* 孙镜涛

阅读数：37582015 年 11 月 23 日

在之前的新闻《Netflix 企业文化与架构设计》一文中我们介绍了 Netflix 高扩展的企业文化与架构，而作为一家在线影片租赁提供商 Netflix 之所以能够在如此大量的用户、海量的数据、复杂的拓扑结构下取得成功，离不开其背后的开源文化与技术。Netflix 利用开源或者自己开发的技术，让公司能够在分布于全世界 50 多个国家的服务器上进行持续地构建和集成；能够对用户所产生的海量行为数据进行分析挖掘从而更好地推荐和改善自身服务；同时能够通过性能剖析和安全审计保护用户的隐私和数据，下面就让我们来盘点一下 Netflix 所使用或贡献的[开源技术](http://netflix.github.io/" \t "_blank)。

**大数据**

数据是无价的，为了向客户提供极致的服务，Netflix 拥有一套完善的大数据技术生态系统用于用户数据的分析和挖掘。Netflix 不仅使用 Hadoop、Hive、Pig、Parquet、Presto 以及 Spark 等被广泛采用的开源技术，同时还开发并贡献了一些其他的工具和服务。

* [Genie](https://github.com/Netflix/genie)：专为 Hadoop 生态系统定制的一组 REST-ful 服务集合，用于管理作业和资源, 它有两个关键的服务：Execution  
    
  Service 和 Configuration Serice。前者提供了 REST-ful  
    
  API，用于提交和管理 Hadoop、Hive 以及 Pig 作业；后者是一个 Hadoop 资源的有效储存库，处理元数据的连接以及运行资源上的作业。
* [Inviso](https://github.com/Netflix/inviso)：对 Hadoop 作业和集群的性能进行详细而深入的剖析。
* [Lipstick](https://github.com/Netflix/lipstick)：以一种清晰且可视化的方式展示 Pig 作业的工作流。
* [Aegisthus](https://github.com/Netflix/aegisthus)：是一个能够读取 Cassandra  
    
  SSTable 的 map/reduce 程序，Netflix 的大数据平台每天都会利用该工具从 Cassandra 中读取数据进行分析。

**重要通知：接下来 InfoQ 将会选择性地将部分优秀内容首发在微信公众号中，欢迎关注 InfoQ 微信公众号第一时间阅读精品内容。**

**构建和交付工具**

* [Nebula](https://nebula-plugins.github.io/)：Netflix 开源的 Gradle 插件集合，可以让开发者更容易地使用 Gradle 构建、测试和部署项目。
* [Aminator](https://github.com/Netflix/aminator)：EBS AMI 创建工具，支持 CentOS/RedHat Linux 镜像，利用该工具创建的镜像能够运行在 EC2 实例上。
* [Asgard](https://github.com/Netflix/asgard)：云部署和管理工具，该工具与 Aminator 配合使用，能够将打包后的 AMI 从开发者的桌面部署到 AWS 上。

**通用的运行时服务和类库**

Netflix 的大部分服务都是以云平台为基础和技术栈，云平台由云服务、应用程序类库和应用容器组成，每一部分 Netflix 都有相应的技术保障。

* [Eureka](https://github.com/Netflix/eureka)：Netflix 的云平台服务发现技术。
* [Archaius](https://github.com/Netflix/archaius)：分布式配置工具。
* [Ribbon](https://github.com/Netflix/ribbon)：弹性且智能化的进程处理和服务通信。
* [Hystrix](https://github.com/Netflix/hystrix)：提供单一服务调用所不具备的可靠性，提供运行时的延迟隔离和容错。
* [Karyon](https://github.com/Netflix/karyon)和[Governator](https://github.com/Netflix/governator" \t "_blank)：JVM 容器服务。
* [Prana sidecar](https://github.com/Netflix/prana)：提供实例内代理，支持非 JVM 运行时。
* [Zuul](https://github.com/Netflix/zuul)：提供云部署周边的动态路由、监控、安全和弹性扩展等服务。
* [Fenzo](https://github.com/Netflix/Fenzo)：为云本地框架提供了更为高级的调度和资源管理功能，为装箱和集群自动扩展提供了插件实现，用户能通过自定义的插件实现自定义的调度优化。

