云计算环境下的可扩展架构

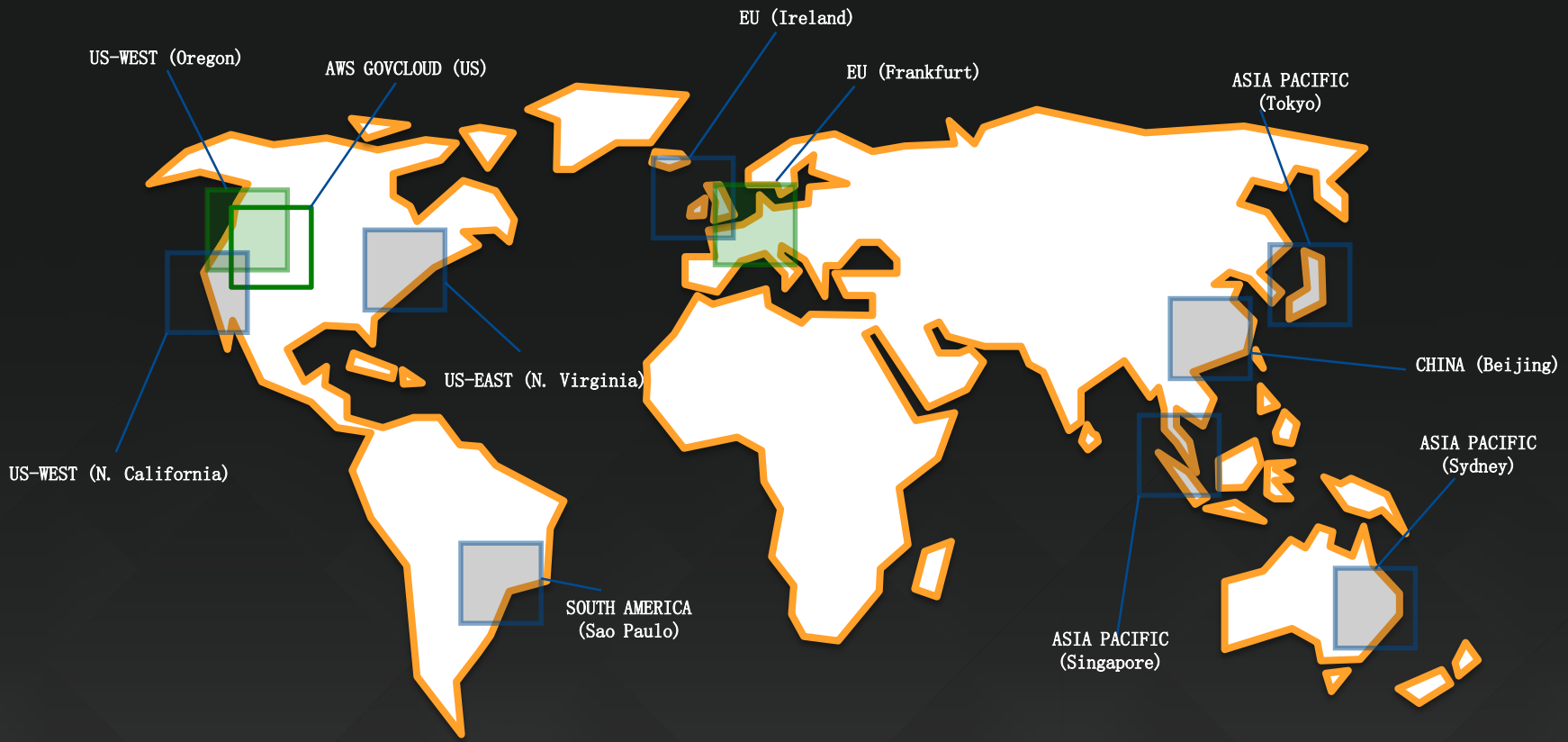
—满足1,000万用户的扩展性

王毅 AWS 解决方案架构师，区域主管

amazon web services

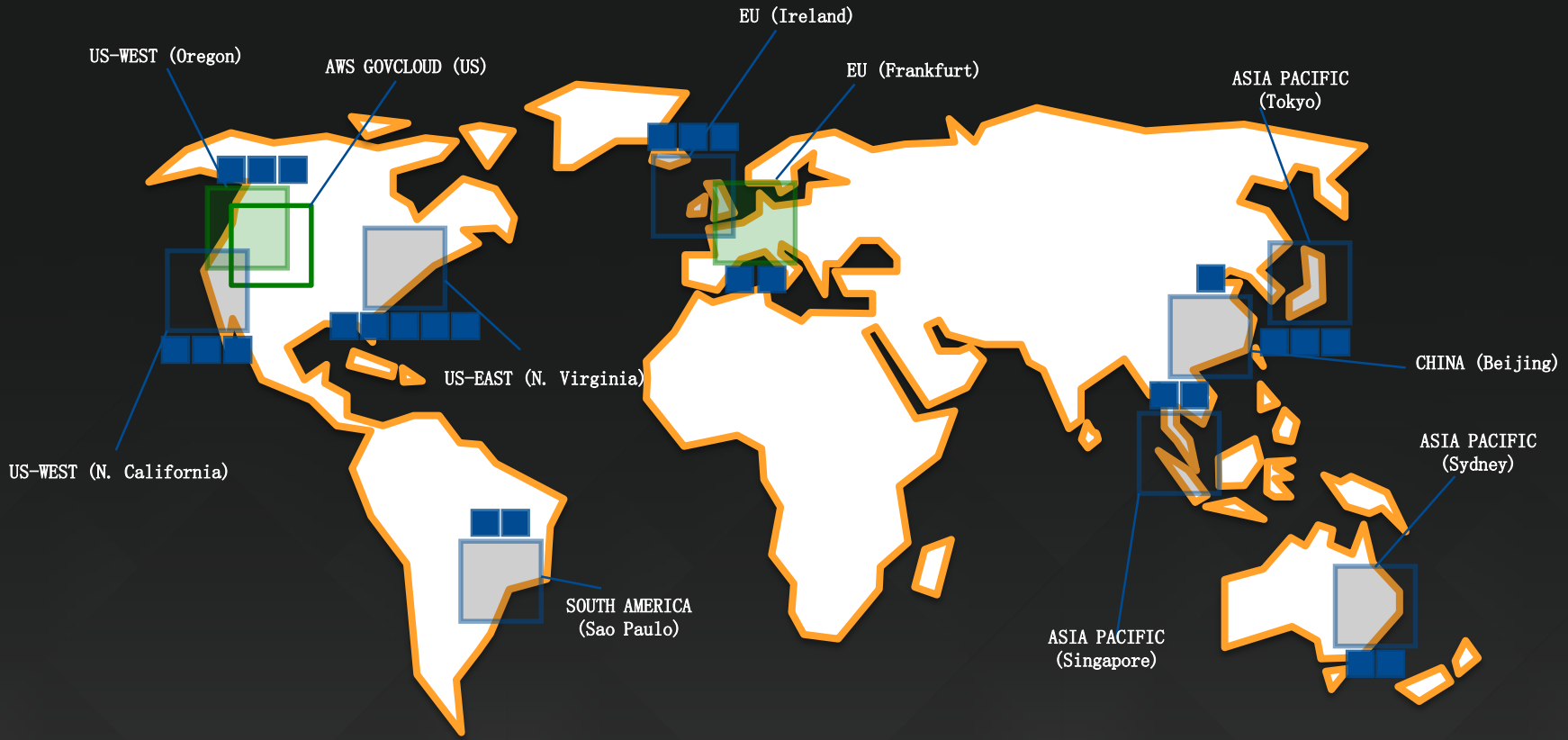
一些基础概念...

区域



amazon web services

可用区



amazon web services

节点网络

amazon web services

应用

平台服务

基础服务

全球基础设施

应用


虚拟桌面


协同与分享

平台服务

数据库
- Relational
- No SQL
- Caching

分析
- Hadoop
- Real-time
- Data Warehouse
- Data Workflows

应用服务
- Queuing
- Orchestration
- App Streaming
- Transcoding
- Email
- Search

部署与管理
- Containers
- Managed User Directories
- Dev/ops Tools
- Resource Templates
- Usage Tracking
- Monitoring and Logs

移动服务
- Identity
- Sync
- Mobile Analytics
- Notifications

基础服务


计算
(VMs, Auto Scaling and Load Balancing)


存储
(Object, Block and Archive)


安全与存取控制


网络

基础设施


区域


可用区


CDN 与 PoP

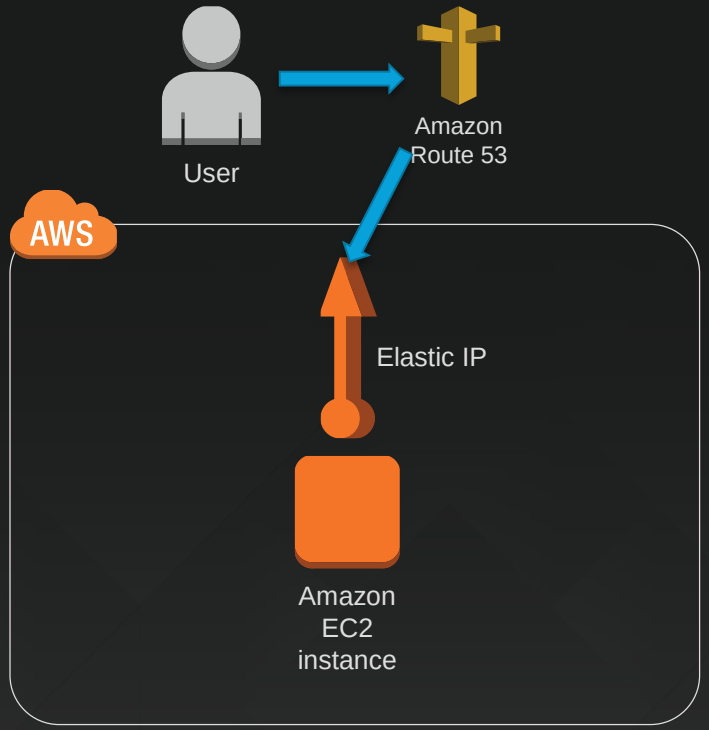
amazon web services

现在让我们就从
一个用户的第一天

开始起步

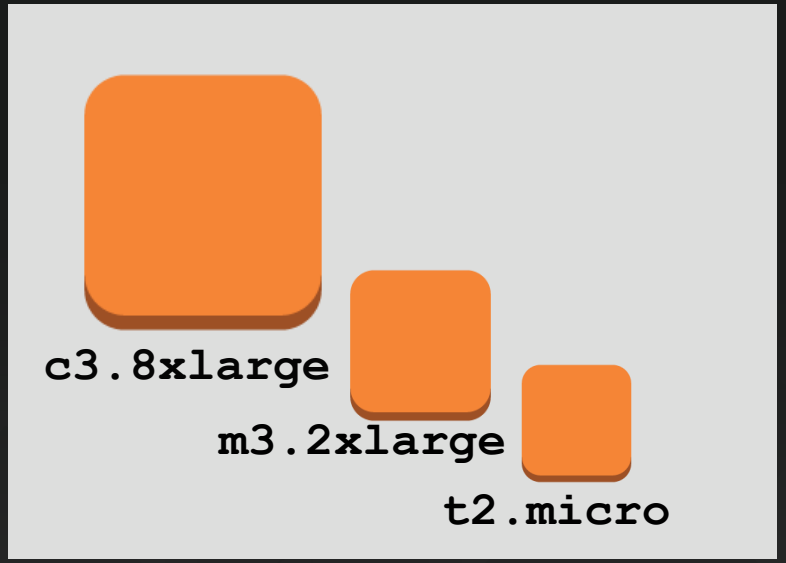
第1天，1个用户

- 1个Amazon EC2 实例
  - 全栈服务器
    - Web app
    - Database
    - Management
    - …
- 1个Elastic IP
- Amazon Route 53 用于DNS



