”启航的地方，也是国内少有的高含金量网络技术社区，本书就是试着实践其学习、分享的宗旨的一次尝试；我还要谢谢李聪的支持，没有李聪的支持和鼓励，本书不可能成形；最后，

我还要感谢我在思科的同事，特别是我的三位老板孙伟、张斌和刘洋，他们在技术工作上表现出来的专业与敬业是我在给这本书定下的标准。由于本人的知识能力有限，本书在内容和编写方面难免存在缺陷，欢迎大家指正批评，虽然我的写作难以达到《CCNA 学习指南》和《TCP/IP 详解卷 1:协议》那样的高度，但是我希望这本书能够成为中文技术书籍中的一本诚意之作，为有志于网络技术的朋友提供一点实实在在的帮助。

第一篇 缘起篇

第1章 云计算的兴起

云计算是一个内涵丰富而模糊的名词，说它丰富是因为云计算已经席卷了 IT 行业的方方面面，说它模糊是因为少有人能清晰地把握住它的本质。认识云计算是认识云计算网络的前提，本书将从剖析云计算的含义开始，带着大家认识云和云计算网络。

本章将由探究云计算的身世开始，逐渐揭开云计算身上的迷雾，然后带着大家认识云计算的特征，接着向大家介绍什么是 IaaS/PaaS/SaaS，以及公有云/私有云/混合云/社区云等几种云计算的类型划分，最后说明在企业 IT 环境中，云计算的优势何在。

如果你是一个心急的读者，你大可跳过第一章直接进入后面详细介绍网络技术的部分，但我还是推荐你花一点时间来熟悉云计算的基本概念，这不但将带给你更多的阅读乐趣，而且对更好地把握云计算网络的本质也大有帮助。

1.1 云计算的身世

今时今日，身为一个 IT 从业人员如果不能忽悠两句云计算，那简直是不可想象的！云计算的影响力已经席卷了各个角落，不管有没有关系的厂家都把自己的产品往云计算的大旗靠，以赶上世纪初的这股潮流。这么看起来，云计算可以算是个人电脑、互联网后的又一个革命性技术，将在未来数年内统领 IT 行业的发展方向，并改变我们使用信息技术的方式。

然而，你也许想象不到，今天大红大紫的云计算概念，实际上在上世纪中叶已经诞生了，经过几十年的发展，云计算已经从一个科学家脑海里对未来世界的畅想，落地为实实在在的产品影响着我们每天的生活。趁着这个机会，让我们先走进历史，回首云计算从孕育到成熟的过程。

1.1.170 年前的畅想

上世纪 60 年代是一个百废待兴的时代，从东方到西方，所有人都提着一股“大干快上”的气势，民用领域的科学技术开始快速发展，计算机刚刚开始用上半导体器件，准备进入后来被称为“摩尔定律”的通道，一切的一切在今天的人们看来都充满着一种梦幻、复古的气息。

有时候你不得不承认人类的想象力是推动这个世界前进的内在动因，就在这个绝大多数人还没用过计算器的时代，来自斯坦福大学的科学家 John McCarthy 就表示“计算机可能变成一种公共资源。”这句话在当时听上去的效果，就好像杨利伟升空后马上有人告诉你火星房地产市场将会急剧升温一样，有着极度超前的逻辑。

John McCarthy 并不是唯一一个做出如此预测的学者，1966 年 Douglas Parkhill 在他的著作《The Challenge of the Computer Utility》对这个理论有个更加具体的描述，他将计算资源类比为电力公司，并提出了私有资源、公有资源、社区资源等概念，类似动态扩展、在线发布等今天被频繁提起的云计算特性，在 Douglas Parkhill 的这本书中已经有了非常详尽的描述。

Douglas Parkhill 的理论还有一个更加具体的诠释者，著名的 ARPAN（Advanced Research Projects Agency Network/高级研究计划署网络，这是世界上第一个数据包交换网络，是因特网的前身）负责人 J.C.R. Licklider 将计算资源与他熟悉的数据网络做了结合，提出了“从任意地点通过网络访问计算机程序”的设

想。

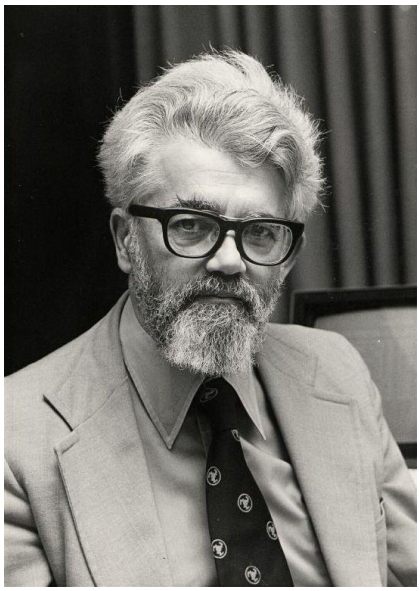


图 1-1 John McCarthy 教授在上世纪 60 年代就提出了云计算的概念

所有这些伟大的思想在那个泛黄的年代里碰撞、激荡，启迪了无数人对计算机朦胧的畅想。虽然提出这些基本概念的先辈科学工作者们并没有看到自己的思想落地生根，他们甚至不知道“云计算”这个特定的词汇，但是他们提出的这些理念非常明确地指明了计算机发展的方向，明确地说明了计算与网络相辅相成的关系。

从上世纪 60 年代的故事中，我们可以看到云计算不是一个偶然的技术产物，它是计算机技术演进的必然方向，是从计算机诞生第一天起就注定发生的技术革命。而数据网络作为分发服务的渠道，与计算资源的整合也是这个过程中必不可少的一环。

1.1.2世纪之交的互联网盛宴

如果你是一名 IT 老兵，那你对世纪之交的那场互联网泡沫一定记忆犹新。上世纪 90 年代后期，世界经济形势一片大好，在美联储低利率刺激下，大量资金涌入以互联网为代表的高新技术行业，一大批互联网企业上演着“车库创业—获得风险投资—业务快速扩张—纳斯达克上市—30 岁退休”的神话。

然而这个失去理智的局面只持续了很短一段时间，随着 1999 年到 2000 年上半年美联储连续六次加息，资本市场给出了剧烈的反应，无数指着靠烧钱撑到上市的互联网公司一夜间失去了资金来源，即使已经上市的公司，股票也一泻千里。正如今天一句时髦的话所说“潮水退去才发现大部分人在裸泳”。

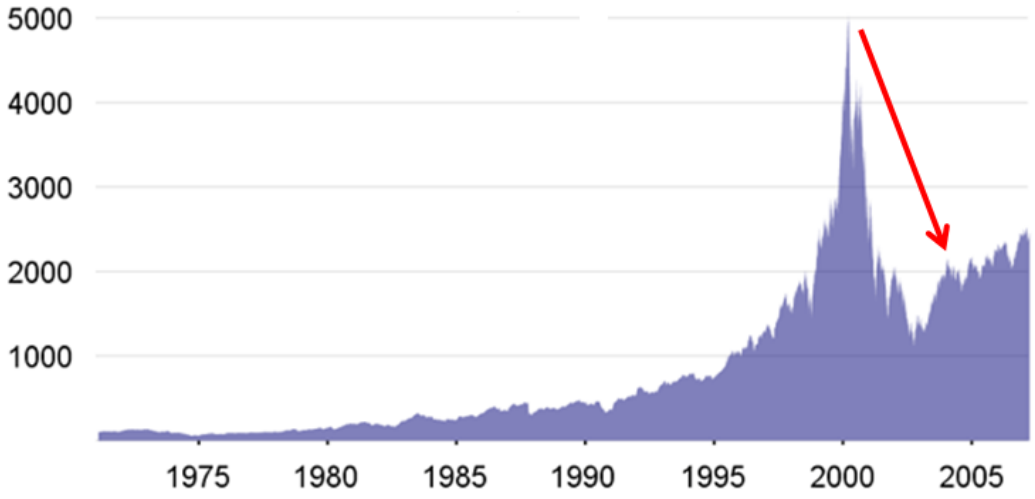


图 1-2 2000 年互联网泡沫破灭，纳斯达克科技股跳水

然而，这场疯狂的盛宴带给我们的并不只有疯狂的记忆，不少有价值的想法趁着宽松的融资环境得到了一试身手的机会，现代的云计算模式便诞生于 90 年代末的这次互联网大潮。

1997 年，Ramnath Chellappa 教授在他的一次演讲中第一次提出了“云计算/Cloud Computing”这个词，他指出“计算资源的边界不再由技术决定，而是由经济需求来决定”。换句话说，计算资源的形式是可以动态的，这种形式根据人们的需求而变化，如果你不在机房内但却需要操作软件程序，那么自然会有一种方式让你能够远程登录计算机。Ramnath Chellappa 教授的理论重复了上世纪 60 年代那些科学家的思想，但同上世纪 60 年代不同，技术的发展使得 Ramnath Chellappa 教授的理论已经可以找到现实中的模型。

在互联网泡沫盛极而衰的 1999 年，一家对后世产生深远影响的公司成立了，它的名字是 Salesforce.com。Salesforce.com 是现在公认的云计算先驱，它由几个前 Oracle 高管成立，主要向企业客户销售基于云的 SaaS（软件即服务，云计算的一种服务类型，本章第五节将详细介绍）产品。Salesforce.com 自成立的第一天起就表现出鲜明的特色，例如这个“.com”并不是好事的媒体加上去的，而是正式名称的一部分，Salesforce.com 通过这个方式告诉世人，它是一家带着网络基因诞生的公司。

Salesforce.com 对云计算产业的重要意义不仅在于它通过销售 SaaS 服务，成长为一家年销售额超过 20 亿美元的公司，更重要的是 Salesforce.com 第一次将 SaaS 服务大规模地销售给了企业用户。Salesforce.com 的主要产品是 CRM（Customer Relationship Management），即客户关系管理系统。这个领域一向是 SAP、Oracle 这种玩家统治的地盘，企业客户利用 CRM 来处理他们最宝贵的客户资源和市场数据，并指导下一个季度的销售策略，对于企业的 IT 部门来说，保证 CRM 的稳定和高效是优先级最高的工作。普通用户使用的在线服务标准同企业用户的标准完全不在一个层面上，我们在家用 Gmail 发个邮件这种行为，在 CRM 的管理人员看起来根本没有任何说服力。你能想象一家年收入几十亿美元的公司将它的 CRM 系统从坐落在自己精密机房内的一排嗡嗡作响的 IBM 主机挪到一个小小的浏览器内吗？况且这个浏览器访问的数据中心还完全不在这家公司的掌控之内！但是 Salesforce.com 做到了，它不但做到了，而且做得相当不错，今天它的客户名单内包括一大串通用电气、荷兰航空、富国银行、Comcast、NBC、电子艺界这样的名字，这些名字一点都不像是买不起昂贵的服务器而不得不使用廉价在线方案的小公司。

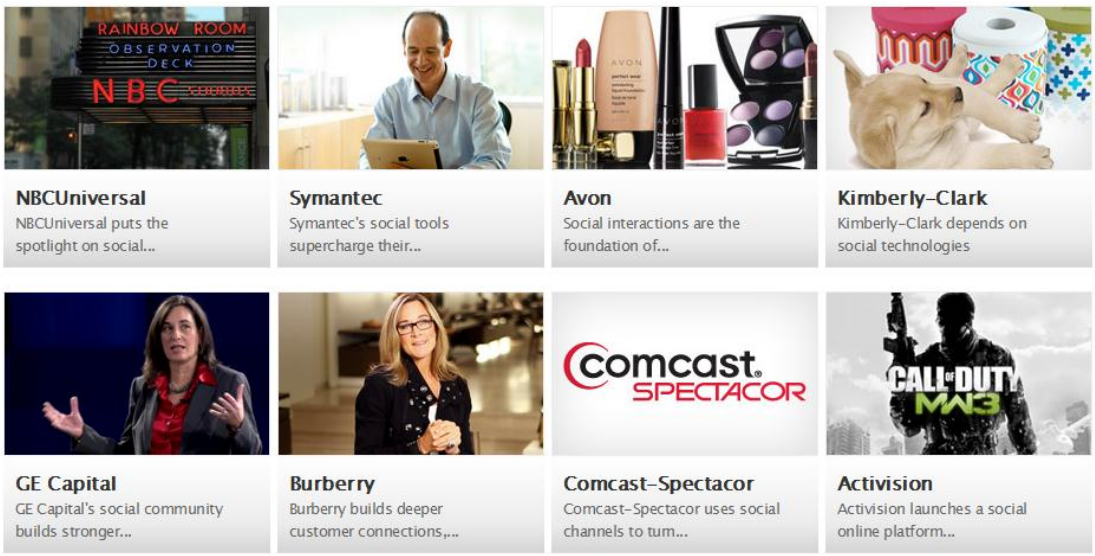


图 1-3 Salesforce.com 的客户包含大量大型企业

Salesforce.com 的成功之处在于它头一次证明了基于云的服务不仅仅是大型业务系统的的廉价替代品，它还可以是真正提高企业运营效率、促进业务发展的解决方案，同时在可靠性方面维持一个极高的标准。至此之后，最苛刻的企业用户开始全面拥抱云计算，我们终于迎来了云计算掀起的第一个高潮。

1.1.3世纪初的爆发

时间进入 21 世纪的第一个十年，接棒 Salesforce.com 将云计算推向下一个高峰是在线零售商 Amazon。Amazon 是一家非常重视客户体验的公司，当发展到一定规模的时候，它发现自己的数据中心在大部分时间只有不到 10%的利用率，剩下 90%的资源都闲置着，这些资源仅有的作用是被用来缓冲圣诞购物季这种高峰时段的流量。于是 Amazon 开始寻找一种更有效的方式来利用自己庞大的数据中心，其目的是将计算资源从单一的、特定的业务中解放出来，在空闲时提供给其他有需要的用户。这个计划首先在内部实施，得到的回馈相当不错，Amazon 接着便将这个服务开放给外部用户，并命名为 AWS（Amazon WEB Service/亚马逊网络服务）。