**数据持久化**  
  
面对每天数以万亿的行为数据，没有哪一种技术能够独自满足所有的用例，所有潜在的需求，为此 Netflix 不仅使用了 Memcached 和 Redis 等非持久化内存存储方案，还使用了可搜索的数据存储 Elastic，以及高可用的数据存储 Cassandra 和 MySQL。此外，Netflix 还创建了一些其他的辅助工具：

* [Raigad](https://github.com/Netflix/Raigad)和[Priam](https://github.com/Netflix/Priam" \t "_blank)：辅助 Elastic 和 Cassandra 集群的部署、管理、备份和恢复。
* [EVCache](https://github.com/Netflix/EVCache)和[Dynomite](https://github.com/Netflix/dynomite" \t "_blank)：用于大规模地使用 Memcached 和 Redis。
* [Astyanax](https://github.com/Netflix/astyanax)和[Dyno](https://github.com/Netflix/dyno" \t "_blank)客户端类库：便于更好地使用云端数据库。

**分析、可靠性和性能**  
  
在任何一个公司的运营中遥测和指标都具有至关重要的作用，高效的性能指令能够让工程师快速地从大量指标中找出自己所需要的那部分从而更快更有效地做出关键决策。除此之外，成本管理和资源在云端使用情况的可视化服务也不可或缺，Netflix 在这些方面使用的工具包括：

* [Atlas](https://github.com/Netflix/atlas)：时间序列遥测平台，每分钟接收超过 10 亿的指标
* [Edda](https://github.com/Netflix/edda)：跟踪云端变化的服务
* [Spectator](https://github.com/Netflix/spectator)类库：让 Java 应用程序代码能够更好地与 Atlas 集成
* [Vector](https://github.com/Netflix/vector)：以最小的代价获得高分辨率主机级指标
* [Ice](https://github.com/Netflix/ice)：监控当前成本和云使用趋势，让工程师能够更好地掌控自己的应用程序在当前环境中的状态
* [Simian Army](https://github.com/Netflix/SimianArmy)：Netflix 实例的随机故障测试，用于验证可靠性

**安全**  
  
对任意类型、任意规模的公司来说安全都是一件越来越重要的事情，Netflix 为开源社区贡献了大量的安全工具和解决方案，主要分为两类，一类是能够让安全团队更有效地保护大型动态环境的运维工具和系统；一类是为现代分布式系统提供严格安全服务的安全基础设施组件。

* [Security Monkey](https://github.com/Netflix/security_monkey)：用于检测和保护大规模的 AWS 环境
* [Scumblr](https://github.com/Netflix/scumblr)：利用因特网进行针对性的搜索定位特定的安全问题并调查
* [MSL](https://github.com/Netflix/MSL)：一个可扩展的、灵活的安全消息协议，可解决大量安全通信用例和需求

**用户界面**  
  
世界各地的 Netflix 会员会通过 TV、手机和桌面等各式各样的设备观看视频，为此 Netflix 使用了 Node.js、React 和 RxJS 等先进的 UI 技术构建富客户端应用程序。Netflix 通过数据驱动的 A/B 测试实验新的理念，理解每一个功能特性的价值，从而持续地提升产品。此外，Netflix 还使用了以下工具。

* [Falcor](https://netflix.github.io/falcor)：允许应用程序把所有远端的数据源用一个单独的 JSON Graph 来表示，提高了数据抓取效率
* [Restify](https://github.com/restify/node-restify)：一个基于 Nodejs 的 REST 应用框架，支持服务器端和客户端
* [RxJS](https://github.com/ReactiveX/RxJS)：JavaScript 的 Reactive 扩展库

感谢[杜小芳](http://www.infoq.com/cn/author/%E6%9D%9C%E5%B0%8F%E8%8A%B3)对本文的审校。