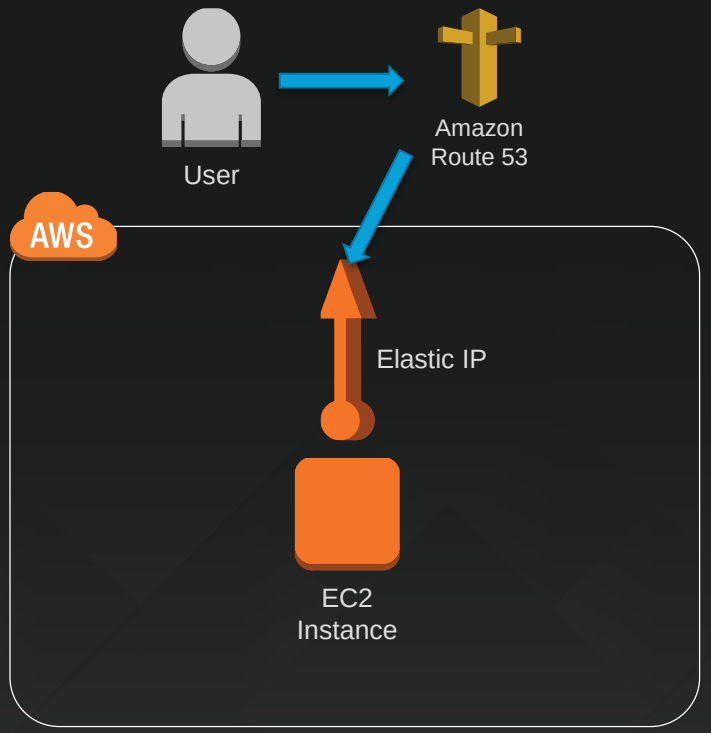
“我们需要更强大的服务器”

- 简单的方法
- 使用预配置 IOPS
- 高 I/O 实例
- 内存优化实例
- CPU 优化实例
- 存储优化实例
- 轻松地改变实例的类型与规格
- **满足最终的需要**



第1天，1个用户

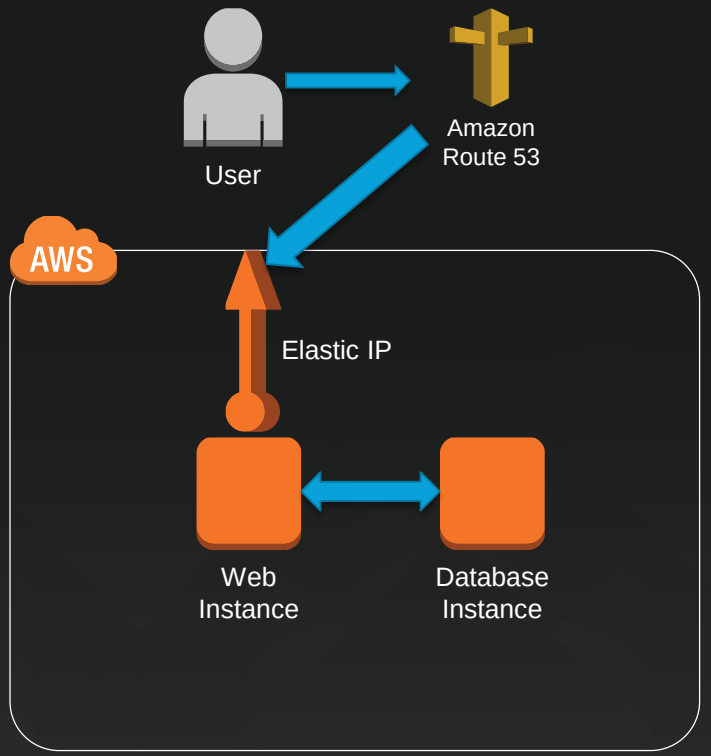
- 我们可能支持几百个到几千个用户，基于应用的复杂性与网络流量
- **没有故障转移**
- **没有冗余**
- **太多的鸡蛋在一个篮子里**



第2天，用户数>1

首先，让我们将单个服务器分成多个

- Web
- 数据库
  - 确定需要使用数据库服务？



数据库服务的选择

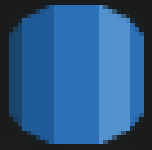
自管理

数据库运行于 Amazon EC2

需要自行管理

解决软件的许可证问题 (BYOL)

托管的服务



Amazon RDS

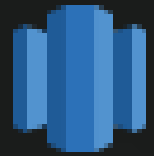
Microsoft SQL Server, Oracle, MySQL, PostgreSQL，Amazon Aurora

灵活的许可证管理模式

Amazon DynamoDB

托管的NoSQL 服务
使用SSD 存储

无缝扩展
零管理



Amazon Redshift

大规模并行的PB级别的数据仓库服务

快速，强大
以及易于扩展

amazon web services

如何选择所需要的数据库技术？
SQL？ NoSQL？

为什么通常从SQL开始?

- 基于成熟的技术
- 大量的资源，代码、社区、图书、经验、工具等
- 清晰的可扩展性模式
- 打算在你拥有1,000万用户的时候还在依赖SQL?

*除非你所做的事情过于特殊，你总会在你的架构中找到SQL适合的地方

假设数据规模在几个TB(>5)，或者是数据密集型的工作负载，那么你应该考虑NoSQL！

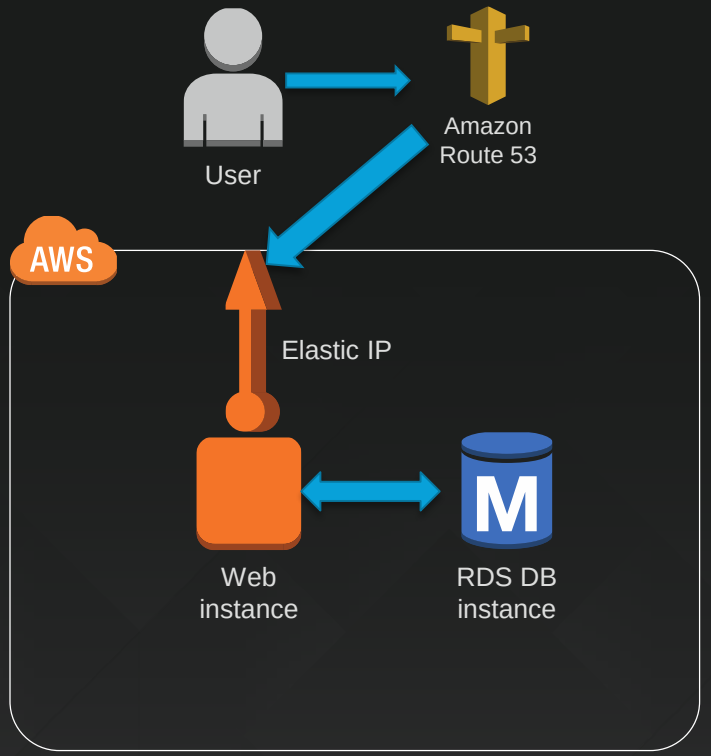
为什么需要 NoSQL?

- 实现“非常”低延迟的应用
- 元数据驱动的数据集合
- 高度的非关系数据
- 需要无模式的数据结构
- 大规模的数据 (在TB 这个级别)
- 快速的数据采集 (数千条纪录/秒)

用户规模 > 100

首先，让我们将单个服务器分拆成多个:

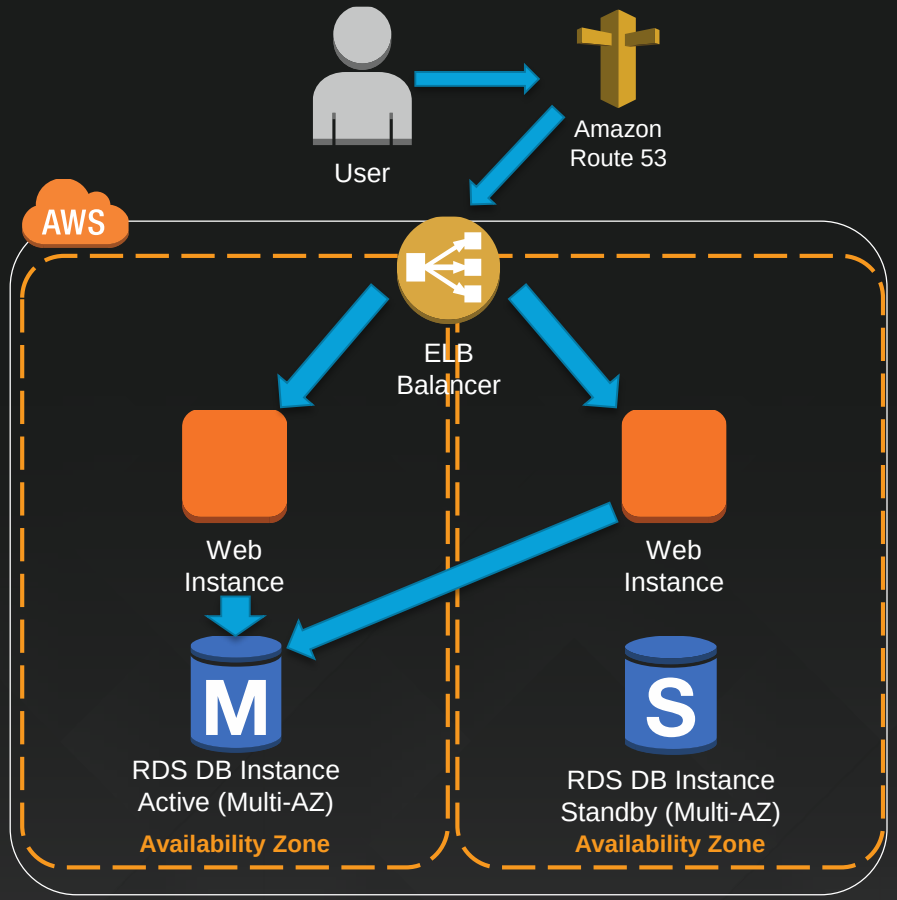
- Web
- 数据库
  - 使用 Amazon RDS 让你更轻松一点



用户规模 > 1,000

接下来，让我们解决缺少故障转移和冗余的问题:

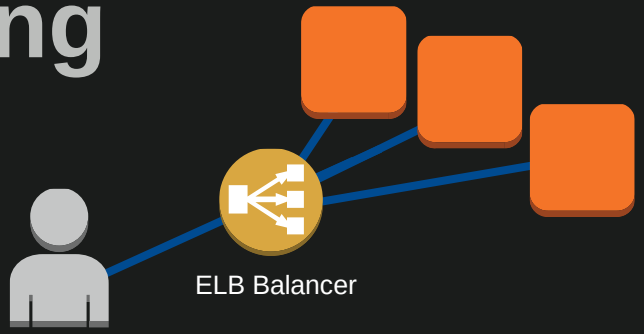
- Elastic Load Balancing (ELB)
- 另外的 Web 实例
  - 部署在另外的可用区
- RDS 多可用区(AZ)



amazon web services

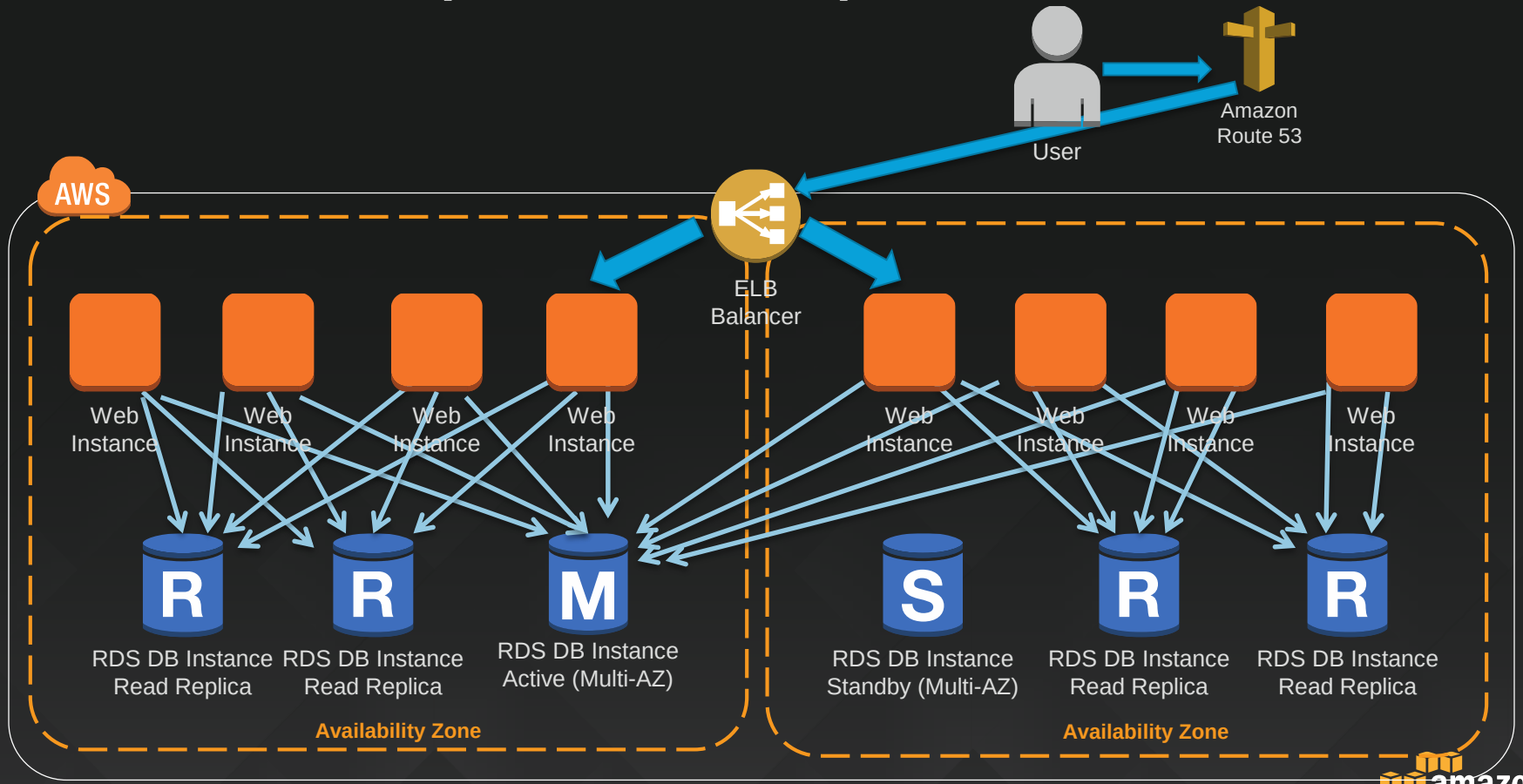
Elastic Load Balancing

- 创建高度可扩展的应用
- 负载可分布在多个可用区的EC2 实例之上



特性	说明
可用性	跨多个可用区的实例上的负载均衡
健康检查	自动检查实例的健康状况，启动或者关闭服务
会话的粘性	请求路由到同一个实例
SSL	灵活的加密支持，支持SSL从Web和应用服务器卸载
监控	为Amazon CloudWatch 提供检测数据，得到请求处理日志

用户规模 > 10,000s–100,000s

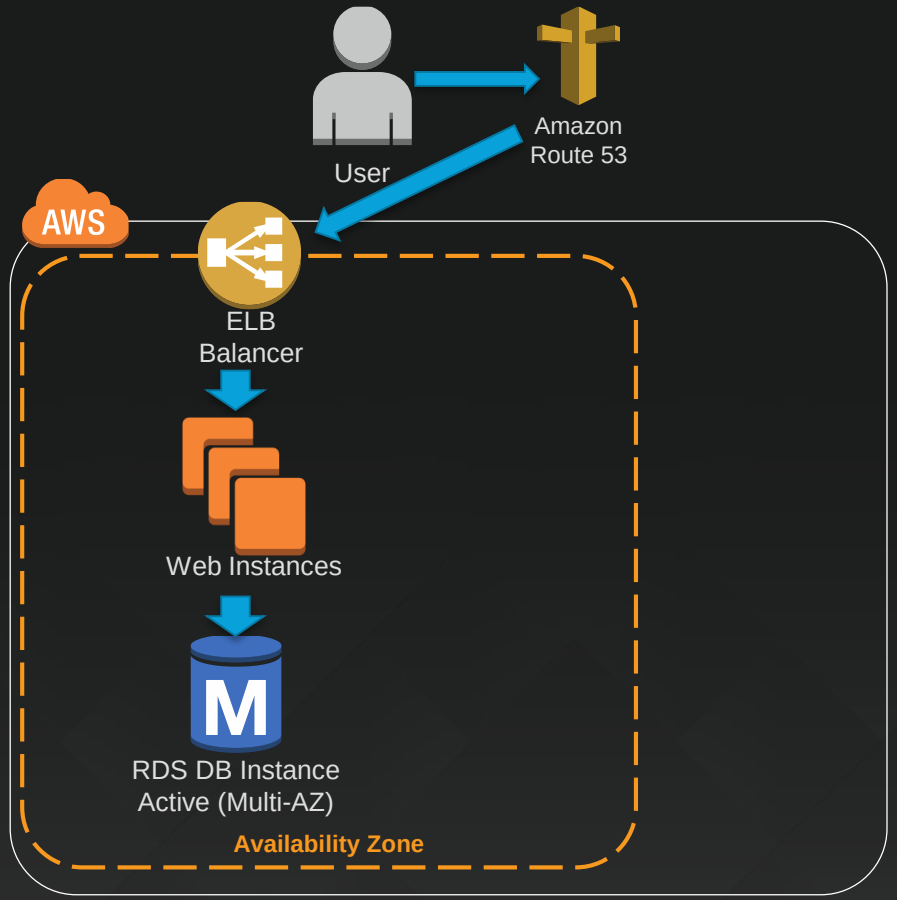


也许目前的水平已经不错，但我们可以继续关注“*性能*”和“*效率*”，*我们还可以优化的更好*

负载转移

让我们来减轻Web和数据库的负载:

- 将静态内容从Web实例转移到 Amazon S3 和 Amazon CloudFront
- 将会话/状态 和DB缓存转移到 Amazon ElastiCache或者 Amazon DynamoDB



Amazon S3

Amazon S3 是面向互联网的云存储:

- 基于对象的存储
- 11个9 的耐久性
- 适合以下场景:
  - 静态资产 (CSS, JS, 图片, 视频)
  - 备份
  - 日志
  - 待处理的文件
- 无限的扩展能力
- 对象的尺寸高达 5 TB

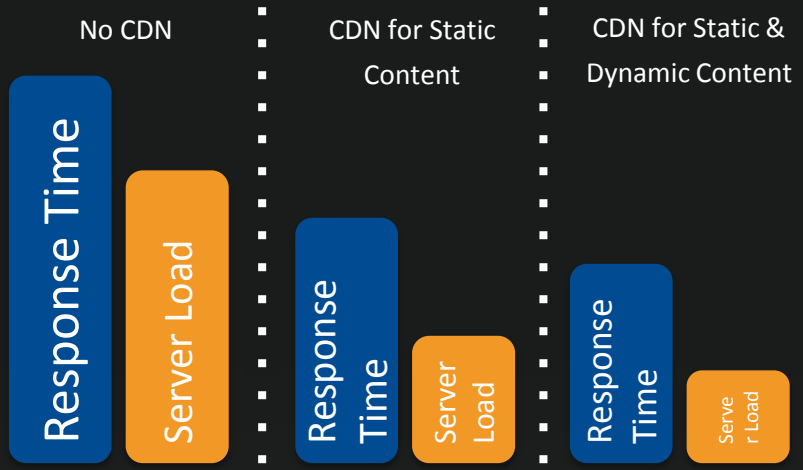
Amazon S3

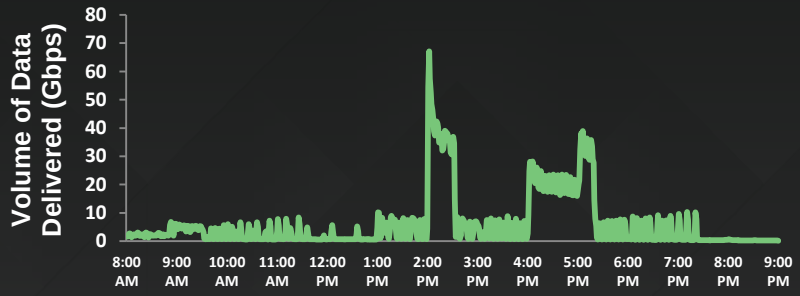
- 用于托管静态网站
- 支持细粒度的权限控制
- 与 Amazon CloudFront 的结合
- Amazon EMR 的集成
- 充当 S3, CloudFront, Billing, ELB, AWS CloudTrail 等的日志端节点
- 支持静态以及传输中加密
- 仅仅是冗余存储成本的1/3
- Amazon Glacier f用于超长时间的存储，成本是 S3 的 1/3