初期的 AWS 只是一个简单的线上资源库，虽然它依托 Amazon 的品牌光环吸引了不少注意力，但在大多数时候，人们只是将 AWS 当作一个互联网公司吸引眼球的玩票行为。这种情况持续了 3 年左右，直到 Amazon 在 2006 年发布了令 AWS 名声大噪的 EC2。

EC2 是 Elastic Compute cloud 的缩写，是一款面向公众提供基础架构云服务的产品。简单来说，EC2 在云端模拟了一个计算机运行的基本环境，如果你接触过虚拟机技术，那么 EC2 可以看作一个架构在云端的虚拟机。打个比方，你需要为一个为期三个月的项目搭建一台 Windows server 2008 服务器，有了 EC2 之后你不用采购服务器、配置硬件，只需要向 Amazon 申请一个为期 3 个月的帐号，然后将你的 Window server 2008 应用上传到 Amazon 的服务器上就完成部署了。Amazon 会提供一个公共网关，通过这个网关你可以访问架设在 Amazon 数据中心内的这个 Windows server 2008 服务器的所有功能。

图 1-4 Amazon AWS 是标志性的云计算服务

之所以说 EC2 是一个里程碑式的产品，是因为 EC2 是业界第一个将基础架构大规模开放给公众用户的云计算服务。EC2 在 Salesforce.com 之后将云计算的服务对象带入了更广泛的领域，云计算的用户不再拘泥于某种特定的服务类型上，他们可以在 EC2 的平台上搭建从 Linux 到 Windows 的任何业务。更棒的是这些业务的体量可大可小，能够随着用户的需求而增减。根据 Amazon 网站上的公开报价显示，如果你只是使用一个最简单的 Linux，其价格为每小时 8 美分！想一想，只需要你掏不到一块钱，就能体验跟世界第一在线零售商店同等级的数据中心服务，这难道不是一件非常神奇、非常酷的事情吗？

除了 EC2 之外，Amazon 还发布了 S3、SQS 等其他云计算服务，组成了一个完整的 AWS 产品线。如果说 Salesforce.com 是给人们做了云计算的启蒙，那么 Amazon 则用 AWS 引爆了云计算这座火山，云计算从此正式成为 IT 产业的主声部。

1.1.4百花齐放的年代

Amazon AWS 之后，各种类似的云计算产品开始层出不穷地冒出来，云计算不再只是一个阳春白雪的概念，而是所有人追逐的方向。短短几年内，Amazon 就不再是市场上唯一的 IaaS（基础架构即服务，云计算的一种服务类型，本章第五节将详细介绍）提供商，Microsoft 等巨头纷纷涌进这个领域。

图 1-5 市场上的主流云服务提供商

除了数量的增长，云计算类型也变得日益丰富，Salesforce.com 和 Amazon AWS 分别代表了 SaaS 和 IaaS

（基础架构即服务，云计算的一种服务类型，本章第五节将详细介绍）两种云计算服务，除此之外的第三种服务 PaaS（平台即服务）也快速发展起来。2009 年，Google 开始对外提供 Google App Engine 服务，Google App Engine 是一个 PaaS 服务，他搭建了一个完整的 WEB 开发环境，用户可以在浏览器里面开发和调试自己的代码，然后直接部署到 Google 的云平台上，并对外发布服务。以 Google App Engine 为代表的 PaaS 服务补齐了云计算的产品版图，从此用户可以在基于云的环境中找到绝大部分计算资源。理论上来说，如果你对服务器硬件已经感到厌倦，那么你已经可以将它们淘汰掉，将你的应用搬迁到这些互联网上的云平台中。

云计算蓬勃发展的另一个特征是，围绕在线资源的应用开始快速出现。2009 年，第一个基于 Amazon AWS API（Application Program Interface/应用程序接口）的私有云平台 Eucalyptus 出现，通过 Eucalyptus 你可以利用 Amazon AWS 的计算和存储资源打造自己的私有云架构。类似这样的技术进一步完善了云计算平台的安全性和可靠性，并打通了它和企业用户原有应用平台之间的通道。同年中，信息研究机构 Gartner 预测企业用户将从基于设备的 IT 建设模型往基于单个用户需求的云计算模型转变。

进入 21 世纪的第二个十年，云计算已经进入了百花齐放的时代。人们现在已经不再谈论云计算是否可行，这早已不是一个问题，现在的问题是云计算未来的发展方向，怎样将云计算的潜力充分发挥出来，以帮助我们塑造一个更加高效的 IT 系统和互联网世界。这个问题的答案涉及到非常广泛的内容，因为云计算是对现有 IT 模式颠覆性的革新，原有的建设思路必然会受到冲击，其中数据网络就是一个重要的领域。本书的目的正是描述在新时代的云计算环境中，数据网络技术的进化，以及这种进化反过来给云计算带来的影响。

1.2 云计算的 DNA

回顾完云计算的历史，让我们把注意力集中到一个最基本的问题上：什么是云计算？这是一个被反复提到、反复回答之后，又反复提出的问题，这本身就是一个非常有趣的现象，因为当无线局域网开始流行后，你不会一直听到有人问“什么是无线局域网？”；802.1X（一种二层认证服务，本书第三章将做详细介绍）安全认证广泛部署后，你也不会一直听到有人问“什么是 802.1X？”，这说明云计算本身是一个非常抽象的概念，要准确地把握其内涵不是一件容易的事，另一方面，这又表示云计算是有着强大生命力的技术发展方向，否则不会有越来越多的人翻来覆去地尝试弄懂这个问题。就让我们从辨明云计算的真实身份开始，解开云计算网络的旅程吧！

不同人从不同角度解释了云计算的含义，这些解释五花八门，各有各的道理，我觉得要理解一件事物的本质就是要抓住其独有的特征，就好像你描述鲸鱼的时候一定会提到“这是地球上最大的哺乳动物，生活在海洋里”，这样的论述一针见血地说明了鲸鱼的特征，就好像将它的 DNA 展现出来一样精确。

云计算作为一种独特的 IT 服务模式，也有其独有的 DNA，纵观各方观点，我认为 CSA（Cloud Security Alliance/云计算安全联盟）在《Security Guidance For Critical Areas Of Focus In Cloud Computing V3.0》中比较精确地说明了云计算的本质：

> ***“云计算的本质是一种服务提供模型，通过这种模型可以随时、随地、按需地通过网络访问共享资源池的资源，这个资源池的内容包括计算资源、网络资源、存储资源等，这些资源能够被动态地分配和调整，在不同用户之间灵活地划分。凡是符合这些特征的 IT 服务都可以称为云计算服务。”***
>
> ——《Security Guidance For Critical Areas Of Focus In Cloud Computing V3.0》

上面的定义很好地说明了云计算的本质，为了将这个定义更方便地匹配到现实世界的 IT 架构中，

NIST（U.S. National Institute of Standards and Technology/美国国家标准与技术学院）提出了一个定义云计算的标准《NIST Working Definition of Cloud Computing/NIST 800-145》，这个文档中提到一个标准的云计算需要具备五个基本元素，分别是通过网络分发服务、自助服务、可衡量的服务、资源的灵活调度，以及资源池化；这个标准还提到云计算按照服务类型可以分为 IaaS、SaaS 和 PaaS 三类，而按照部署模式可分为公有云、私有云、混合云和社区云四种。

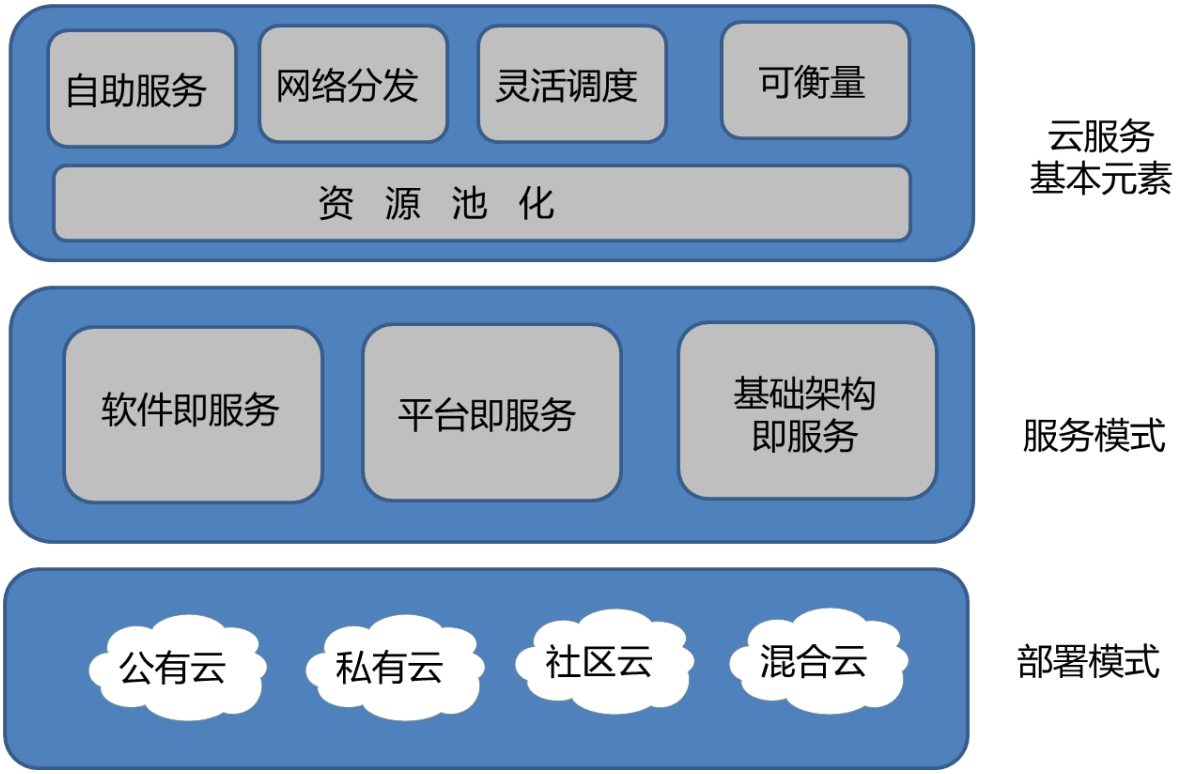


图 1-6 NIST 提出的云计算架构图

NIST 800-145 被业界普遍接受的原因是其提出的云计算五大要素非常简练地说明了一个云计算系统的特征，只有同时具备这五点的 IT 架构才可以被称为云服务架构，通过这五个特征能够快速地将云计算系统同传统 IT 系统区分开来，抓住这五点就好像在亲子鉴定中准确地抓住了特征 DNA，能够快速地把云计算从庞杂的 IT 服务类型中提取出来。

1.3 云计算的基本元素

我们先来看一看这五大要素的具体内容，后文再对部署模式和服务模式做介绍。

1.3.1自助服务

云计算与传统架构的区别，从用户接触云计算的第一步就体现出来了。用户大多通过自助方式获取服务。例如在 WebEx 公司推出的在线会议系统中，用户自助挑选会议类型、设定参会人数、上传会议资料，然后点击“确定”即可。WebEx 的后台服务器便会在指定时间将与会人员连接到一个虚拟在线会议室中。云服务提供商并不需要人工干预这个流程，所有的细节都是用户自己在网页上选择决定的。

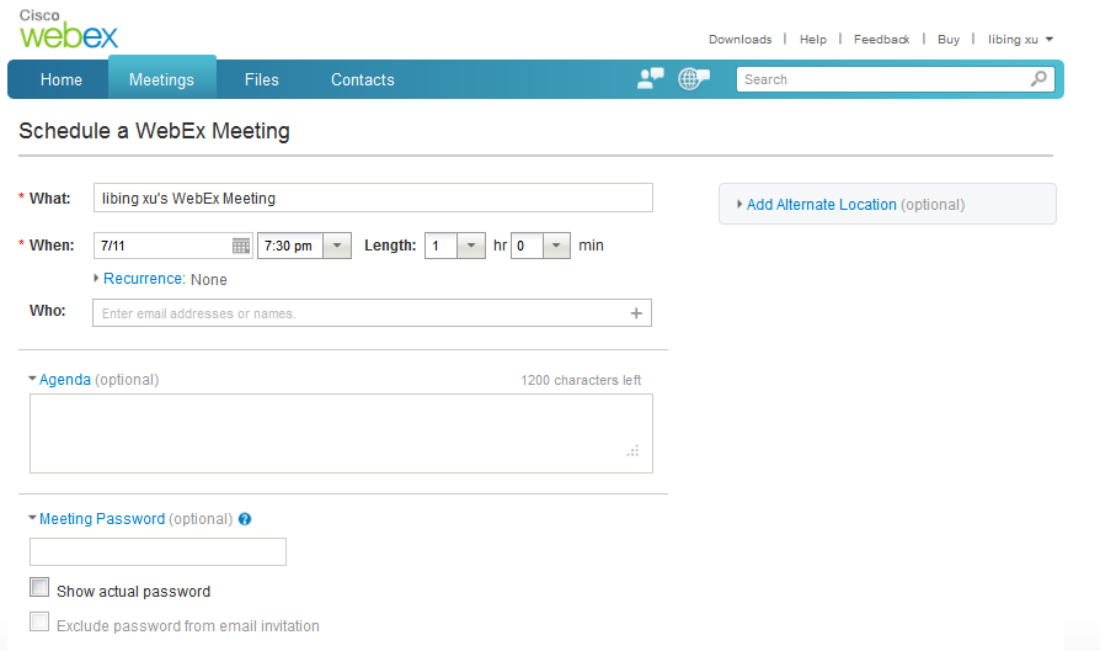


图 1-7 用户可以通过 WebEx 的页面设置丰富的会议选项

自助式服务也是区分简单的 B/S 架构与真正云计算的一个重要标准，让我们来看一个反面例子，在 WebEx 出现之前，市场上很早就有很多提供多方语音会议服务的公司，用户只需要购买点卡就可以通过拨打特殊号码进行多方会议甚至多媒体视频会议，这同 WebEx 的概念非常相似，但在这类会议服务中，如果用户需要从一个普通的多点会议变更成一个能远程控制对方电脑桌面的多人培训时，往往要致电服务提供商，由服务提供商的工作人员人工改变服务类型，甚至重开一个新的服务接入号。而在 WebEx 系统中，所有这一切用户只需要在网页中简单地点几下鼠标就完成了。以往的会议服务仅仅是将提供多方会议服务的设备从分散的用户机房集中到统一的中心机房，其软硬件设施仍然是僵化的传统架构，无法自动跟上用户需求的快速变更，仍然需要人工干预来完成资源的重新划分和调整，称不上是真正的云计算。

