**Encapsulation 封装**

Bittorrent比特流

BitTorrent是一款新概念P2P的下载工具。它采用了多点对多点的原理，一般下载文件或软件，大都由HTTP站点或FTP站台下载，若同时间下载人数多时，基于该服务器带宽的因素，速度会减慢许多。而该软件却不同，恰巧相反，同时间下载的人数越多你下载的速度便越快，因为它采用了多点对多点的传输原理。

API（Application Programming Interface，应用程序接口

Exponential 指数级

Bidirectional 双向的

Canonical 经典的

Symbolic象征的

failover 故障转移切换

Syntax 语法

Concurrency 并发 parallelism 并行

云服务器 (Elastic Compute Service, 简称 ECS)

bootstrapping 自举法

访问控制列表(ACL)是一种基于包过滤的访问控制技术，它可以根据设定的条件对接口上的数据包进行过滤，允许其通过或丢弃。访问控制列表被广泛地应用于路由器和三层交换机，借助于访问控制列表，可以有效地控制用户对网络的访问，从而最大程度地保障网络安全。

Glossary 词汇表

Sync synchronize 同步

Asynchronous 异步

RPC是远程过程调用（Remote Procedure Call）的缩写形式

traverse 横过nv

Transaction 处理 n

Deploy 部署，调度v

a quorum of nodes 一群节点

primitives 原语

semantics 语义学

TTL time to live生存时间

scalability 可扩展性

archived 存档、归档

Permutation 排列、组合

Anagram 同字母异序词

CLI command-line interface

Intervene 干扰

微服务 是一种云原生架构方法，其中单个应用程序由许多松散耦合且可独立部署的较小组件或服务组成

有自己的堆栈，包括数据库和数据模型；

通过REST API，事件流和消息代理的组合相互通信；

它们是按业务能力组织的，分隔服务的线通常称为有界上下文。

Alias 别名

Spec 规格

rest api 是前后端分离最佳实践，是开发的一套标准Representational State Transfer 客户端通过四个HTTP动词，对服务器端资源进行操作，实现"表现层状态转化"。

**Retrieval 数据检索**

**CDN content delivery network 内容分发网络**

HTTP PULL Resource

网站管理者将内容存放于服务器，透过设定DNS 的CNAME(别名记录) 功能，让使用者在浏览网页时，直接指向多个CDN节点。CDN节点会在该区第一次浏览时，首先转到来源服务器抓取资源，同时CDN主机会将这个档案暂存留下直到到期为止。所以下次同区的使用者要求浏览网页时，则不需要再重复该步骤。HTTP PULL 的优势为设定简单，只要当DNS CNAME设定完成，HTTP PULL功能即生效。

然而 HTTP PULL 可能有读取速度稍慢的状况，其原因有二，

其一是如果有人第一次浏览该档案；其二是在CDN缓存内容到期后，需设置合适的到期时间将会有助解决该问题。

HTTP PUSH 的工作原理类似于辅助服务器（每个区域建议使用多个DNS服务器，可以提供域名解析容错功能）网站管理者主动将需要发布的资源传递到CDN主机，再由CDN供应商分发到其各个节点，这意指网站管理者可以只上传他所指定的内容部分。

HTTP PUSH 的优点在于节省来源网站的流量，因为只有在新档案或是有档案更改需要上传时才会使用到流量。以及当流量高峰来临的时候，来源网站流量不会被占满，因为已经提前在离峰时间，将内容放到CDN节点上。然而较为不便的是，需要透过FTP的方式将内容推送到CDN的指定位置。

Pull多为静态网页内容，无频繁的档案下载服务。最合适小档案的传输，连在该节点第一次有使用者浏览网页时，都几乎感觉不到差异。当开发变化时，你可以在笔记本上运行你的监控如果目标实例挂掉，你可以很容易地知道。你可以手动指定一个目标，并通过浏览器检查该目标实例的监控状况

PUSH:多为提供档案下载服务。较合适有大档案的传输，像是软体安装、或是档案一般超过50MB。

orchestrates 策划

Leverage n 影响力、杠杆作用 v 利用

Yml 配置文件（json麻烦，发明了yml语言）

Repository 仓库

identical = same

GC = garbage collection

Consecutive = continuous

Select

select的用法。在如下的代码中，先准备一个数组（下面代码中的fds），让fds存放着所有需要监视的socket。然后调用select，如果fds中的所有socket都没有数据，select会阻塞，直到有一个socket接收到数据，select返回，唤醒进程。用户可以遍历fds，通过FD\_ISSET判断具体哪个socket收到数据，然后做出处理。

epoll将这两个操作分开，先用epoll\_ctl维护等待队列，再调用epoll\_wait阻塞进程。如果内核维护一个“就绪列表”，引用收到数据的socket，就能避免遍历。如下图所示，计算机共有三个socket，收到数据的sock2和sock3被rdlist（就绪列表）所引用。当进程被唤醒后，只要获取rdlist的内容，就能够知道哪些socket收到数据。