Amazon CloudFront

Amazon CloudFront 是可扩展的内容分发网络服务:

- 在边缘节点缓存静态内容以实现更快的交付
- 有助于降低基础设施的负荷
- 动态和静态的内容
- 流媒体视频
- 根域名(Zone apex) 支持
- 自定义SSL 证书
- 低的 TTLs (短至0秒)
- 降低获得源数据的成本 （Amazon S3 / Amazon EC2 和 Amazon CloudFront)
- 与Amazon EC2, Amazon S3, Elastic Load Balancing 和 Amazon Route 53 的协同优化



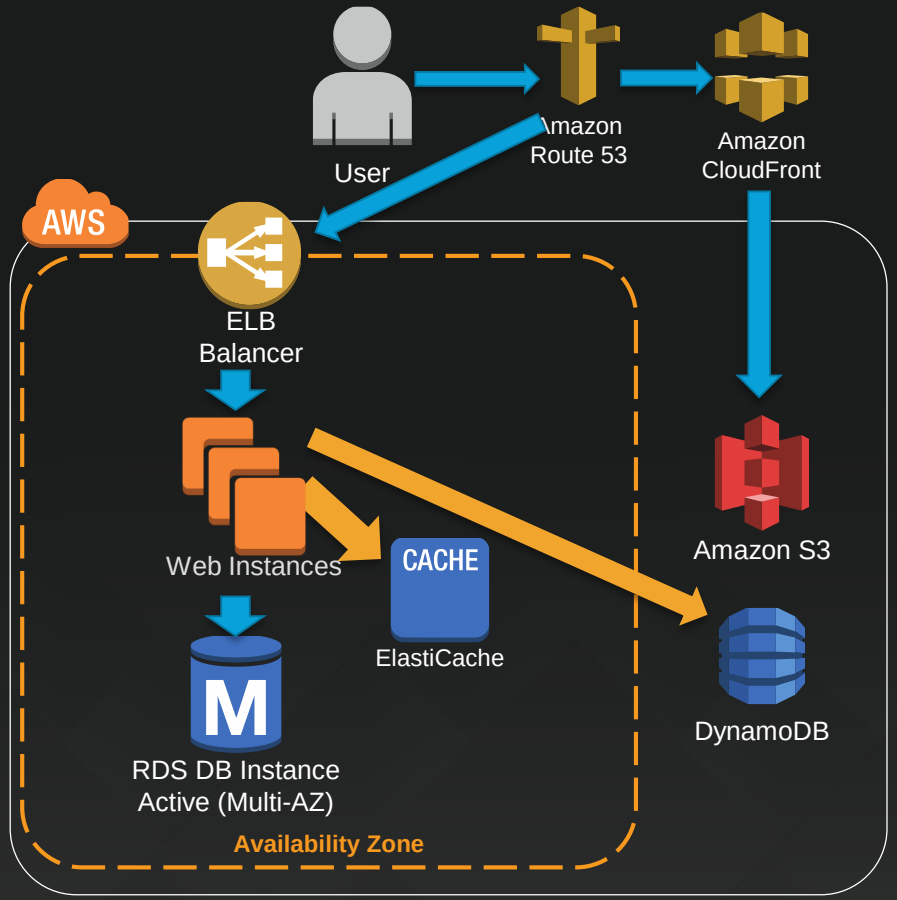


amazon web services

负载转移

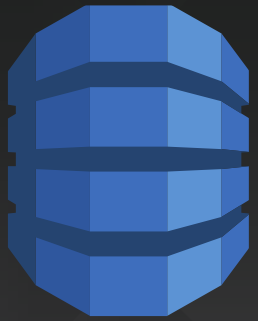
让我们来减轻Web和数据库的负载:

- 将静态内容从Web实例转移到 Amazon S3 和 Amazon CloudFront
- **将会话/状态 和DB 缓存转移到 Amazon ElastiCache 或者 Amazon DynamoDB**



Amazon DynamoDB

- 托管的、吞吐量可调整的 NoSQL 数据库
- 快速、可预测的性能
- 全分布式、容错体系结构
- JSON 支持 (新特性)
- 项目支持 高达400 KB (新特性)



特性	描述
预分配吞吐量	向上或向下调整读 / 写能力
可预测的性能	基于SSD的基础设施提供了平均个位数（毫秒）的延迟
强大的一致性	确保你读取的是最近更新过的数据
容错性	跨可用区复制数据
监控	与Amazon CloudWatch 集成
安全	与AWS Identity and Access Management (IAM) 集成
Amazon EMR	针对大规模数据的复杂分析可与Amazon EMR 结合

amazon web services

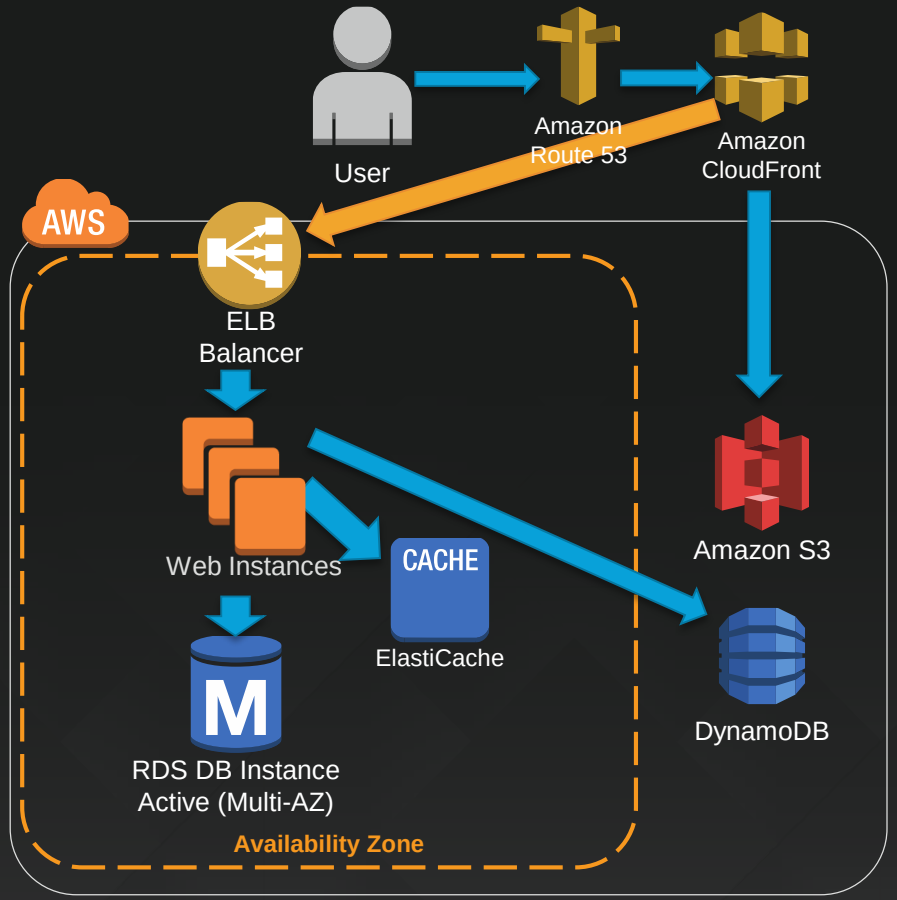
Amazon ElastiCache

- 托管的 Memcached 与 Redis
  - 与传统的开源的项目 Memcached 有 Redis 相同的API
- 从一个到多个节点的扩展
- 自我修复 (替代死掉的实例)
- 非常快 (通常个位数的毫秒级别或者更少)
- Memcache 设置在单个的AZ, 没有持久性以及复制功能
- 对于Redis 可以设置在多个AZ间复制以提供持久性
- 使用AWS Auto Discovery 客户端在不影响应用的前提下简化集群的增长和收缩

负载转移

让我们来减轻我们的网站和数据库实例的负载:

- 从Web 实例将静态内容转移到Amazon S3 和 Amazon CloudFront
- 将会话 / 状态以及 DB 缓存到 ElastiCache 或者 DynamoDB
- **将会话/状态 和DB缓存转移到 Amazon ElastiCache或者 Amazon DynamoDB**

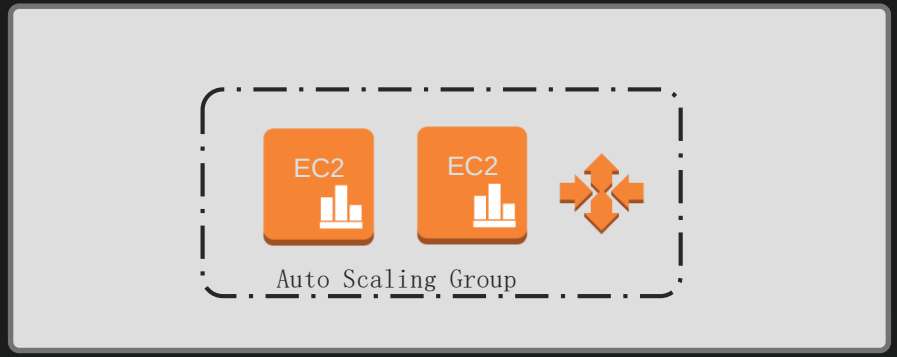


现在我们的 Web 层已经变得更轻巧， 我们可以从新审视我们开始的话题...

自动扩展!