自助式的服务方式充分发挥了云计算后台架构强大的运算能力，同时，用户也获得了更加快捷、高效的体验。

1.3.2通过网络分发服务

在云计算出现之前，当人们需要使用计算机完成工作时——例如写一本书——都会购买一台计算机。这句话听上去比较可笑，但这就是真实情况，因为我们必须实际接触到这台电脑，才能享受到它提供的服务。

云计算的第二个重要特征就是要打破这种一一对应的关系，因为大量的云计算服务都是通过网络来传递的。

还是以写书为例，当云计算真正普及后，理论上我不需要拥有一台配置完备的 PC 机，我只需要在 IPAD 上登录 Google Docs，就能够进行在线写作。Google Docs 提供了绝大部分的电子文档编辑功能，其在线存储功能的可靠性甚至比我自己买一块移动硬盘保存我的文稿效果还要好。通过 Google Docs 写作，我不需要新添置计算机，不需要购买 Office 软件，我需要的只是一条网线，然后就可以通过任何带 WEB 浏览器的工具进行工作了。

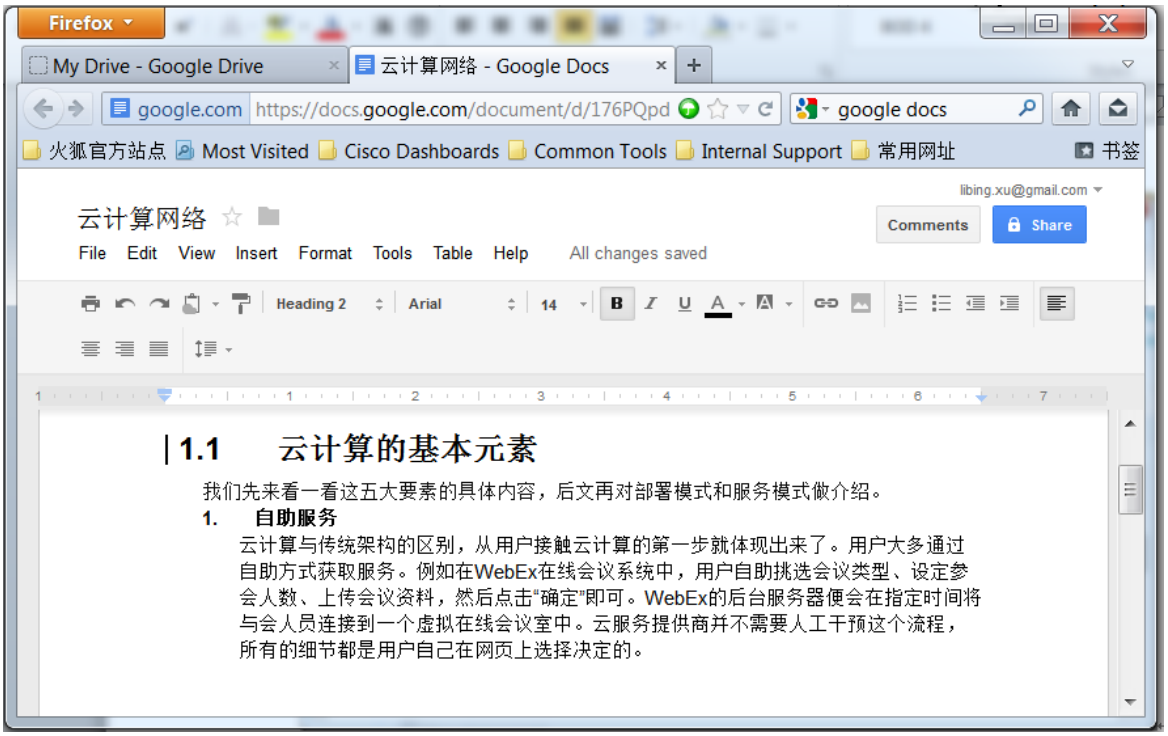


图 1-8 Google Docs 的界面同 Office 软件非常类似

通过网络分发服务打破了地理位置的限制、打破了硬件部署环境的限制，只要有网络就有计算，革命性地改变了我们使用电脑的习惯。

1.3.3资源池化

资源池化是个很时髦的名词，它也是由云计算引申出来的概念。在云计算中，计算资源——CPU、存储、网络等等——有了新的组织结构，也就是资源池，所有设备的运算能力都被放到一个池内，再进行统一分配。

如果你觉得这种说法过于抽象，想象一下下面的场景，一个企业内部的开发人员向 IT 部门申请了一台服务器，这台服务配有一颗 Intel X5600 2.4G CPU、32GB 内存、500GB 硬盘、1Gbps 上联链路，虽然 IT 满足了他的需求，但这台服务器并不真实存在，它也许只是从一台配置为 4 路 CPU、2TB 内存、采用集中式存储、配置双路 10Gbps 网卡的服务器中切割出来的一部分而已。

对于 IT 部门来说，计算资源不再以单台服务器为单位，云计算打破了服务器机箱的限制，将所有的 CPU 和内存等资源解放出来，汇集到一起，形成一个个 CPU 池、内存池、网络池，当用户产生需求时，便从这个池中配置能够满足需求的组合。在传统的 IT 架构中，这几乎是天方夜谭，上面那个例子中，如果用户请求的服务器配置在机房内正好找不到空闲的设备，那么只有两种选择，要么 IT 新采购一台设备，要么用户修改需求，显然不管是哪一种都会降低效率或增加成本。

资源的池化使得用户不再关心计算资源的物理位置和存在形式，IT 部门也得以更加灵活地对资源进行配置。

1.3.4资源的灵活调度

资源的灵活调度是资源池化的下一步发展。目前广为流行的云计算服务，例如 WebEx、Amazon EC2 等，都有一个明显的特征，就是云计算服务商总是可以满足用户不断增长的需求。在 WebEx 平台上，已经有召开过同时上千人参与的全球视频会议，而 WebEx 表示这个人数仍然没有达到上限。

这就是云计算在资源调度方面的灵活性，由于计算资源已经被池化，因此云计算服务提供商可以非常快速地将新设备添加到这个资源池中，满足不断增长的需求。对于用户来说，这种感觉就好像只要愿意付账单，就可以要求无限制的资源。

另一个著名的例子发生在 OMGPOP 游戏公司，这家公司开发了迅速走红的手机游戏“Draw Something/你画我猜”，在 6 个星期内，在线用户数从 0 暴涨到 2500 万，然后被 Zynga 以 2 亿美元闪电收购。通过云计算平台，OMGPOP 在这 6 个星期内高效地扩展了后台系统的容量，从容地应对了 2500 用户产生的数据洪流，可以说，没有云计算技术，OMGPOP 不可能保证高水平的用户体验，也就不会有后来 2 亿美元的商业奇迹。

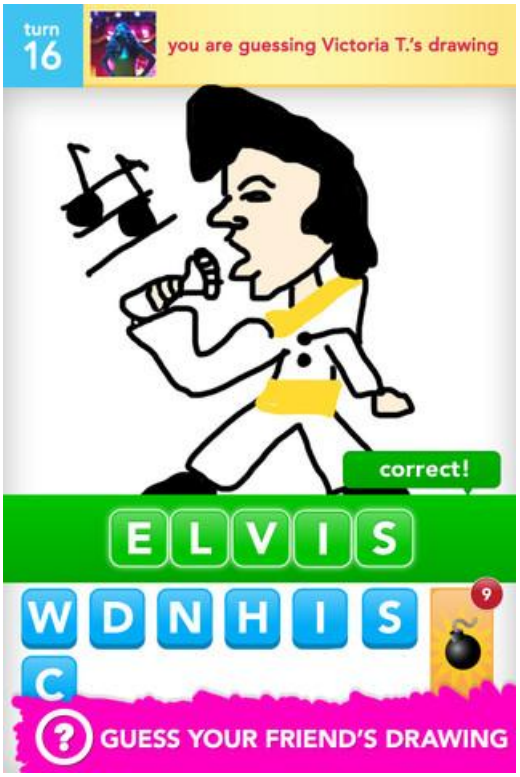


图 1-9 OMGPOP 的 Draw Something 游戏在 6 个星期内取得 2500 万用户

1.3.5可衡量的服务

云计算的最后一个特点是对计算资源的衡量。一个完整的云计算平台会对存储、CPU、带宽等资源保持实时跟踪，并将这些信息以可量化的指标反映出来。基于这些指标，云计算平台运营商或管理企业内部私有云的 IT 部门，能够快速地对后台资源进行调整和优化。

NIST 提出的以上这五点非常形象地提炼出不同云计算模式的共性，是蕴藏在云计算系统内的 DNA。在绝大部分获得成功的云服务身上，都能轻易找到这五点特征。Amazon AWS 的 EC2 就是一个典型的例子，Amazon EC2 的服务全部可以在 Amazon AWS 的网站上自助开通，用户通过网络获取 Amazon EC2 的后台计算资源。Amazon EC2 支撑了很多从零起步成长起来的大型互联网公司，例如位置签到服务公司 Foursquare 等。Amazon EC2 有一个完善的后台管理系统，能够在不同的数据中心之间调配资源，满足瞬息万变的用户需求，所有这些，将 Amazon EC2 塑造成一个优秀的成功的云服务提供商，也帮助我们界定出优秀云计算服务的基本模型。

1.4 IaaS/PaaS/SaaS，它们都是什么？

前文谈到了云计算的分类方式，在谈论云计算的场合，PaaS、SaaS 和 IaaS 是常常被提起的热门字眼，他们代表了最基础的云计算服务模式，了解这些模式的特点和区别是进行所有基础架构——包括网络——设计的前提。

虽然云计算的服务模式仍在不断进化，但业界普遍接受将云计算按照服务的提供方式划分为三个大类：SaaS（Software as a Service/软件即服务）、PaaS（Platform as a Service/平台即服务）和 IaaS（Infrastructure as a Service/基础架构即服务），PaaS 基于 IaaS 实现，SaaS 的服务层次又在 PaaS 之上，三者分别面对不同的需求。IaaS 基础架构即服务提供的是用户直接访问底层计算资源、存储资源和网络资源的能力；PaaS 平台即服务提供的是软件业务运行的环境；SaaS 软件即服务是将软件以服务的形式通过网络传递到客户端。下面分别看看三者的详细定义。

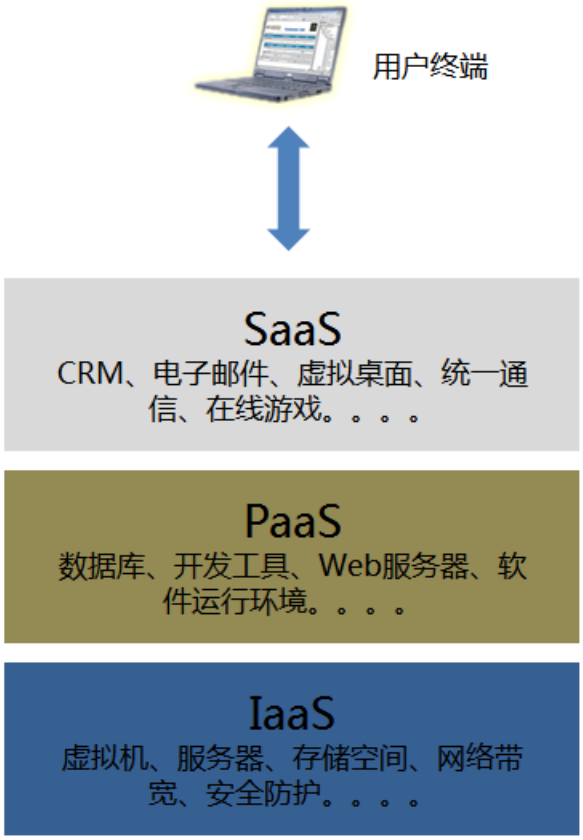


图 1-10 三种云计算模式的关系

1.4.1基础架构即服务/IaaS

IaaS 通过虚拟化技术将服务器等计算平台同存储和网络资源打包，通过 API 接口的形式提供给用户。用户不用再租用机房，不用自己维护服务器和交换机，只需要购买 IaaS 服务就能够获得这些资源。Amazon 是目前最大的 IaaS 服务提供商之一，Amazon EC2 不但包括基本的基础架构环境，还通过自身强大的数据中心为用户提供一定级别的高可用性保障，用户购买 Amazon 的服务后无需再操心数据中心冗余、业务数据备份等工作。

1.4.2平台即服务/PaaS

PaaS 构建在 IaaS 之上，在基础架构之外还提供了业务软件的运行环境，个人网站常常用到的“虚拟主机”实际就属于 PaaS 的范畴，个人站长只需要将网站源代码上传到“虚拟主机”的地址，“虚拟主机”会自动运行这些代码生成对应的 WEB 页面。除了形成软件本身运行的环境，PaaS 通常还具备相应的存储接口，这些资源可以直接通过 FTP 等方式调用，用户无需从头进行裸盘的初始化工作。

