**参考文档：**<https://www.jianshu.com/p/f6d7f832961d>

**一、高可用系统架构：**

* **资源隔离**：让你的系统里，某一块东西，在故障的情况下，不会耗尽系统所有的资源，比如线程资源。
* **限流**：高并发的流量涌入进来，比如说突然间一秒钟100万QPS，废掉了，10万QPS进入系统，其他90万QPS被拒绝了。
* **熔断**：系统后端的一些依赖，出了一些故障，比如说mysql挂掉了，每次请求都是报错的，熔断了，后续的请求过来直接不接收了，拒绝访问，10分钟之后再尝试去看看mysql恢复没有
* **降级**：mysql挂了，系统发现了，自动降级，从内存里存的少量数据中，去提取一些数据出来
* **运维监控**：监控+报警+优化，各种异常的情况，有问题就及时报警，优化一些系统的配置和参数，或者代码。

**二、雪崩效应**

分布式系统环境下，服务间类似依赖非常常见，一个业务调用通常依赖多个基础服务。  
比如我们现在有3个业务调用分别是查询订单、查询商品、查询用户，且这三个业务请求都是依赖第三方服务-订单服务、商品服务、用户服务。三个服务均是通过RPC调用。当查询订单服务，假如线程阻塞了，这个时候后续有大量的查询订单请求过来，那么容器中的线程数量则会持续增加直致CPU资源耗尽到100%，整个服务对外不可用，并且这种可不用可能沿请求调用链向上传递，这种现象在集群环境下就是雪崩。如下图

image.png

image.png

复杂分布式体系结构中的应用程序有几十个依赖项，每个依赖项在某个时候都不可避免地会失败。如果主机应用程序没有从这些外部故障中隔离出来，那么它就有可能与这些外部故障一起宕机。

* 当一切正常时，请求流可以是这样的:

image.png

* 当许多后端系统之一成为潜在，它可以阻止整个用户请求:

image.png

* 在高并发下，一个后端依赖项出现问题，可能会导致所有服务器上的所有资源在几秒钟内饱和。应用程序中通过网络或客户机库到达可能导致网络请求的每个点都是潜在故障的来源。比故障更糟的是，这些应用程序还可能导致服务之间的延迟增加，从而备份队列、线程和其他系统资源，从而导致系统中出现更多级联故障。

image.png

一个应用中，任意一个点的不可用或者响应延时都有可能造成服务不可用  
更可怕的是，被hang住的请求会很快耗尽系统的资源，当该类请求越来越多，占用的计算机资源越来越多的时候，会导致系统瓶颈出现，造成其他的请求同样不可用，最终导致业务系统崩溃

**雪崩效应常见场景**

* 硬件故障：如服务器宕机，机房断电，光纤被挖断等。
* 流量激增：如异常流量，重试加大流量等。
* 缓存穿透：一般发生在应用重启，所有缓存失效时，以及短时间内大量缓存失效时。大量的缓存不命中，使请求直击后端服务，造成服务提供者超负荷运行，引起服务不可用。
* 程序BUG：如程序逻辑导致内存泄漏，JVM长时间FullGC等。
* 同步等待：服务间采用同步调用模式，同步等待造成的资源耗尽。

最终的结果就是一个服务不可用导致一系列服务的不可用，而往往这种后果往往无法预料的。

**雪崩效应应对策略**

* 硬件故障：多机房容灾、异地多活等。
* 流量激增：服务自动扩容、流量控制（限流、关闭重试）等。
* 缓存穿透：缓存预加载、缓存异步加载等。
* 程序BUG：修改程序bug、及时释放资源等。
* 同步等待：资源隔离、MQ解耦、不可用服务调用快速失败等。资源隔离通常指不同服务调用采用不同的线程池；不可用服务调用快速失败一般通过超时机制，熔断器以及熔断后降级方法等方案实现。

作者：网恋被骗600万  
链接：https://www.jianshu.com/p/f6d7f832961d  
来源：简书  
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。