Auto Scaling

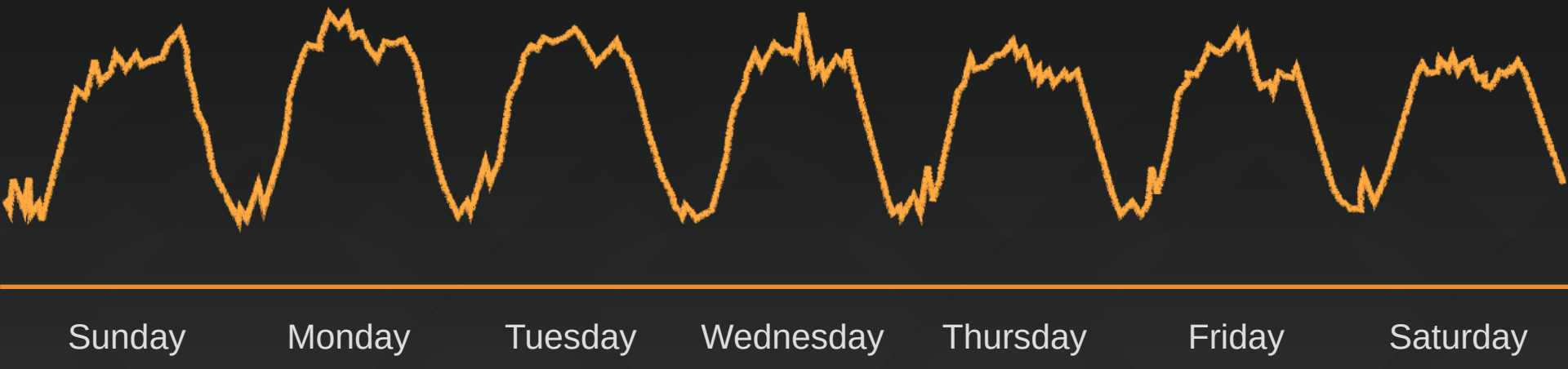
计算集群自动的按需调整



特性	说明
控制	设定最小和最大实例池的尺寸
与Amazon CloudWatch 的集成	利用CloudWatch 驱动 扩展
实例类型	对于按需实例和竞价实例运行Run Auto Scaling。与VPV兼容

aws autoscaling create-auto-scaling-group
--auto-scaling-group-name **MyGroup**
--launch-configuration-name **MyConfig**
--min-size **4**
--max-size **200**
--availability-zones **us-west-2c, us-west-2b**