PaaS 面向的用户是没有能力或者不愿意维护一个完整运行环境的开发人员和企事业单位，通过使用 PaaS 服务，他们可以从繁琐的环境搭建中抽身出来，将更多的精力投入到业务软件的开发中。同 IaaS 相比，PaaS 提供的服务类型更加丰富，因为软件的生态系统本身就呈现百花齐放的状态，且 PaaS 可以基于 IaaS 服务搭建，初期的硬件投入较小，所以 PaaS 服务提供商的数量和种类也较 IaaS 更多，互联网上比较著名的 PaaS 包括 Google App Engine 和 Microsoft Azure 等。

1.4.3软件即服务/SaaS

SaaS 是最成熟、知名度最高的云计算服务类型，在云计算真正变得火热之前，软件即服务本身就已经是一个非常流行的概念了。SaaS 的目标是将一切业务运行的后台环境放入云端，通过一个瘦客户端——通常是 WEB 浏览器——向最终用户直接提供服务。最终用户按需向云端请求服务，而本地无需维护任何基础架构或软件运行环境。

SaaS 同 PaaS 的区别在于，使用 SaaS 的不是软件的开发人员，而是软件的最终用户，一个 PaaS 用户如果通过 WEB 方式向外分发其服务，则其提供的也就是 SaaS 服务。SaaS 诞生多年来，不但发展出成熟的技术模型，而且已经是一个经过验证的成功商业模式，Salesforce.com 通过向大企业销售云端 CRM 服务已经成长为一家年收入超过 20 亿美元的纳斯达克上市公司。

1.5 私有云/社区云/公有云/混合云

根据 NIST 的定义，云计算按照部署模式可以分为私有云、公有云、混合云与社区云四种，不同的部署模式对基础架构提出了不同的要求，在正式进入云计算的网络设计之前，我们有必要弄清几种云计算部署模式之间的区别。私有云是部署在企业内部，服务于内部用户的云计算类型；社区云是由数个有共同利益关系或目标的企业和组织共同构建的云计算业务，其服务面向这几个组织的内部人员；公有云一般是由云服务运营商搭建，面向公众的云计算类型；混合云则是包含了两种以上类型的云计算形式。

1.5.1私有云

私有云的建设、运营和使用都在某个组织或企业内部完成，其服务的对象被限制在这个组织内部，对外没有公开接口。一个企业的软件开发部门是最有可能部署私有云的场景之一，开发部门需要频繁地在不同的硬件环境中进行软件测试，参考我们之前在“资源池化”中举的例子，通过在测试服务器上部署云平台，开发部门能够在最短的时间内变更服务器配置，从而加快开发流程，而不增加硬件投资。

私有云不对组织外部的用户提供服务，但是私有云的设计、部署与维护可以交由组织外部的第三方完成，也就是说，企业或事业单位在建设自己的私有云系统时，可以充分利用外部咨询公司和系统集成商的能力，不必事必躬亲，将所有工作大包大揽给自己的 IT 部门。

1.5.2公有云

与私有云对立的概念就是公有云，公有云也是我们日常接触最多的云服务种类。从 Microsoft Office 365 到 Amazon EC2 到 Dropbox，这些著名的云服务都是面向公众的公有云。

公有云通常由一个云服务运营商维护，通过互联网向公众提供服务，任何人都可以申请、使用公有云资源。公有云的受众规模不局限于一地，一旦发布出来，全世界的人都可以通过网络申请使用，因此，公有云的规模一般很大，对可靠性、安全性的要求也更高，其基础架构的组成往往也更加复杂。

1.5.3社区云

社区云是面向一群有共同目标、利益的用户群体提供服务的云计算类型。社区云的用户可能来自不同的组织或企业，因为共同的需求走到一起，社区云向这些用户提供特定的服务，满足他们的共同需求。

由大学等教育机构维护的教育云就是一个标准的社区云业务，大学和其他高等机构将自身的教育资源放到云平台上，向校内外的用户提供服务。在这个模型中，用户除了在校学生，还可能有在职进修学生、其他科研机构的研究人员等，这些用户来自不同的机构，但因为共同的课程作业或研究课题走到一起。

社区云虽然也面向公众提供服务，但与公有云的区别在于，社区云的目的性更加强，社区云的发起者往往是有着共同目的和利益的机构，而公共云只是面向公众提供特定类型的服务，这个服务可以被用于不同的目的，一般没有限制。所以，社区云的规模通常也比公有云小。

1.5.4混合云

混合云顾名思义就是两种或两种以上云计算的综合，混合云可以是公有云与私有云的混合，也可以是私有云与社区云的混合。混合云服务的对象非常广泛，包括特定组织内部的成员，以及互联网上的开放受众。混合云架构中有一个统一的接口或管理平面，不同的云计算模式通过这个结构以一致的方式向最终用户提供服务。

同单独的公有云、私有云或社区云相比，混合云具备更大的灵活性和可扩展性，企业在部署云计算时常常面临瞬息万变的需求，混合云在应对需求的快速变化时有无可比拟的优势。

为了更好的理解混合云的概念，我们来看一个具体的例子。VMware 为了完善自身的混合云解决方案，2011 年在其云计算管理平台 vCloud 中添加了 vConnector 组件，通过这个组件，vSphere 的管理员可以在一个 vCloud 界面内管理企业内部私有云的虚拟机和企业外部的公有云资源，这些公有云资源包括由 Verizon、BlueLock 和 Colt 等公司提供的基于 VMware vSphere 架构的云计算服务。管理人员在统一的 vCloud 界面中可以直接在公有云和私有云之间调动虚拟机、虚拟程序和软件模板等资源。

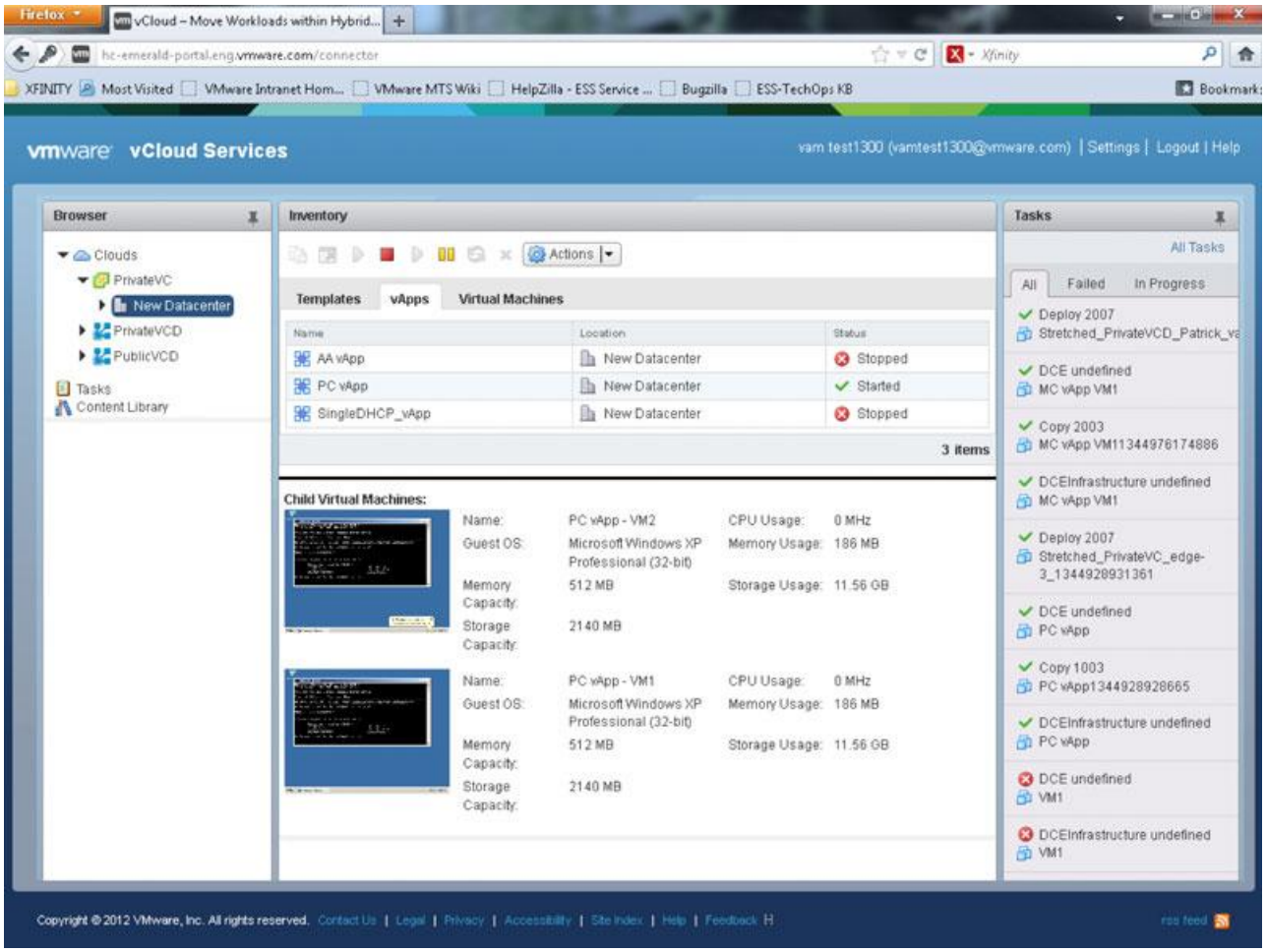


图 1-11 vConnect 可以在单一界面内显示公有云和私有云资源

混合云的出现极大地拓展了云计算的优势，企业用户在私有云能力不足时，能够快速借用外部公有云的资源，从而保证业务不因资源不足而断线。

1.6 云计算的独有优势

相较于传统的 IT 架构，云计算具备鲜明的特点和独特的优势，正是这些优势使得云计算的概念刺激着大众的眼球，让云计算即使还在襁褓阶段就声名远播。所以我决定本书第一章好好总结一下云计算的优点，这样才使得你更有动力来读这本书，也让这本书接下来章节的论述对现实世界更有指导意义。

1.6.1降低成本

这几乎是所有人对云计算的第一印象，云计算自打出生起就被贴上了高性价比解决方案的标签。更准确地说，云计算不止是一个廉价的选择，还能够让用户更灵活地支配他们的预算。

以前用户搭建一个 IT 系统的时候需要采购全套的服务器、网络、存储设备，现在有了云计算技术，他可以把所有的软件都放到云端运行，甩掉维护一大堆硬件的麻烦。更妙的是，系统刚上线用户较少时，他可以只租用一个较小的网络出口，随着业务不断成熟，用户可以线性地增加网络带宽，而不会出现购买了一个昂贵的核心交换机，而业务流量迟迟保持在低水平，投入的资金被白白浪费的情况。

让 IT 资源随着业务需求的增长而增长、减少而减少，这才是云计算优化成本的精髓，用户现在付出

的每一分钱都使在了刀刃上，在一个成熟的云计算环境中，不会出现闲置的计算资源，所有的资源都被分配到需要的地方，不多不少。

1.6.2扩展性

扩展性是另一个云计算常常被人提起的特色，夸张一点说，一旦你将业务放入云计算环境，你就无需再担心带宽不足或存储空间不够这样的问题。

大部分 IT 工作人员都对以下这种升级流程耳熟能详，隔个两三年进行一次业务评估，找出老化或不足的设备，然后申请预算购买新的软硬件，待申请到预算做完升级后，接下来三年内机房内的设备就不会再有大的变化，万一出现资源不足的情况要重新申请预算采购新设备那真比登天还难，更糟糕的是当初负责设计的人员可能还会被扣上“计划不足、水平不够”的帽子。但是说实话，很多时候连老板都不知道公司两年后是什么情况，区区一个 IT 运维小兵怎么说得清这两年里需要用到多少块硬盘呢？所以比较常见的情况是，IT 部门在做预算时往往按照容量的上限申请，这个上限一般都远远高于实际需求，所以机房内会出现大量闲置的设备。

这显然不是一个合理的状态，但却是常态。其原因是传统的 IT 架构扩展起来太困难，扩张一次就伤筋动骨，不但要采购大量设备，还涉及到与原有系统的合并，弄不好就要频繁中断业务，你随便抓一个身边做 IT 运维的朋友，问问他喜不喜欢做系统割接，他一定会瞪着你丢过来一句话“有病吧！”。

图 1-12 熬夜割接是每一个 IT 运维人员的噩梦

在这一点上云计算的优势就非常明显了，通过自助式的服务，用户可以随时随地地添加、减少计算资源，最重要的是这种增减底层资源的做法对上层业务的影响被限制在最小的范围。IT 人员无需关心新设备与老系统的互联，无需关心数据往新存储盘柜的迁移，无需关心原来的配置命令在新的服务器上是否可用，无需关心新网卡的驱动是否正确，所有这些让人头疼的麻烦全都没有了，你说棒不棒？反正我觉得好极了！

1.6.3高可靠性

数据中心的可靠性同投资一般成正比关系，配置了备用电源的机房肯定比普通机房可靠，而在不同的物理地点部署备份机房可以提供的灾难恢复时间一定比单一的数据中心短。每个人都希望得到高可靠的数据服务，但不是每个公司都有能力为数据中心做那么大的投资。

如果你的是一间中型服装公司的机房管理员，你所掌管的机房内服务器数量一般在 100 台以下，很少有老板会为这样规模的 IT 系统投资两地三中心（高级别的数据中心模式，即同城两个数据中心互为实时备份，异地再增加一个数据机房提供非实时备份）的备份机制，可是一旦机房瘫痪，你所承担的压力一点不比那些大企业的 IT 部门主管小。

云计算为那些没有能力建设高等级灾备设施的用户提供了另一个选择。也许你已经有过这样的经历，历年来周游各地的照片已经积累了几千张，数量有好几个 GB，这些照片记录了你生活中最宝贵的时光，保存在什么地方好呢？普通的电脑硬盘和外置硬盘的故障率太高，光盘介质的稳定保存时间只有两三年，而在线相册则不存在这些问题。目前网上有大量的在线相册服务，不但容量几乎无上限，而且只要服务商不倒闭（这个几率比你摔坏移动硬盘的几率要低得多）就能够安全地存放你的照片，即使你自己的设备断电、损坏，这些照片仍然安全地呆在在线服务商的存储空间内。