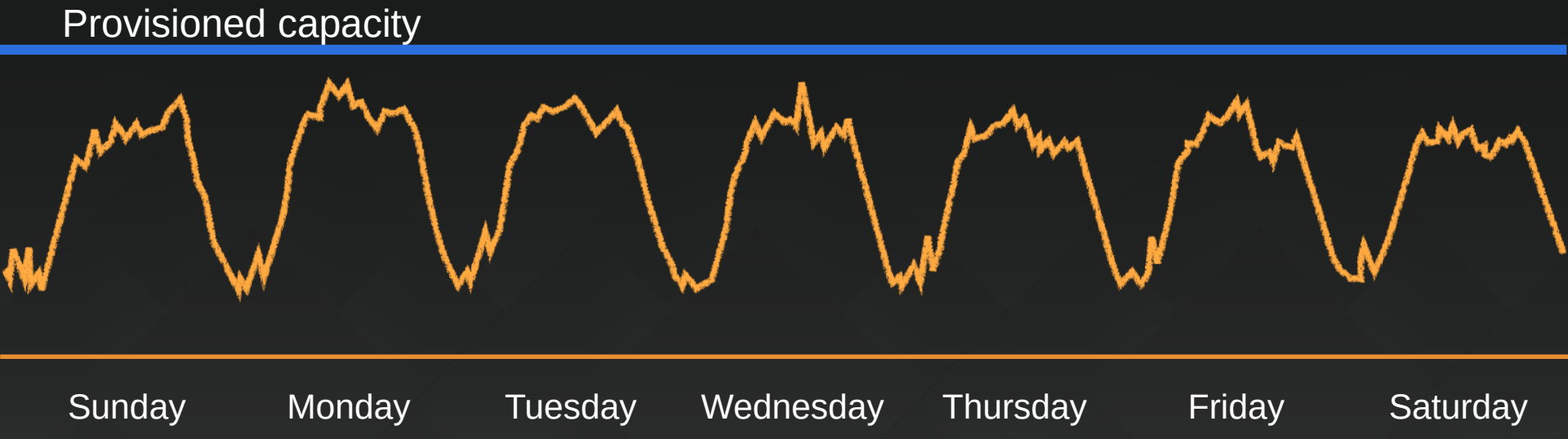
典型的 Amazon.com 一周的流量



Sunday Monday Tuesday Wednesday Thursday Friday Saturday

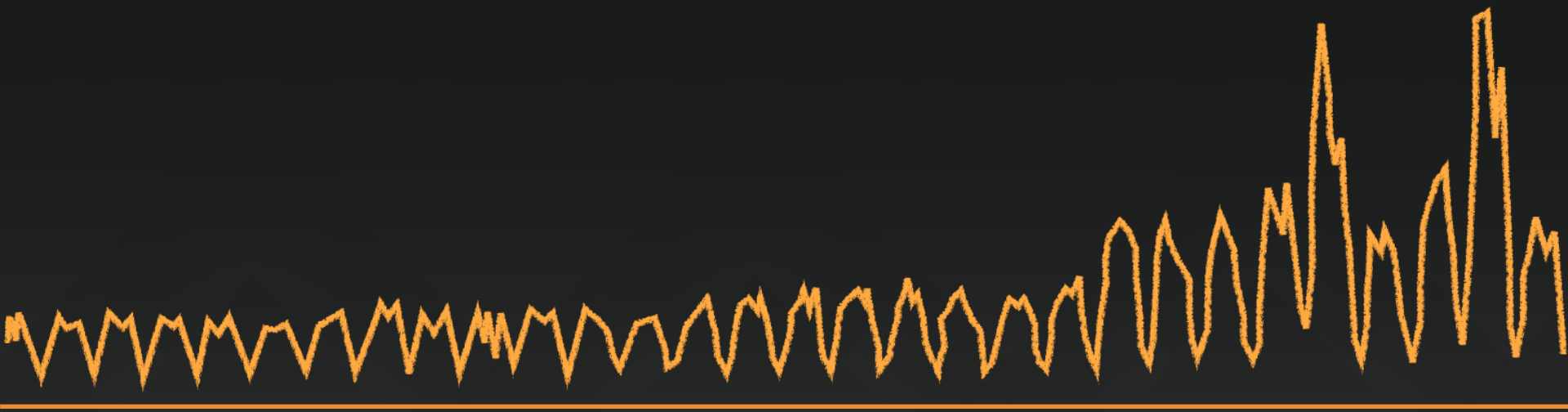
amazon web services

典型的 Amazon.com 一周的流量



amazon web services

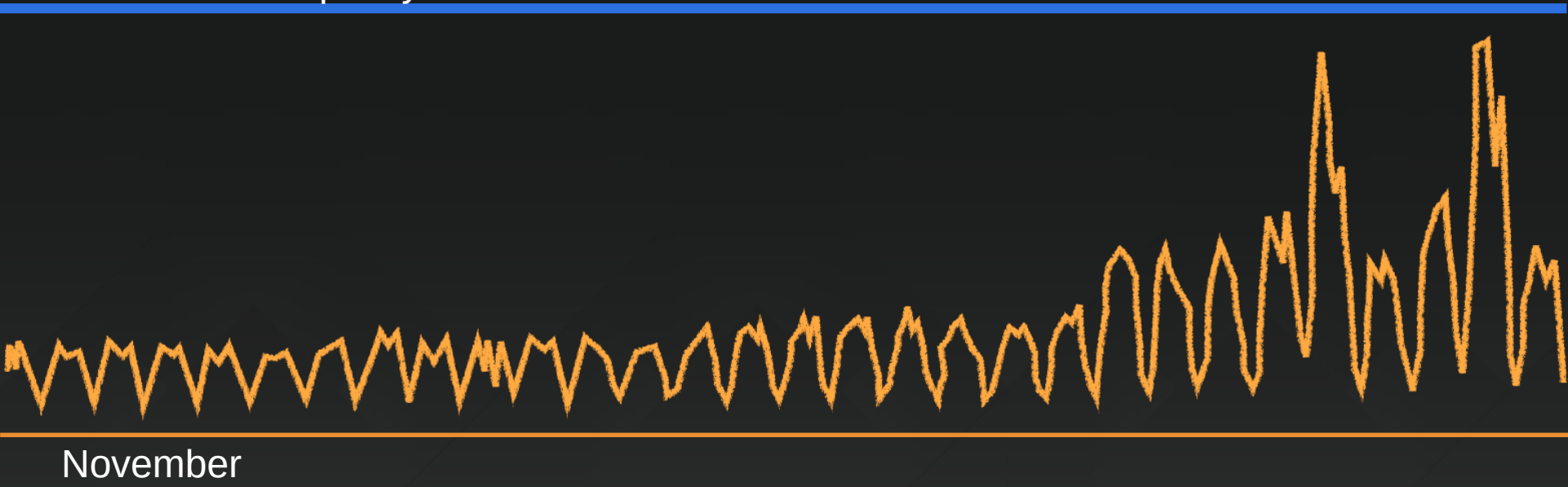
Amazon.com 11月份的流量



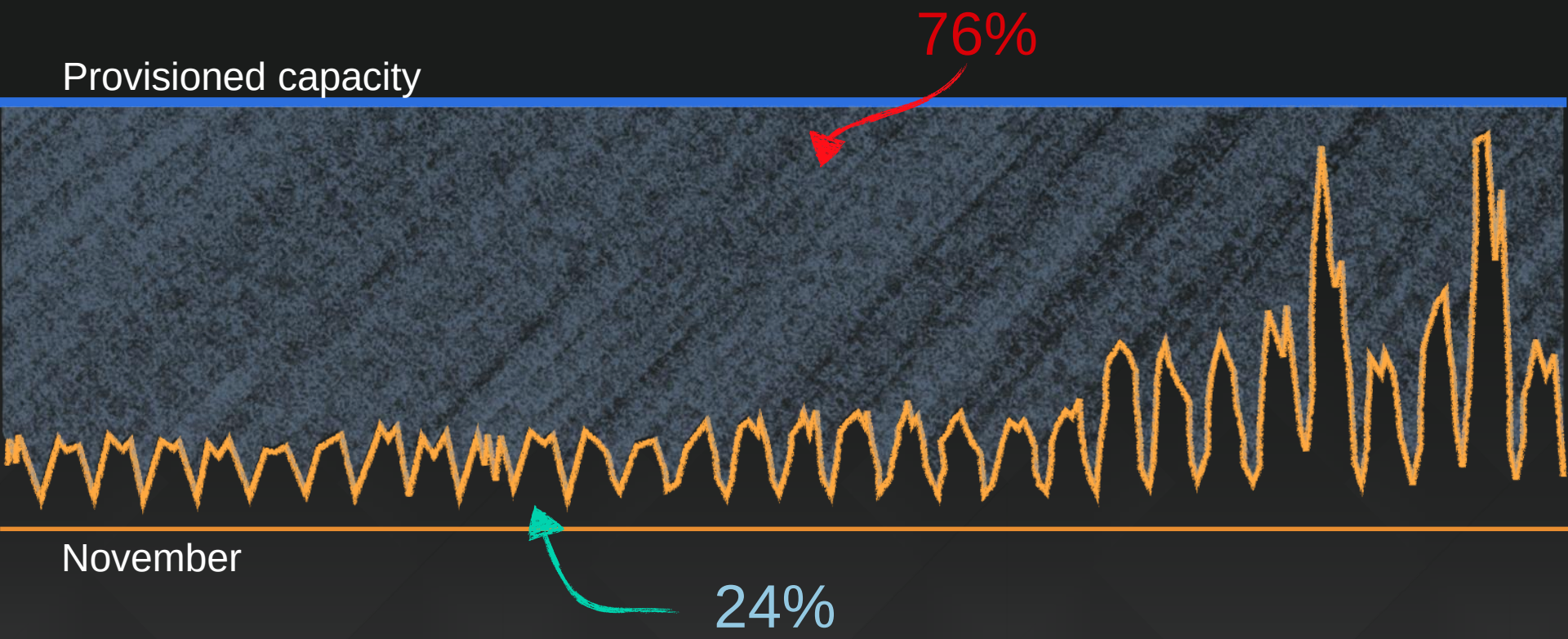
November

amazon web services

Amazon.com 11月份的流量

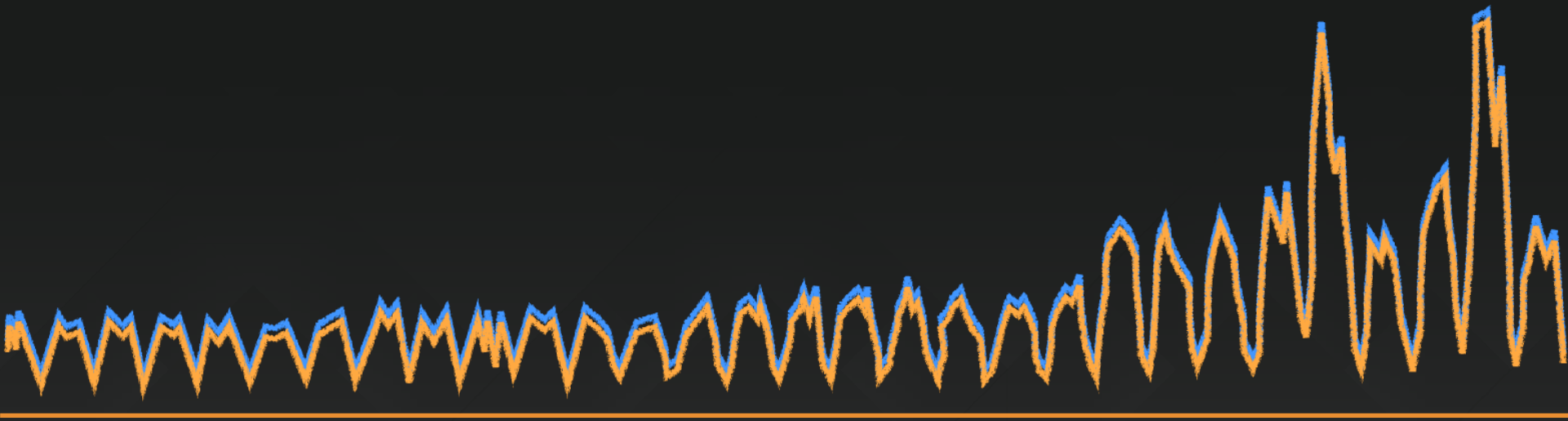


Amazon.com 11月份的流量



amazon web services

Amazon.com 11月份的流量

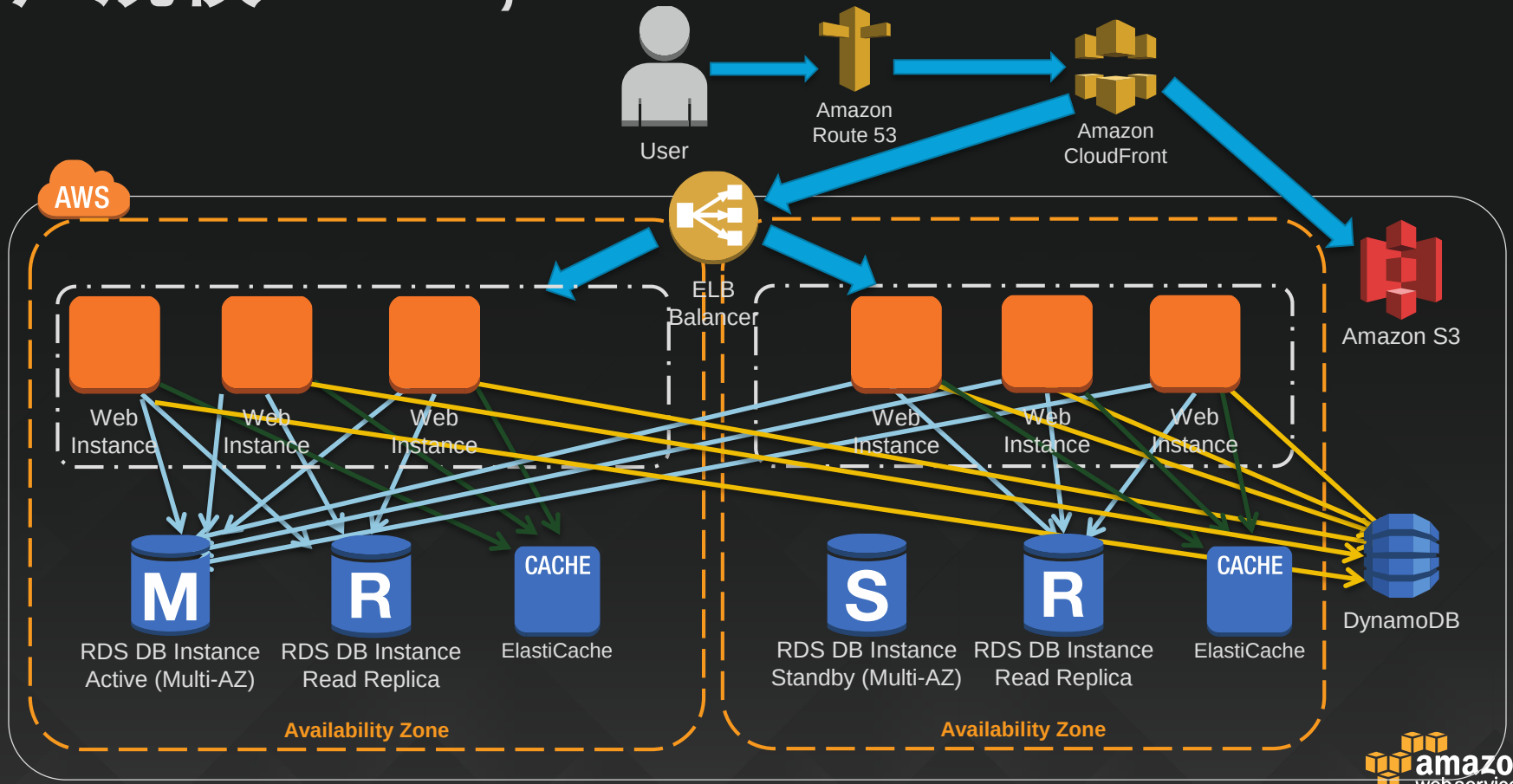


November

amazon web services

Auto Scaling
让你可以做到这一切!

用户规模 > 500,000+



amazon web services

使用自动化部署

基础架构的管理已经成为我们的一项重要的工作。 可以用工具来自动执行重复性的任务:

- 用工具管理 AWS 资源
- 用工具管理实例之上的软件和配置
- 针对日志和用户行为进行自动化的数据分析

AWS 应用管理解决方案

高级服务

DIY

AWS
Elastic Beanstalk

AWS
OpsWorks

AWS
CloudFormation

Amazon EC2

便利

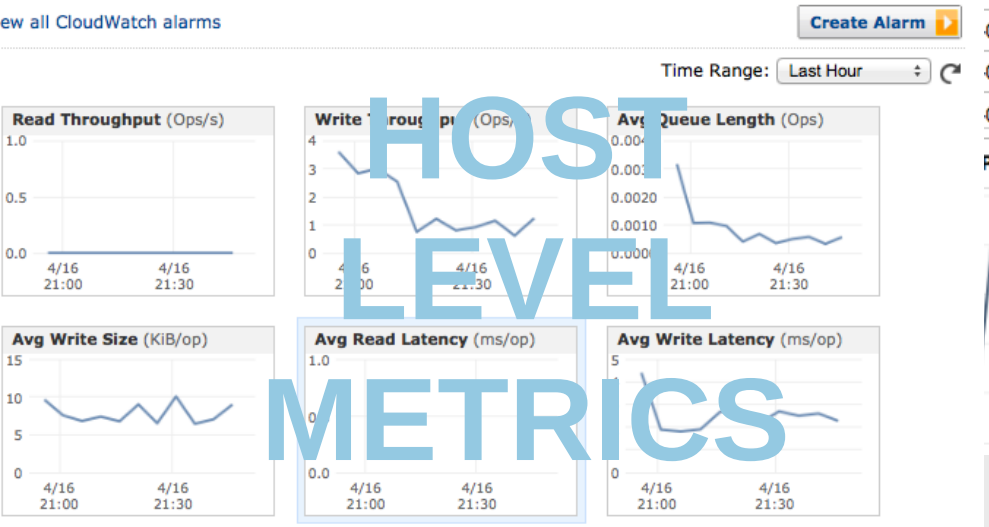
控制

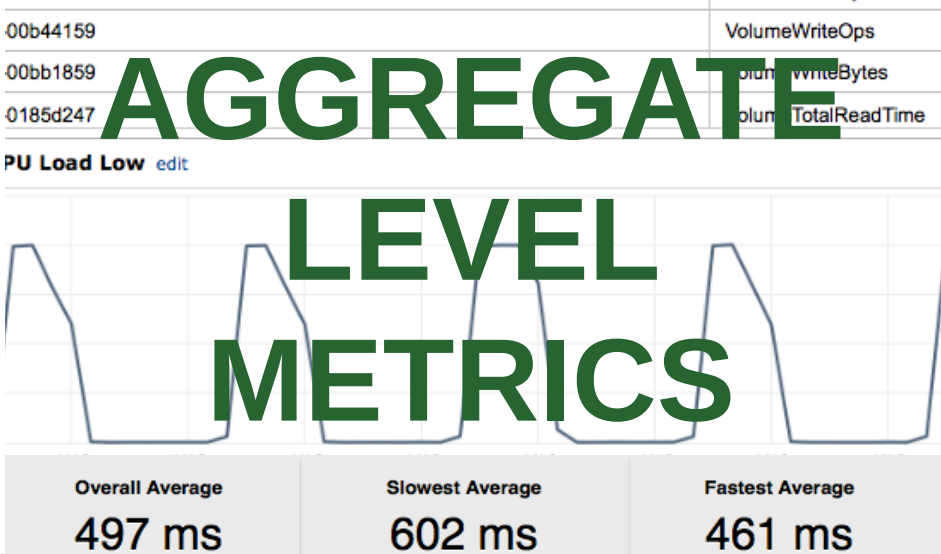
amazon web services

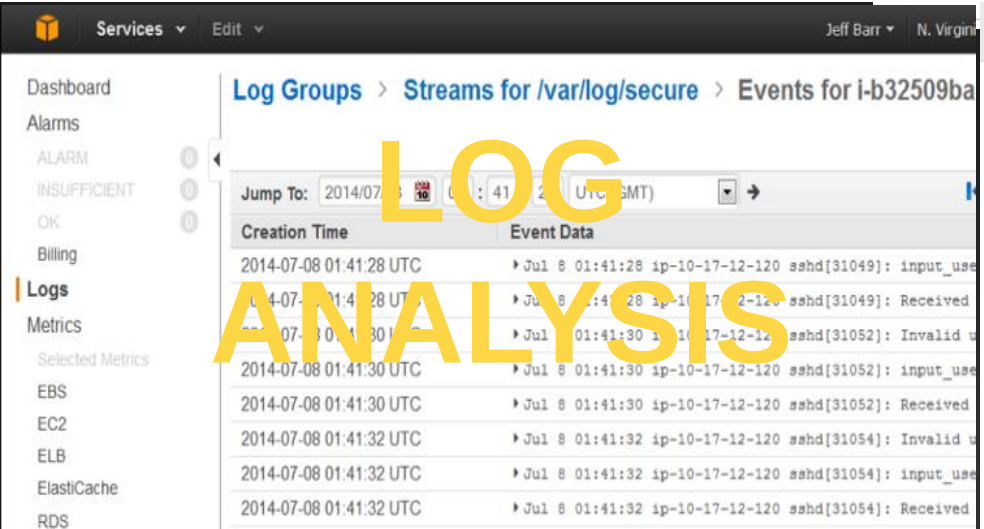
用户规模> 500,000+

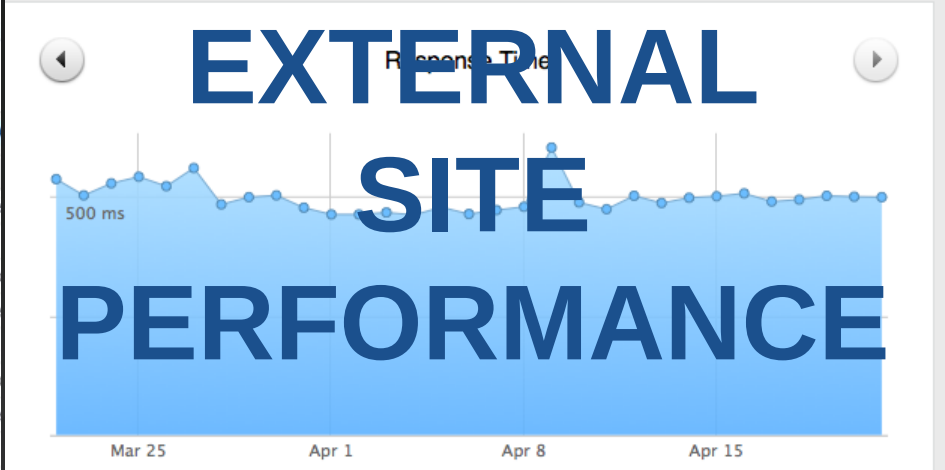
你可能开始遇到一些新的问题，关于应用的速度和性能:

- 确保在每个地方都有监控、指标和日志
- 如果不能在内部建立它就外包出去!
- 了解你的客户对什么感到满意 / 不满意
- 对每一个服务 / 组件尽可能多的榨取更多的性能


ew all CloudWatch alarms
Create Alarm
Time Range: Last Hour
Read Throughput (Ops/s)
Avg Queue Length (Ops)
Avg Write Size (KiB/op)
Avg Read Latency (ms/op)
Avg Write Latency (ms/op)
4/16 21:00
4/16 21:30
HOST LEVEL METRICS


00b44159
VolumeWriteOps
00bb1859
TotalReadTime
0185d247
AGGREGATE LEVEL METRICS
PU Load Low edit
Overall Average
497 ms
Slowest Average
602 ms
Fastest Average
461 ms


Services
Edit
Jeff Barr
Dashboard
Alarms
Billing
Logs
Metrics
EBS
EC2
ELB
ElastiCache
RDS
Log Groups
Streams for /var/log/secure
Events for i-b32509ba
Jump To:
Creation Time
Event Data
2014-07-08 01:41:28 UTC
2014-07-08 01:41:30 UTC
2014-07-08 01:41:32 UTC
LOG ANALYSIS


EXTERNAL SITE PERFORMANCE
500 ms
Mar 25
Apr 1
Apr 8
Apr 15

进一步的提升，划分成不同的 Web/App 层

SOA…这意味着什么?

amazon web services

SOAing

- 将服务安排到专属的层 / 模块. 在架构中将其视为100% 独立的部分, 并独立地去扩展它们。

- Amazon.com 和 AWS 广泛的应用了这个原则! 它提供了灵活性以及对每个组件的更好的了解。

松耦合 + SOA = 成功

在起步阶段,如果你需要的服务已经存在，建议使用它们而不是自己建立一套

不要重新发明轮子

例如:

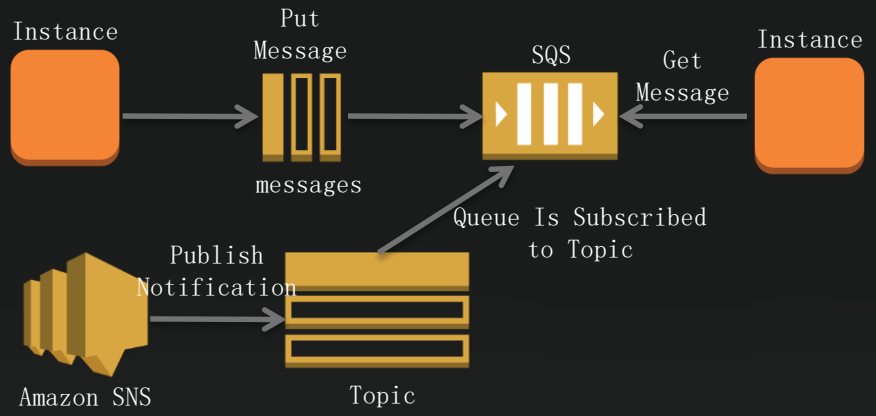
- Email
- Queuing
- Transcoding
- Search
- Databases
- Monitoring
- Metrics
- Logging
- Compute

amazon web services

面向应用地服务

Amazon SQS

用于在实例间存储消息的、可靠的、高扩展性的队列服务



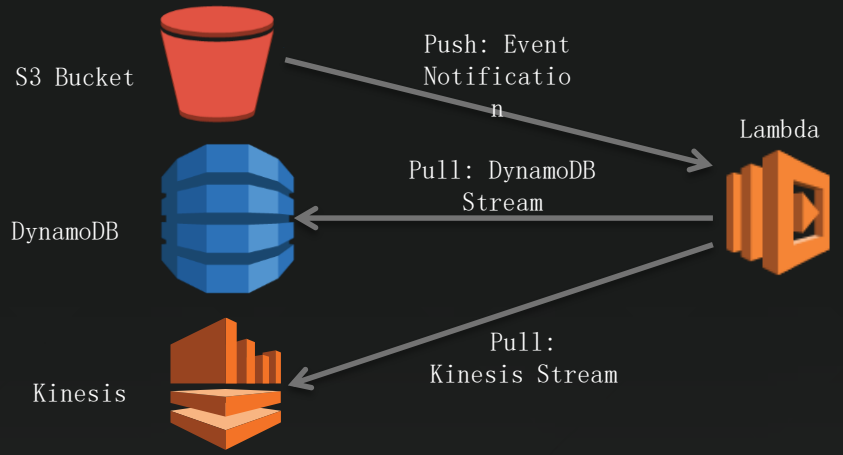
- 应用服务
- 平台服务
- 功能服务
- AWS 全球基础架构

特性	说明
可靠	消息冗余存储在多个可用区
简单	简单的APIs 用来发送和接受消息
扩展性	消息的数量没有限制
安全	队列认证以确保访问受控

计算/ 平台

AWS Lambda

事件驱动的计算，在AWS 服务间进行连接



特性	说明
无状态	由事件触发的请求驱动调用的代码lambda函数
容易	固定的开发语言JavaScript
管理	AWS 拥有并管理基础设施
扩展	隐式的扩展；只需要作出请求