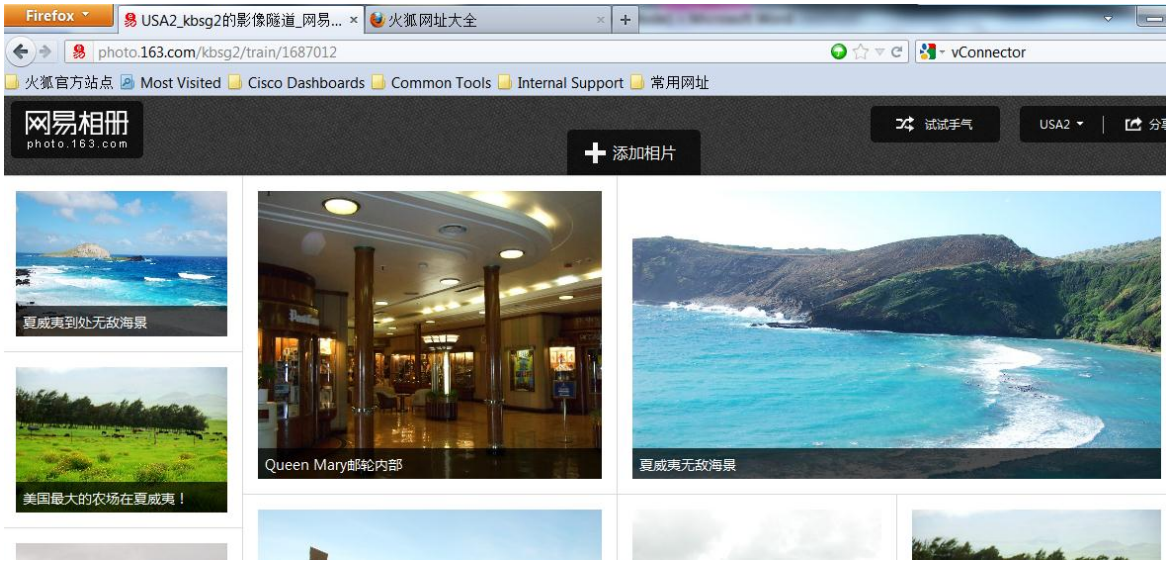


图 1-13 在线相册几乎可以永久保存你的相片

云计算服务供应商拥有高等级的机房设施和完善的备份机制，云计算用户虽然只是用了这其中的一部分资源，但是他得到的服务是一致的。云计算的基础设施虽然部署在用户看不到的地方，但它能提供的高可靠性等级往往比用户自建的机房还要高。

1.6.4远程访问

在讨论云计算的五大要素时，我们指出云计算通过网络向用户提供服务，基于网络的服务方式既是云计算的特点，也是云计算的优势。

互联网经过十多年的发展已经遍布了世界的每一个角落，只要用户可以接入网络，就能够享受云计算服务。通过网络远程分发服务的方式将极大地改变 IT 服务的提供方式，以往需要将所有人集中到一起举行的会议，现在参会人员可以分散在世界各地，花费在差旅上的时间更少，用于开展业务的时间更多。

1.6.5模块化

云计算通常以模块化的方式提供服务，例如用户可以在邮件、CRM、OFFICE 等多种服务类型之间自由组合，根据用户自身的情况选择在适当的时间、适当的种类和适当容量的云服务内容。

1.6.6高等级服务

服务质量是云计算另一个独特的优势。一般来说，对外提供大规模云计算业务的公司都拥有较强的技术实力，相较普通企业的 IT 部门，这些云计算服务商拥有大批经验丰富的技术专家。维持这样一个专家队伍的费用是很高的，如果他们单为一家公司服务，成本会让大多数机构的财务主管们心脏病发作。而云计算服务提供商的专家队伍不是为一个客户服务，他们可以同时为大量的用户提供自己的专业知识，这样平摊到每个用户头上的成本就很低了，而对于最终用户来说，则享受了以前他们以前从来不可能接触到的专业级别服务。

第2章　灵活的网络通道

在云计算网络的三大基础要素里面，最后一项是网络的灵活性。互联网自打出生便是一个动态选路的系统，云计算的基础是虚拟化，虚拟化带来的高可靠性和灵活性是云计算的 DNA，而虚拟化的特征之一就是资源的动态迁移。进入云计算时代，网络的动态属性需要被进一步放大。当计算资源随着请求和后台系统的变化而在不同的地理位置之间切换时，基础网络随之做出相应的反应是非常重要的，只有一个足够灵活的网络通道才能够从容应对这种资源的动态调度，从而将云计算的优势充分挖掘出来。

本章将从 DNS 和健康路由注入开始介绍传统网络在计算位置变化的情况下，如何做出应对，并逐渐深入，带出 LISP 的基本思想和实现方式。接下来的部分还会介绍 LISP 的适用场景，以及开发 LISP 的迫切性。

2.1　移动性是云计算网络的基本特征

Google 在 2006 年正式发布了 Google Docs，一个通过浏览器就可以编辑文档的 WEB 服务，只要电脑能够联上互联网，就可以撰写文章、编辑表格，在办公室没做完的工作可以保存在云端，回家后换台电脑联网就能够继续工作，Google Docs 带来的是革命性的便捷体验。

Google Docs 无需用户电脑上没有安装任何软件，所有的处理都在 Google 的云端服务器进行。不久之后，这种服务如雨后春笋般涌现出来，从文字处理到商业流程，几乎想得到的传统软件都有对应的 SaaS 产品，这个行业挤进了微软、思科一类的传统 IT 大佬，还孕育出 Salesforce.com 这样的新兴领军人物，SaaS 的商业模式已经撑过了黎明前的摸索，进入高速发展期。

在所有这些成功的故事中，广域网络都扮演了一个重要的角色，由于 SaaS 更加依赖网络来传递服务，因此对网络的灵活性提出了更高的要求。

想象一下，Google 为了保证 Google Docs 的服务质量，会部署多个数据中心，这些数据中心位于不同地点，互相之间互为主备，当其中一个出现问题时，在线用户仍然能够登陆到剩余的数据中心完成文档编辑工作，这就要求底层网络能够灵活、高效地将用户请求在任意时刻送到任意地点。

这个过程说起来简单，但当范围扩张到一个大型网络时就不容易了。对于一般企业用户来说，在自建私有云时不可能大量购买 CDN（Content Delivery Network）服务（内容分发网络，用于克服互联网上的性能和带宽瓶颈），更倾向于利用现有网络资源打造一个灵活的平台。因此近年来，那些以往只有超大型用户才关心的广域网负载技术又重新变成网络发展的热点。

由于 OSI 模型的每一层都有自己的寻址机制，在每个层次上都可以实现对网络流量的重定向，所以有不止一个方法可以架构出广域网上的灵活负载机制。不管哪种方法，其基本思路都是根据发起请求的用户位置，将流量送到最合适的数据中心，并且在数据中心发生故障时，尽可能快地利用备份中心来吸收请求流量。

目前比较成熟的两种方式是基于 DNS 的重定向和健康路由注入，下面我们来看看这两种方式的具体实现模式。

2.2 现有解决方案 1——DNS 重定向

DNS 是计算机域名系统 (Domain Name System) 的缩写，它是由解析器和域名服务器组成的。大家都知道，当我们在上网的时候，通常输入的网址其实就是一个域名。我们向一个 WEB 服务器中请求 WEB 页面时，我们可以在浏览器中输入网址或者是相应的 IP 地址，例如我们要上新浪网，我们即可以在 IE 的地址栏中输入网址，也可输入 IP 地址，但是这样子的 IP 地址非常不利于记忆，而计算机网络上的计算机彼此之间只能用 IP 地址才能相互识别，域名的作用就是替代 IP 地址，这样的域名会让我们容易记住。

域名与 IP 地址之间是一对一（或者多对一）的关系，域名虽然便于人们记忆，但机器之间只能互相认识 IP 地址，域名与 IP 地址之间的转换工作称为域名解析，DNS 就是专门完成域名解析的系统。

DNS 采用的是标准的 C/S 架构，当用户在一台联网 PC 的浏览器内首次敲入 www.sina.com.cn 时，PC 本身并不知道要到哪里去获取新浪首页的数据，因此它会向 DNS 服务器发出查询请求，DNS 服务器是架设在互联网上专职域名解析的设备，DNS 服务器将新浪的 WEB 服务器 IP 地址发送给 PC 机，PC 机随之向这个地址发起 HTTP 请求，这样浏览器就能顺利取得新浪的首页内容了。

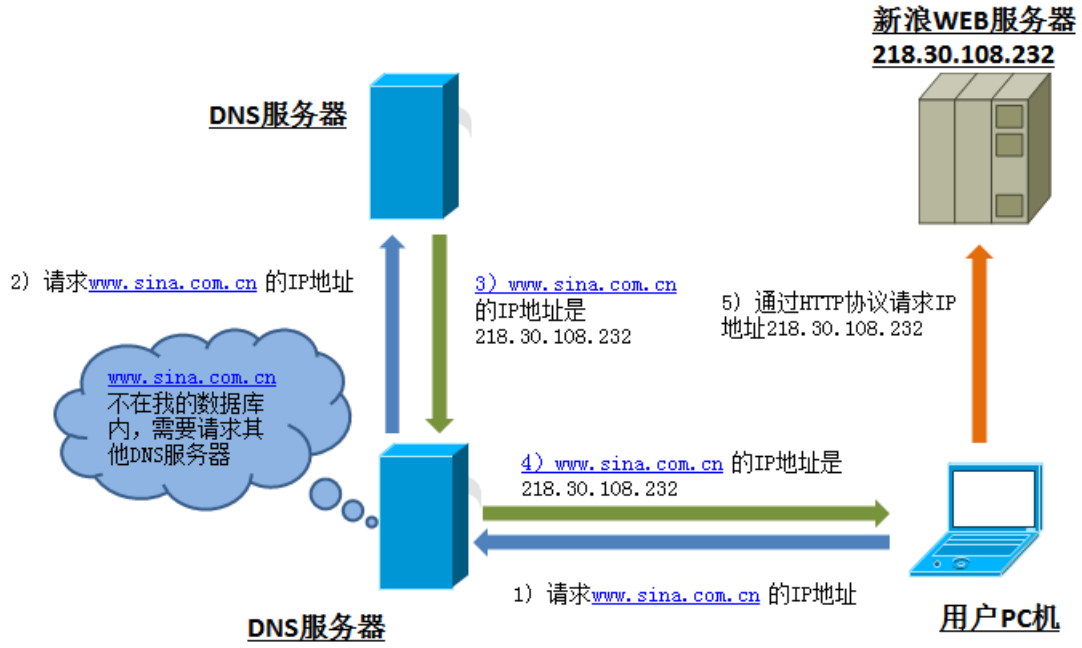


图 2-1 DNS 请求过程

由于客户端的请求对象完全由 DNS 服务器的响应内容决定，因此 DNS 服务器自然被考虑用来作为流量调度的节点。

DNS 重定向（DNS Redirection）的思路是在 DNS 服务器中动态地维持几个后台数据中心的出口 IP 地址，针对一般的用户请求返回优先级最高的出口 IP，如果其中一个数据中心出现问题时，DNS 服务器便将其对应的 IP 地址暂时停用，当收到后续的用户 DNS 请求，便将流量导向仍在正常工作的数据中心，从而在广域网上实现后台资源的负载均衡。

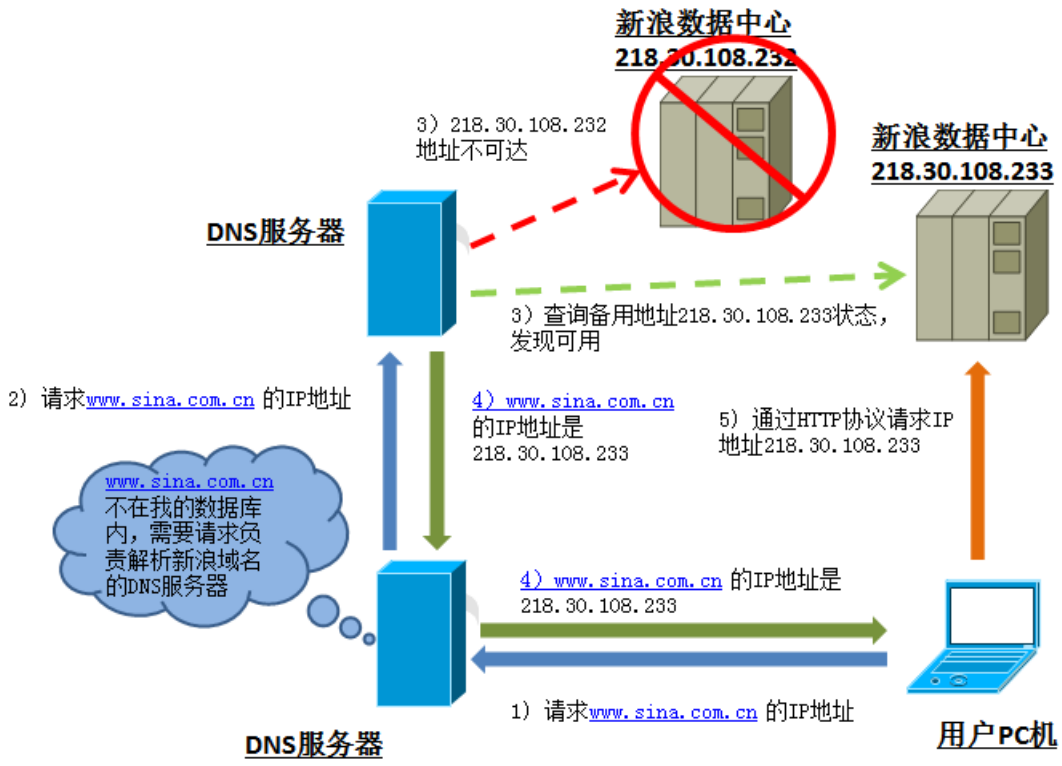


图 2-2 DNS 重定向过程

DNS 重定向的实现比较简洁，对现网结构几乎没有改动，在大规模的互联网业务中得到了广泛应用。

但 DNS 重定向远不是一个完美的解决方案，首先，DNS 重定向最适合基于域名解析的 WEB 业务，对于互联网上那些非 WEB 业务，这种方案就显得不那么方便了。

其次，DNS 的大部分流程都在电脑的操作系统和 DNS 服务器之间进行，与路由器或者交换机这样的专用网络设备没有交互，网络设备只能被动地接受被改变的网络流量，无法主动参与到这个过程中进行优化，发生在 2009 年 5 月 21 日的暴风影音 DNS 劫持事件就是一个典型案例。当时，暴风影音使用的 DNS 服务商 DNSPod 遭到黑客攻击瘫痪，导致全国的暴风影音客户端集体转向电信的 DNS 解析服务器发起请求，由于暴风影音的装机量巨大，电信服务器很快便撑不住宕机，结果最后演变为影响到南方六省的大面积断网故障。在这个特定事件中，孤立的用户端 DNS 行为完全没有考虑到底层网络的负载能力，结果酿成严重后果。今天的运营商都在努力转型，以避免成为单纯的“管道工”，DNS 这种和底层网络割裂的行为与网络运营商的演进方向是不一致的。

正是由于 DNS 重定向存在继续完善的空间，出现了其他的广域网负载均衡技术。

2.3 现有解决方案 2 - - 健康路由注入

DNS 是一种应用层与 IP 层结合的技术，实际上传统的 IP 层机制本身也具备一定的灵活性。对于 IP 来说，调整数据网络从来不是个问题，最原始的动态路由协议都可以随机选择数据转发的路径。动态路由协议关注的是链路本身的变化，当数据中心的出口链路中断时，动态路由协议可以在极短的时间内识别出来，并将流量引导到新的可用路径上。但是动态路由协议只有发生链路级别的故障时才会发生作用，如果数据中心的网络通道正常，但是内部服务器出现问题，客户端的请求仍然会源源不断地发送到发生问题的服务器地址，这显然是不合理的。

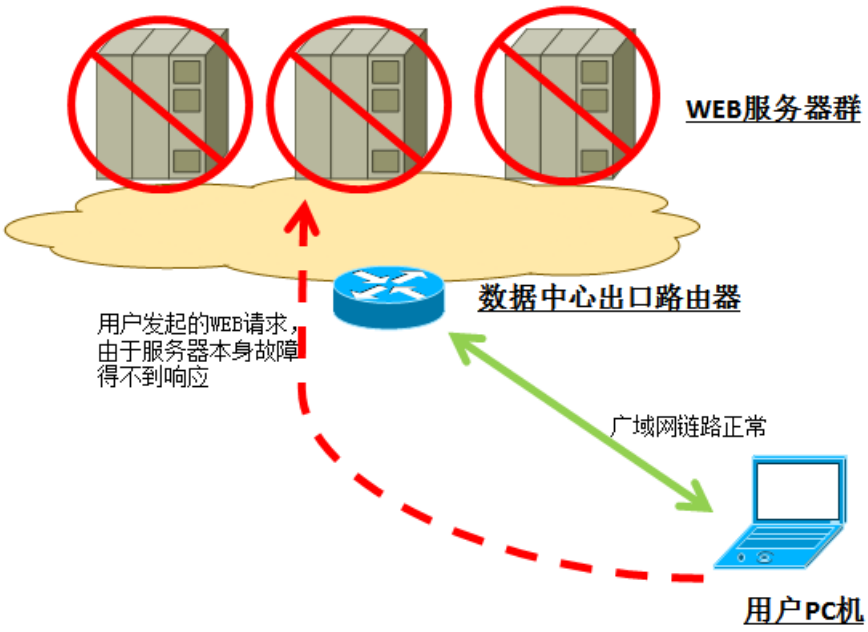


图 2-3 数据网络无法感知业务层面的故障

网络设备本身没有感知上层业务的能力，因此传统的网络机制同业务层面是割裂的，这也导致了很长一段时间内，业务规划人员只会考虑使用类似 DNS 重定向的机制来完成流量的动态迁移。

随着客户需求的变化以及网络技术本身的发展，网络架构不再仅作为“数据管道”存在，现代网络设计的一个重点就是“对上层业务的感知能力”，数据转发的依据除了路由协议计算的结果，还开始考虑上层业务运行的状态。这种技术的典型代表就是健康路由注入 RHI（Route Health Inject）。

健康路由注入是网络设备厂家提出的一种调度机制，一般在链路负载均衡设备或者出口路由设备上实现。具备相应功能的网络设备会周期性地监测数据中心内部的业务地址，检测的方法通常是发送定时查询，如果业务地址能够正常返回确认信息，表明业务运行正常，如果发送的请求包无法在一定时间内获得反馈，则说明上层业务发生了故障，不能再把用户请求转发到这个地址，需要对数据转发途径进行调整了。

2.4 LISP 是对现有方案的改进

DNS 重定向和 RHI 都是对网络通道的一种改进，虽然他们能解决部分问题，但这些方式往往只是单个产品级别的解决方案，无法在全网范围内构建一个灵活的架构，更重要的是，它们更多地是被动地应对网络环境的限制，当数据中心内部的资源主动发生漂移时就很难应对了。例如，当一批虚拟机因为系统升级临时从位于甲地的机房漂移到乙地时，无论 DNS 还是 RHI 都很难保证原有的用户请求随之也自动送往乙地。正是由于云计算资源地理位置的不确定性，导致传统固定地点的数据网络应付移动的后台资源显得力不从心。

LISP 就是为了改变现状做出的一个尝试。

LISP 的全称是 Locator/Identifier Separation Protocol，即位置/身份分离协议，其目的是直接从 IP 协议层下手，提升数据网络在云计算环境下的表现。要解释 LISP 的工作，首先要弄清楚为什么现有的 IP 网络难以应付云计算中资源的频繁移动。

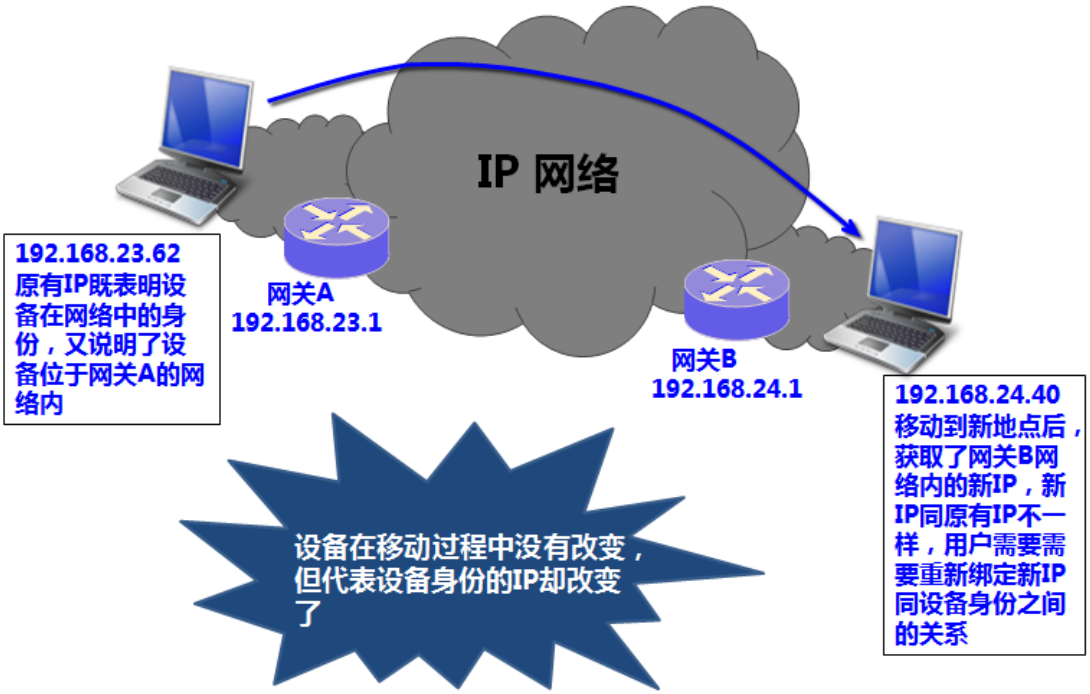
在 IP 网络诞生之初，所有的计算设备都通过固定的网络线缆连接到一起，每台设备分得一个 IP 地址，所有发出的数据包都标注了本机唯一的 IP 地址，路由器/交换机依照这个地址进行寻路和转发，所有的服务器一旦上架就几乎不会再移动位置，这个 IP 地址既说明设备在网络中的位置又标注了设备的身份，一举两得。此后很长一段时间内，这套机制都运转正常，直到云计算的浪潮开始涌现。

云计算的一大特点就是计算资源的虚拟化，单个应用程序不再绑定在一台服务器上，一个虚拟机在整

个生命周期内可能在数台物理服务器上驻留过，这种移动性变得越来越简单、越来越普遍，最终形成了云计算后台资源与地理位置的分离。这种分离对基础架构的影响是深远的，传统的 IP 地址规则此时就受到了挑战。

当一台虚拟机发生漂移时，其身份在移动前后保持一致，而寄存的服务器位置发生了变化，然而由于 IP 地址的天然属性，既代表了设备的身份又表明了其在网络中的位置，如果在漂移后修改 IP 地址，必然导致上层业务对虚拟机的身份重新识别，如果不改变 IP 地址，则虚拟机在新的地址空间内可能完全无法通信。传统 IP 地址规则身份与位置的统一，使其在云计算环境中显得格格不入。

图 2-4 位置的变化导致 IP 的变化



2.5 LISP 的核心思想 - - Map-and-encap

为了改进这一局面，IAB's（Internet Architecture Board's/互联网架构委员会）率先于 2006 年提出了对 IP 地址空间的改进建议，通过这几年的发展，这种尝试最终归结为一个比较统一的思路，即将传统的地址分拆为表明位置的 RLOCs（Routing Locators/路由标识符）和表明身份的 EIDs（Endpoint Identifiers/节点标识符），也就是 Loc/ID split 机制。

ROLCs 定义了设备是如何接入网络，如何能被找到，EIDs 则定义了设备是谁，属于什么组织。这两个信息并存在一个地址内，但互相之间保持独立，打破了原有的位置与身份之间的纽带。在这种地址空间中，站点发出的数据包被打上两层包头，内部包头为 EIDs，外部包头为 ROLCs，网络设备依靠 ROLCs 将数据包送到目的地附近，再去除外部包头，将原始数据包送到目的站点，这种模式被称为 Map-and-encap（映射与封装）机制。由于 Map-and-encap 采用的是两层封装的方式，大部分现网设备无需做很多改动，只需升级网络边缘的路由器使其支持数据包的封装/解封装即可。

这种思路就像下图中的酒店门牌号，包含名称和位置两个部分，Map-and-encap 模型不断发展的结果就是 LISP。

图 2-5 包含名称为位置两部分信息的酒店门牌号

2.6 LISP 的基本架构

LISP 是由 IETF 主导发展的新一代 IP 地址空间框架，LISP 精确地继承了 Loc/ID split 的精髓，基于 Map-and-encap 定义了一整套全新的 IP 地址路由机制。

在 LISP 网络中每个站点都有独立的 EID，这些站点将数据包发送到离它最近的 LISP 边界路由器上完成通信。为了完成这个工作，LISP 在传统的 IP 网络中添加了两个重要的新网元

- **ITR（Ingress Tunnel Router /入向隧道路由器）**
- **ETR（Egress Tunnel Router /出向隧道路由器）**

ITR 部署在 LISP 网络的边界，接受非 LISP 站点发来的数据包，并添加上 ROLCs，并依据 ROLCs 做出转发决定；ETR 的工作恰恰相反，坐落于 LISP 数据路径的最后一站，ETR 将接收到的数据包去掉 ROLCs 信息，还原成普通 IP 包，并转发给非 LISP 站点。在很多网络设计中，LISP 网络的边缘路由器会同时身兼 ETR 和 ITR 两种功用，也被成为 XTR。

作为连接 LISP 和非 LISP 网络的网关，XTR 的一个重要工作就是维护 EID 和 RLOCs 之间的对应关系。每个 XTR 都会保存一张 EID-RLOCs 表格，这张表说明了每个 EID 在 LISP 网络中的最终位置，如果 XTR 收到一个不存在于 EID-RLOCs 中的陌生 EID，那么说明这个 EID 指代的主机连接在哪个 ETR 上还未可知，或者这个 EID 根本不存在。XTR 收发数据包的唯一依据也就是 EID-RLOCs，EID 表示最终目的地，RLOCs 则类似 MPLS（MultiProtocol Label Switching/多协议标签交换）中出方向的 LER 地址。

同常规路由表中等价路由的概念一样，一个 EID 可能对应多个 RLOCs。每一条 EID 与不同 RLOCs 的对应关系都带有优先级（P）和权重（W）两个属性。P 告诉 ITR 应该优先使用哪个 RLOCs，W 说明了怎样在多个目的 EID 相同的 RLOCs 之间分配流量。如果一个 EID 对应每个 RLOCs 的 P 和 W 值都一样，那么流量将在通往所有 ROLCs 的链路上被均匀分配。