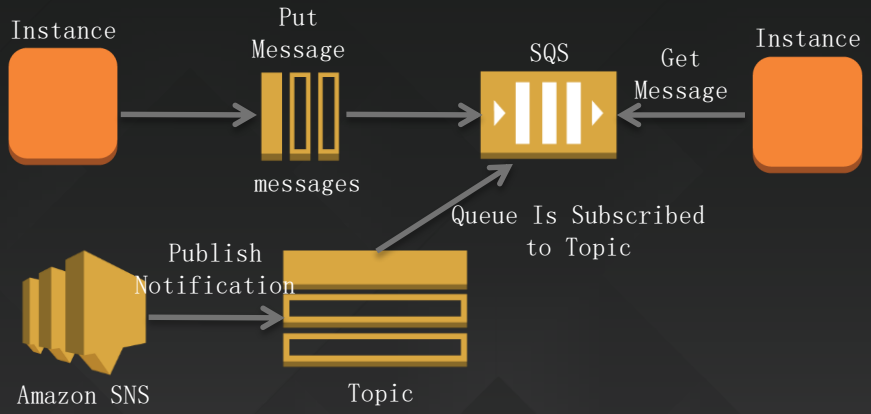
- 应用服务
- **平台服务**
- 功能服务
- AWS 全球基础架构

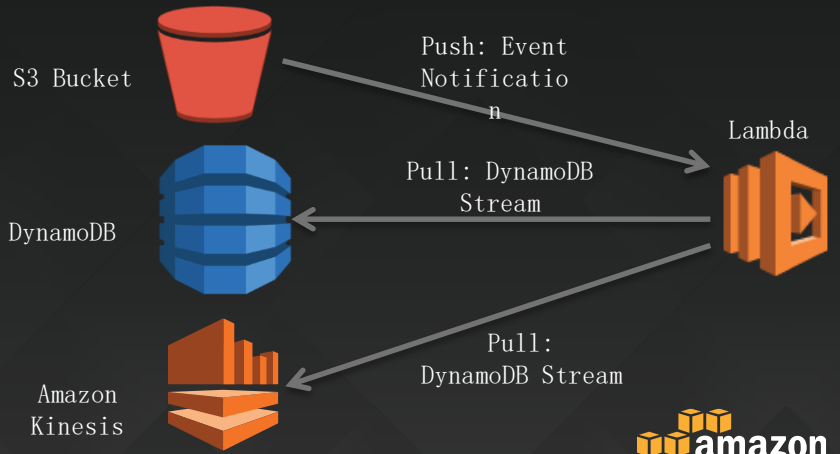
amazon web services

松耦合让你得到自由！

更松散的耦合，更好的扩展

- 独立的组件
- 为黑盒设计
- 交互去耦
- 服务内置冗余和扩展性，而不是复制服务





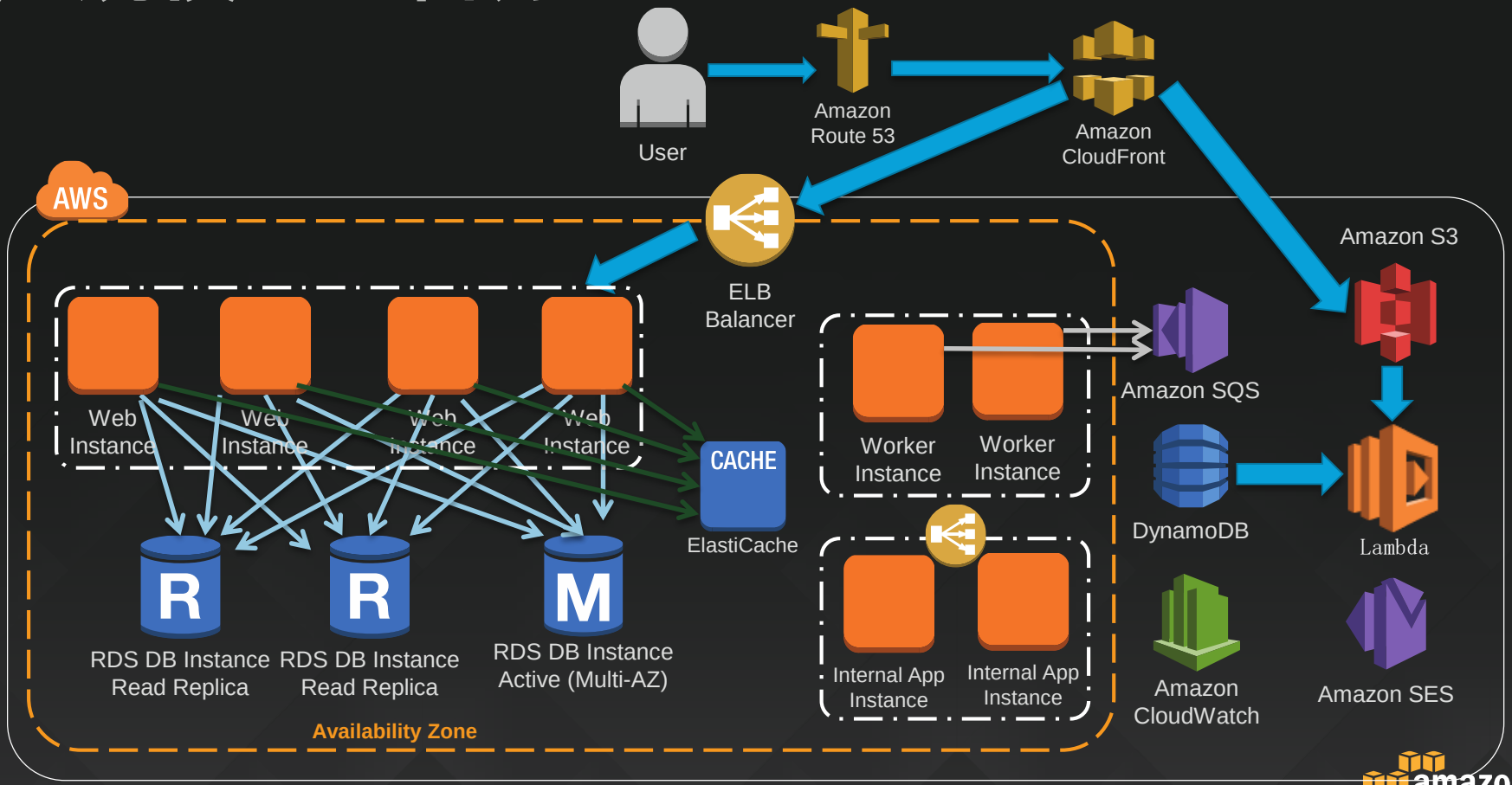
amazon web services

用户规模> 1 百万+

在这个规模上，需要之前谈过的一切以及…

- 多-AZ
- 在各层间使用 Elastic Load Balancing
- 自动扩展
- SOA
- 更聪明的内容服务 (Amazon S3/CloudFront)
- 为数据库缓存
- 会话无状态以利于扩展

用户规模 > 1 百万 +



amazon web services

下一步

用户规模 > 5 百万– 1 千万

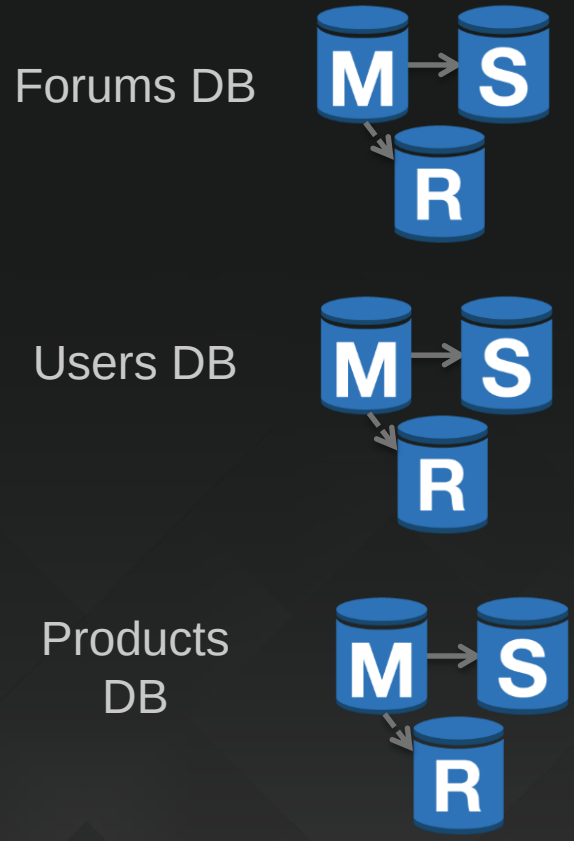
你可能遇到主数据库上“写”的问题

如何解决?

- 联合— 基于功能划分成多个数据库
- 分片— 将一个数据集分布到多个主机上
- 将部分功能转移到其它类型的数据库上 (NoSQL, Graph)

数据库联合(Database federation)

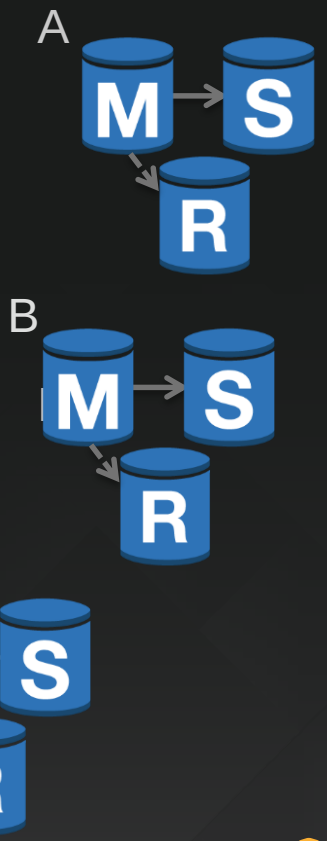
- 对数据库按照功能 / 目的进行分割
- 难于实现跨功能的查询
- 本质上延缓了诸如分片/NoSQL一类的需求
- 无助于全功能或者全表的实现



分片实现的横向扩展

- 应用层变得更复杂
- 能够支持ORM
- 伸缩性没有限制
- 操作上更精巧，更复
- 可按功能 / 键值分区
- RDBMS 或者 NoSQL

User	ShardID
002345	A
002346	B
002347	C
002348	B
002349	A



向 NoSQL 迁移

- 感觉上类似与联合(federation)
- 想一下前面提到的 NoSQL vs SQL
- 使用托管的服务，例如： DynamoDB
- 一些使用场景:
  - 排行榜/得分
  - 快速摄入的点击流 / 日志
  - 临时数据 (购物车)
  - “热” 表
  - 元数据/对照表

DynamoDB

快速总结

快速总结

- 基础架构中的多-AZ
- 使用自扩展的服务—ELB, Amazon S3, Amazon SNS, Amazon SQS, Amazon SWF, Amazon SES 等
- 在各个层面建立冗余
- 从SQL 起步，要仔细规划
- 在基础架构的内外要缓存数据
- 在基础架构中使用自动化的工具

快速总结（续）

- 确保有完善的指标 / 监控 / 日志工具
- 将各个层划分成独立的服务 (SOA)
- 一旦就绪，使用自动扩展
- 不要重复发明轮子！
- 感觉有必要迁移到NoSQL

用到了这一切，就意味着我们现在应该很容易能够处理1,000万用户！

以至无穷...

用户规模 > 1,000 万

- 对应用更好的优化（fine-tuning ）
- 更多的SOA 特性 / 功能
- 从多-AZ 到多区域
- 可能着手开发定制方案
- 对整个技术栈的深入分析

下一步?

读这些!

- aws.amazon.com/documentation
- aws.amazon.com/architecture
- aws.amazon.com/start-ups

开始体验AWS

- aws.amazon.com/free/

谢谢！