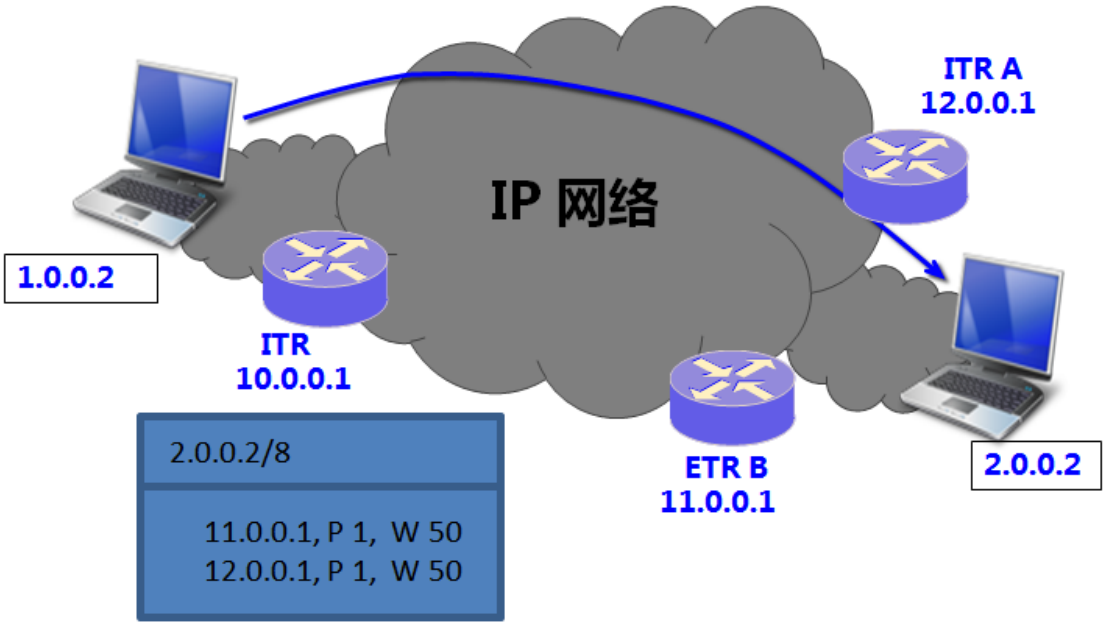


图 2-6 到同一个 EID 可能存在多个等价 ITR

2.7 LISP 的新包头

相较于传统的 IP 包，LISP 最大的变化是将 IP 包头分为外层包头和内层包头。外层包头即 ROLCs，表征离目的 EID 最近的 LISP 站点，通常是一个 ETR；内层包头携带 EID 信息，是一个非 LISP 的常规站点。LISP 原生支持 IPv4 和 IPv6，并且可以实现在 IPv6 的 ROLCs 上传送 IPv4 的 EID 数据包，或在 IPv4 的 ROLCs 网络上传送 IPv6 的 EID 数据包。

外层包头	版本	包头长度	服务类型	数据包总长度	
	标识符			标记	分段序号
	生存时间		协议=17	头部校验	
	源ROLC				
	目的ROLC				
UDP包头	源端口			目的端口	
	UDP长度			UDP校验	
LISP包头	LISP控制字段				
	LISP认证字段				
内层包头	版本	包头长度	服务类型	数据包总长度	
	标识符			标记	分段序号
	生存时间		协议类型	头部校验	
	源EID				
	目的EID				

图 2-7 LISP 包头结构

LISP 基本上复原了 Loc/ID split 的思想，作为一个协议，LISP 的意义在于找到将一种思想具体化的方

式，以及以最优的效率解决在这个抽象到具体的过程中可能发生的问题。

2.8 通过 LISP-ALT 实现可扩展网络

了解了 LISP 的基本原理后，我们来看看 LISP 的高级特性，这其中最重要的一点就是协议本身的可扩展性。

可扩展性是衡量一个网络协议的基本指标之一，一个能处理好 10 个节点通信的网络不一定能够应对 100 个节点之间的数据交互，这其中增加的难度往往不止是将处理能力提高 10 倍这么简单。LISP 在计算数据包的目的路由时需要找到 EID 与 ROLCs 的对应关系，由于 EID 和 ROLCs 是两个独立的地址空间，他们之间可能没有任何规律，因此 EID 和 ROLCs 之间是一个近似 Full Mesh 的对应关系，一个 LISP 网络的边缘路由器即 XTR 需要知道对应每一个站点 EID 的 ROLCs 是多少，才能够正确地转发数据包。随着 LISP 网络的扩张，这张对应关系表变得越来越长，当收到新的数据包时，每次查找所用的时间也越来越大。当网络规模达到一定程度时，在一个路由器肚子里装下整张大网的拓扑关系就显得不太不现实了，传统的动态路由协议如 OSPF、IS-IS 利用了 IP 地址空间的层级关系，将一个网段内的地址合并为一条路由，配合链路状态值，大大缩减了路由器需要了解的路由表长度。但在 LISP 中，EID 和 ROLCs 之间没有层级关系，传统的方法不再起作用，需要新思路来解决新问题。

LISP 给出的答案是 LISP-ALT（LISP-Alternative-Topology），LISP-ALT 是一个架构在基础网络上的虚拟拓扑，专门用来找出 EID 与 ROLCs 的对应关系。凡是 LISP 的 ITR 和 ETR 都加入到 LISP-ALT 中，他们之间形成 EBGP（External BGP/外部边界网关）邻居关系，所有的 BGP（Border Gateway Protocol/边界网关路由协议）信令都通过 GRE 协议封装，这些 ITR 和 ETR 就利用 BGP 交换 EID 的可达信息。换句话说，LISP 节点之间运行封装的 BGP 协议，在现有网络上形成一个叠加的网络，ITR 和 ETR 通过这个叠加的网络学习 EID 和 ROLCs 的对应关系。在传统的路由协议中没有地方可供 LISP 传递 EID 和 ROLCs 的对应关系，有了 LISP-ALT 之后，LISP 的节点就有了一个专用渠道交换 EID 的可达信息。

新的 IP 包头和 LISP-ALT 构成了 LISP 协议的精髓，正是这两个要素保证了一个 IP 数据包能够通过一种完全不同的方式被安全送到目的地。当一个 ITR 收到一个节点发送过来的数据包后，它首先查看自己的缓存内是否有目的节点的 EID-ROLCs 对应关系，如果这个关系存在，则 ITR 为数据包加上外层包头，写入找到的 ROLCs 并转发出去，对应的 ETR 收到 IP 包后，发现目的 ROLCs 正好是自己，于是去掉外层包头，将原始数据包送给目的站点。

在 LISP 网络内可以配置一组专门用于路由查询的路由器，如果 ITR 在缓存内找不到目的 EID 与 ROLCs 的对应关系，则它会在外层包头里写入目的站点的 EID，然后通过 LISP-ALT 发送给一个指定的 LISP 路由器，这个路由器被授权完成 EID-ROLCs 的对应工作，保存有比较完整的 EID-ROLCs 信息，当指定的路由器收到这个数据包后，发现外层包头和内层包头都是同样的 EID，于是它判断出这是一个查询报文，于是，它首先将数据包去除外层包头并送往目的站点，然后向 ITR 发送一条应答报文，说明目的 EID 和 ROLCs 的对应关系，ITR 收到这条报文之后会将这个 EID-ROLCs 关系缓存下来，接下来收到的 IP 包就可以直接发送给对应的 ETR 了。

通过这样一个过程，在 LISP 网络中只需要设置几个指定的路由器保存比较完整的 EID-ROLCs 关系，其他 ITR 和 ETR 只需要保存常用的对应关系，如果碰到不认识的 EID，只需要向指定的路由器发起查询即可，保证了大部分路由器上路由表项的简洁。

2.9 一个 LISP 转发实例

让我们通过一个数据包的转发实例来真正掌握 LISP 的原理。

如下图所示，主机 A 准备向主机 B 发送一封电子邮件，它将数据包的源地址和目的地址分别填写为自

己的 EID 1.0.0.2 和 B 主机的 EID 2.0.0.2，然后发往路由器 R1。

R1 是一台启用了 LISP 协议的 ITR 设备，收到 A 发来的第一个数据包后，它发现自己的 EID 与 RLOC 对应关系表中没有 B 主机 EID 2.0.0.1 的信息，为了让这个数据包能够通过 LISP 网络到达对端，R1 为这个数据包封装上一个新的 LISP 包头，将自己的 RLOC 10.0.0.1 写入外层包头的源地址段，而外层包头的目的地址段仍然保持与内层包头的目的地址一致，即 B 主机的 EID 2.0.0.1。

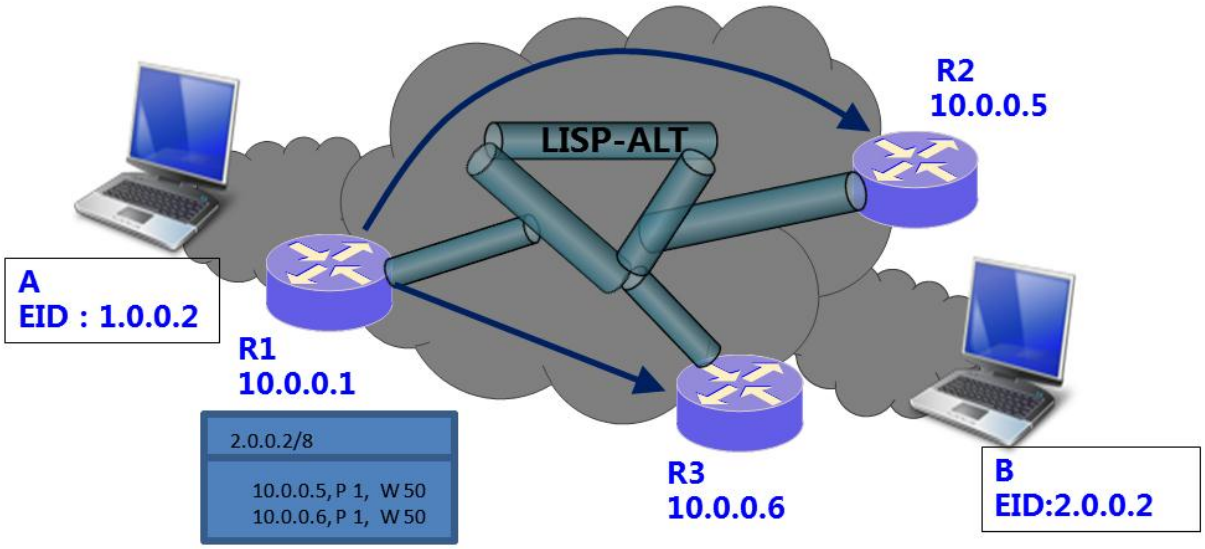


图 2-8 LISP 转发实例

R1 随机将这个数据包作为一个查询报文发送到 LISP-ALT 中。经过两次封装的数据包在 LISP-ALT 网络中通过标准的 BGP 路由来到 ETR 设备 R2，R2 同时是 B 主机的 LISP 网关，R2 将外层包头接封装后，直接送给 B 主机。同时，R2 发现这个数据包的两层包头具有相同的 EID，是一个通过 LISP-ALT 处理的路由查询，于是 R2 向外层包头源地址 RLOC 10.0.0.1 代表的 R1 发送一个 EID 到 RLOC 的更新报文，这个报文说明了从 R1 到 B 主机的两条路径分别是通过 ETR R2（10.0.0.5）和 ETR R3（10.0.0.6）。

R1 收到 R2 返回的报文，得知数据包已经顺利达到目的地，并且根据报文内的信息更新了自己的 EID-RLOCs 对应关系。对于 A 主机那封电子邮件剩余的数据包，R1 就能够根据更新的路由信息，直接发送给 R2 和 R3 了。

2.10 LISP 的应用场景

利用 LISP，一个基于 IP 的应用程序理论上可以部署在世界上任何地方，而不受到地址空间的限制，上层应用看到的 IP 地址不过是内层包头里的 EID，真正决定 IP 包路由去向的是外层包头里的 ROLCs。

在大规模的虚拟机环境中，LISP 能够在以下几方面帮助云计算获得极大的灵活性：

2.10.1 IP 终端的灵活移动

LISP 在 IP 层面提供了前所未有的灵活性，使用固定 IP 地址的虚拟机可以在任何一个数据中心内完成初始化，然后根据需要自由地漂移到其他地方。在漂移前后，最终用户感受不到后台系统的迁移，所有上层应用绑定的网络地址都保持一致，而所有的数据包始终保持在最优路径上被转发。

LISP 使得路由策略随着虚拟机的移动而自动调整优化，这种程度的移动性是其他技术难以提供的。只要有了 LISP，虚拟机就可以带着初始分配的 IP 地址在不同的网段和物理地点之间移动，无需任何辅助

性的路由策略来帮助优化移动前后的转发路径。简单高效的机制使得网络得以成为大范围资源调度的平台。

由于 LISP 允许终端在网内自由地移动，这去除了终端 IP 地址的地理属性，这个 IP 地址仅仅表明了设备的身份。对于网络管理来说，这大大简化了网络的设计和规划过程。如果你有过规划大型网络的经验，你一定碰上过 IP 地址不足的困扰。在 LISP 架构下，IP 地址的分配变得更加自由，在设计终端设备使用的 IP 地址时只需要考虑其身份属性，而不用担心机架内是否有足够的地址段，因为终端的 EID 对网络流量的路由不造成影响。

LISP 的这种灵活性给了网络设计人员极大的想象空间，未来 LISP 在以下几个领域很可能大展拳脚。

- **灾备数据中心的快速启用**

  当生产数据中心发生问题时，通过 LISP 能够迅速将业务切换到备份中心，而无需改动业务层面的 IP 地址，从而大大缩短了业务重启的时间；

- **突发的云计算事件**

  当面对突然爆发的业务请求时，LISP 可以协助将虚拟机从资源较少的数据中心快速迁移到资源富裕的数据中心，实现业务的快速响应和动态增长；

- **多地数据中心**

  建在多个地理位置的数据中心之间可以方便地交换虚拟机负载，实现更加灵活的维护与升级策略。

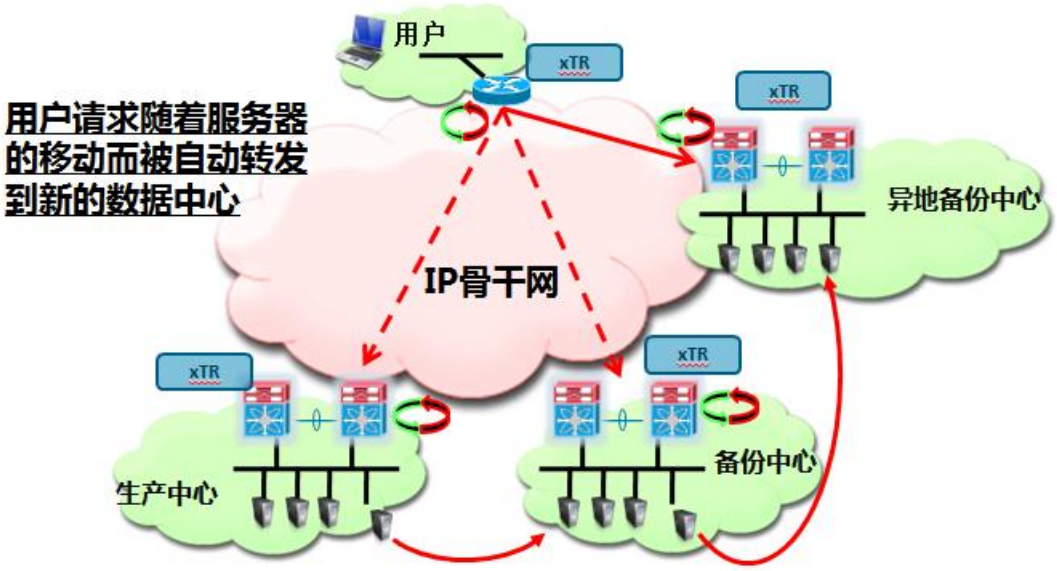


图 2-9 用户请求随着资源而移动

2.10.2 IPv6-IPv4 混合部署

当第一代网络专家摸索互联网的模型时，他们根本没有想到这项技术日后爆出的巨大威力，当时他们觉得 32 位的 IPv4 地址已经足够了。结果因特网的发展远远超出了这些专家们的想象，IPv4 地址段一共包含 400 多万个可使用的地址，这些地址已经在 2011 年 2 月 3 日被分配殆尽。为了满足不断增长的联网设备的需求，IETF 制定了新的互联网地址空间 IPv6，IPv6 的地址长度从 IPv4 的 32 位一举扩张到 128 位，能够提供 2 的 128 次方个地址，号称“能为地球上的每一粒沙分配一个 IP”。自 2011 年开始，Internet Society，即因特网协会将每年 6 月的一天定为 World IPv6 Day（世界 IPv6 日），在这一天的 24 小时内，测试在互联网上真实部署 IPv6 地址的效果和产生的影响。IPv6 已经从纸面上的概念一步一步走入现实社会，我们使用的互联网将逐步开始向 IPv6 地址迁移。

IPv6 庞大的地址空间可以彻底解决 IP 地址不足的问题，但是它与现有的 IPv4 之间有大量的兼容性工

作要做。目前互联网上几乎所有的应用都是基于 IPv4 编写的，怎样将这些基础设施迁移到 IPv6 的世界中是一个巨大的挑战，如果处理不得当，IPv6 的进程将被大大拖累。

科研机构和网络厂家提出了各种各样的 IPv4-IPv6 技术，LISP 虽然不是一个专门用于迁移 IPv4 的技术，但凭借其独特的机制，LISP 在 IPv6 与 IPv4 迁移方面的优势开始被越来越多人重视。

大部分 IPv4-IPv6 技术的原理都是将 IPv4 的地址转换或封装到 IPv6 环境中，如果你已经熟悉了 LISP 的工作方式，你会发现 LISP 天然就是一个强大的封装机制。LISP 的内外层包头之间没有耦合关系，两者完全可以使用不同的地址空间，这样一来 LISP 就能够用 IPv6 的外层包头封装一个 IPv4 的内层数据包，或者在外层包头内写入 IPv4 地址，而内层包头使用 IPv6。

当用户将自己的基础网络开始向 IPv6 过渡时，往往采用分批建设的方式。在 LISP 的帮助下，用户可以将部分网络替换成 IPv6 地址空间，这些网段的设备通过 LISP 网关与其余仍然使用 IPv4 地址的设备通信。

在下面这个例子中可以看到，所有的局域网设备已经开始使用 IPv6 设备，而广域网核心仍然运行在 IPv4 协议栈上，局域网与广域网之间是一个同时运行 IPv4 和 IPv6 双协议栈的 LISP XTR。LISP XTR 将设备的 IPv6 地址作为 EID，而它面向广域网的接口地址则是 RLOCs。当数据包在广域网上传输的时候，RLOCs 作为寻址依据，一旦数据包通过 XTR 进入局域网，IPv4 的 RLOCs 就被脱去，只有剩下内层的 IPv6 EID。

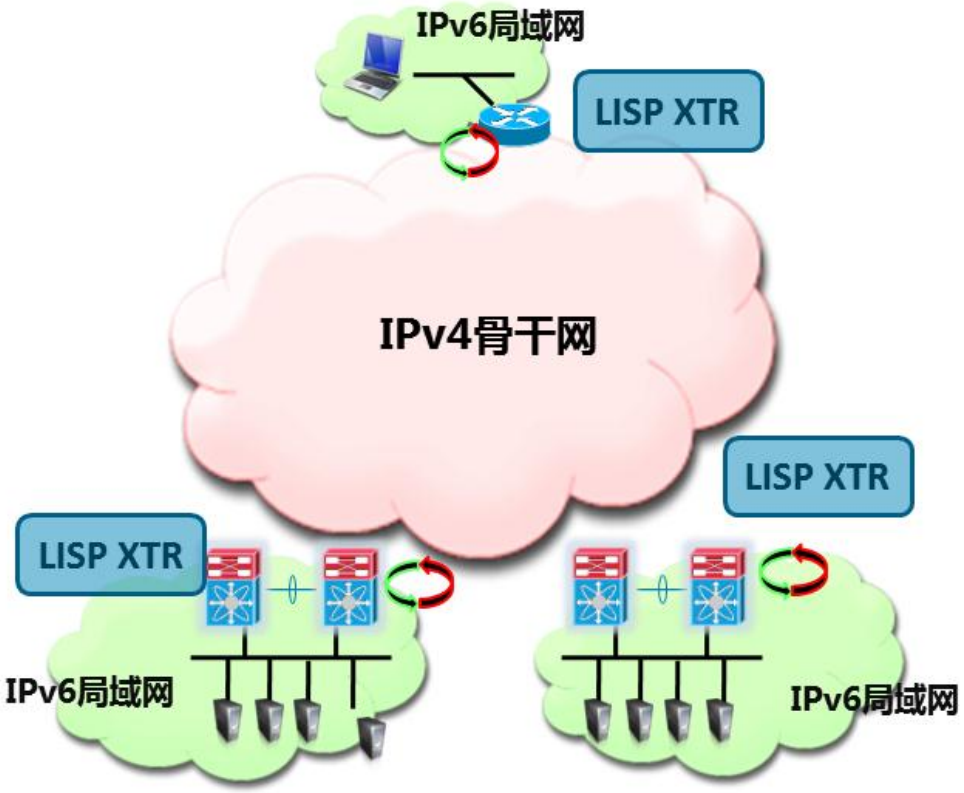


图 2-10 运行 IPv6 的局域网通过 LISP 网关同 IPv4 骨干网通信

LISP 对终端设备的透明性，保证了这种 IPv4-IPv6 的转换不会影响正常业务的进行，最终用户也感觉不到两个协议栈的存在，不失为一个简便、快捷的迁移方案。

另一个在 IPv6 中使用 LISP 的例子是将组织内部的服务发布到外部的 IPv6 网络上去。由于各地 IPv6 建设的速度不一致，很可能会出现骨干网络已经开始使用 IPv6，而企业或组织内部的网络仍然采用 IPv4 的情况（这种情况非常常见，因为骨干网只要统一升级核心路由器即可，而改变局域网的地址空间还需要变更每台电脑的网卡和驱动，牵涉面太广）。如果在局域网内部有一台 IPv4 的电子邮件服务器需要发布到 IPv6 骨干网上去，就必须在网络边缘做 IPv4-IPv6 的转换。

这个时候，同样可以利用 LISP 网关的能力，将内部的 IPv4 服务转换成 IPv6 服务发布到骨干网上。

使用 LISP 完成转换无需额外的硬件投资，部署于维护成本也相应要低得多。

2.10.3 多租户 VPN 环境

前文谈到了 LISP 的封装机制在 IPv4-IPv6 混合部署环境中具备天然的优势，实际上，这种优势还可以延伸到其他领域。LISP 的封装机制是 Map-and-Encap 的一种表现，Map-and-Encap 的本质是掩盖原始地址（内层包头地址），在转发数据的过程中采用新的地址（外层包头地址）完成路由，这与 VPN 的行为是非常类似的。因此，LISP 完全可以被用作一种多租户环境下的 VPN 方案。

多租户顾名思义就是在一个基础网络上承载多个用户，不同用户之间的流量严格隔离。当不同用户的流量在网络上传输时，需要用到 VPN 技术进行区分，常用于多租户环境的 VPN 技术包括 MPLS VPN、VRF-Lite（Virtual Routing and Forwarding-Lite/轻便虚拟转发实例）和 EVN（Easy Virtual Network/简单虚拟网络）等。

相较这些传统的 VPN 技术，LISP 的特点在于功能的丰富性，当我们需要在多租户环境下部署云计算业务时，相关的移动性、扩展性、甚至未来向 IPv6 的迁移全部可以由 LISP 这个单一协议框架搞定，从部署的角度来说要简洁得多，而对于 IT 运维人员来说，技术的简洁性永远是一个巨大的优点。

另一方面，LISP 的实现是一个单纯的二层包头机制，对现有的路由协议没有改动，数据转发的最优路径完全依靠动态路由协议对 RLOCs 的自动路由来完成。LISP 与路由协议之间的独立性，使得 LISP 网络在扩张时更加灵活，无需考虑路由策略的调整，这对于变化相对频繁的多租户环境是非常有意义的。当租户的网络规模扩张时，LISP 能够随之快速扩张，而不会受到路由策略的限制。

2.11 我们真的需要 LISP 吗？

通过以上的描述，实际也回答了另一个重要的问题

> *“为什么不采用传统的封装技术来解决现有的问题（例如 IPsec VPN）？”*

这也是我最喜欢的问题之一，技术标准的诞生与演进除了有产业力量的商业驱动之外，技术本身的特性是决定其成功与否的根本原因。

IPsec VPN 这样的技术确实能够将普通 IP 数据包的原始包头掩盖，目前最简单的 VPN 设备就能够让我们在任何地方通过互联网接入公司业务系统，并保留一个与公司内网网段一致的 IP 地址。问题是 VPN 自诞生之初就是一个点对点协议，其存在的意义是解决两点之间互联的需求，当我们把视野放到一个多点之间复杂互联的大网中，会发现在 VPN 协议中没有地方能够让网络设备动态地学习网络拓扑，并保持高效地更新，虽然某些网络厂家开发出一些类似的初级解决方案（如 Cisco 的 DMVPN 等），但仍然只是点对点连接的增强，这些修修补补的办法一旦遇上对扩展性要求极高的网络环境立刻会显得捉襟见肘。正是由于现有的网络协议中没有一个合格的候选者，才导致了 LISP 的平地而起。

在进一步解释 LISP 的与传统 VPN 的区别之前，我要花一点时间说明什么是“控制平面”和“数据平面”，这两个概念非常重要，在本书接下来的章节中将被反复提及。控制平面和转发数据平面又被称为 Control plane 和 Data Control，是网络设计中的两个基本元素。控制平面是做出转发决定的元素，换句话说，控制平面决定一个数据包该走哪条路才能到达最终目的地，控制平面可以是一个网络协议，典型的如 OSPF、BGP 等路由协议，或网络协议的一部分，如 LISP 中的 LISP-ALT，也可以用来指代网络设备的一部分功能，如路由器中计算路由条目的能力。而数据平面的功能则是执行控制平面做出的决定，数据平面同样可以是一个网络或网络协议的一部分，或是网络设备的部分功能，例如路由器的端口和转发芯片。如

果你将数据网络比作一个工匠，那么控制平面是他大脑中的专业知识和技巧，而数据平面就是他灵巧、强健的双手。控制平面决定了网络的智能化程度、可扩展性、收敛时间等，而数据平面则决定了网络的运行效率、转发效率等指标，几乎所有的网络技术和设备都能被区分为控制平面部分和数据平面俩部分。也许你第一次接触这两个概念觉得有些抽象，这很正常，不用着急，我相信你将在阅读本书的过程中更加深入地理解数据平面和控制平面的数据网络中的重要地位。

说回 LISP，LISP 较 IPsec 先进的地方就在于其具备了强大的控制平面，的每一个 ITR 和 ETR 都会维护 EID-ROLCs 的关系表，并且 LISP 还定义了 LISP-ALT 这样的架构专门用于控制平面的通信。可以说，LISP 好像是一个添加了控制平面的 IPsec VPN，正是由于有了完整的控制平面机制，LISP 才能够从容地以倍数规模扩张网络规模。这个情况同后文将要谈到的 TRILL/FabricPath 非常类似，其根本原因是云计算对底层网络在灵活性和扩展性上提出了新的要求，从而推动了基础架构开始发生翻天覆地的变化。

LISP 作为 IETF 主持开发的一项公开标准，志在成为未来云计算环境中的大规模网络扩张的基础技术，也是目前业界基于 Loc/ID split 结构的最成熟的实现。Cisco 在 2011 年已经宣布在 Nexus 7000 交换机上支持 LISP 协议，其后这个支持列表逐渐添加了 ASR、ISR G2 等 Cisco 的主流路由器产品上。除了 Cisco 之外，一个名为 LISPmob 的组织也在 LINUX 平台上开发 LISP 的开源软件平台，相信随着云计算产业的不断发展，会有越来越多的网络厂家加入到支持 LISP 的行列中来。