什么是缓存雪崩？服务器雪崩的场景与解决方案

题目标签

学习时长：20分钟

题目难度：中等

知识点标签：缓存雪崩、熔断模式、隔离模式、限流模式、隔离设计

题目描述

分布式系统都存在这样一个问题，由于网络的不稳定性，决定了任何一个服务的可用性都不是 100% 的。

当网络不稳定的时候，作为服务的提供者，自身可能会被拖死，导致服务调用者阻塞，最终可能引发雪崩连锁效应。

当缓存服务器重启或者大量缓存集中在某一个时间段失效，这样在失效的时候，也会给后端系统(比如DB)带来很大压力，造成数据库后端故障，从而引起应用服务器雪崩。

1. 雪崩效应产生的几种场景

- 流量激增：比如异常流量、用户重试导致系统负载升高；
- 缓存刷新：假设A为client端，B为Server端，假设A系统请求都流向B系统，请求超出了B系统的承载能力，就会造成B系统崩溃；

- 程序有Bug：代码循环调用的逻辑问题，资源未释放引起的内存泄漏等问题；
- 硬件故障：比如宕机，机房断电，光纤被挖断等。
- 数据库严重瓶颈，比如：长事务、sql超时等。
- 线程同步等待：系统间经常采用同步服务调用模式，核心服务和非核心服务共用一个线程池和消息队列。如果一个核心业务线程调用非核心线程，这个非核心线程交由第三方系统完成，当第三方系统本身出现问题，导致核心线程阻塞，一直处于等待状态，而进程间的调用是有超时限制的，最终这条线程将断掉，也可能引发雪崩；

2. 缓存雪崩的解决方案

缓存失效的几种情况:

1、缓存服务器挂了

2、高峰期缓存局部失效

3、热点缓存失效

解决方案:

1、避免缓存集中失效，不同的key设置不同的超时时间

2、增加互斥锁，控制数据库请求，重建缓存。

3、提高缓存的HA，如：redis集群。

3.雪崩的整体解决方案

一般情况对于服务依赖的保护主要有3种解决方案：

（1）熔断模式

这种模式主要是参考电路熔断，如果一条线路电压过高，保险丝会熔断，防止火灾。放到我们的系统中，如果某个目标服务调用慢或者有大量超时，此时，熔断该服务的调用，对于后续调用请求，不在继续调用目标服务，直接返回，快速释放资源。如果目标服务情况好转则恢复调用。

重点监控的机器性能指标

- cpu(Load) cpu使用率/负载
- memory 内存
- mysql监控长事务(这里与sql查询超时是紧密结合的，需要重点监控)
- sql超时
- 线程数等

总之，除了cpu、内存、线程数外，重点监控数据库端的长事务、sql超时等，绝大多数应用服务器发生的雪崩场景，都是来源于数据库端的性能瓶颈，从而先引起数据库端大量瓶颈，最终拖累应用服务器也发生雪崩，最后就是大面积的雪崩。

（2）隔离模式

这种模式就像对系统请求按类型划分成一个个小岛的一样，当某个小岛被火少光了，不会影响到其他的小岛。

例如可以对不同类型的请求使用线程池来资源隔离，每种类型的请求互不影响，如果一种类型的请求线程资源耗尽，则对后续的该类型请求直接返回，不再调用后续资源。这种模式使用场景非常多，例如将一个服务拆开，对于重要的服务使用单独服务器来部署，再或者公司最近推广的多中心。

（3）限流模式

上述的熔断模式和隔离模式都属于出错后的容错处理机制，而限流模式则可以称为预防模式。限流模式主要是提前对各个类型的请求设置最高的QPS阈值，若高于设置的阈值则对该请求直接返回，不再调用后续资源。这种模式不能解决服务依赖的问题，只能解决系统整体资源分配问题，因为没有被限流的请求依然有可能造成雪崩效应。

4. 熔断设计

在熔断的设计主要参考了hystrix的做法。其中最重要的是三个模块：熔断请求判断算法、熔断恢复机制、熔断报警

（1）熔断请求判断机制算法：使用无锁循环队列计数，每个熔断器默认维护10个bucket，每1秒一个bucket，每个blucket记录请求的成功、失败、超时、拒绝的状态，默认错误超过50%且10秒内超过20个请求进行中断拦截。

（2）熔断恢复：对于被熔断的请求，每隔5s允许部分请求通过，若请求都是健康的（RT<250ms）则对请求健康恢复。

（3）熔断报警：对于熔断的请求打日志，异常请求超过某些设定则报警。

隔离设计

隔离的方式一般使用两种

（1）线程池隔离模式：使用一个线程池来存储当前的请求，线程池对请求作处理，设置任务返回处理超时时间，堆积的请求堆积入线程池队列。这种方式需要为每个依赖的服务申请线程池，有一定的资源消耗，好处是可以应对突发流量（流量洪峰来临时，处理不完可将数据存储到线程池队里慢慢处理）

（2）信号量隔离模式：使用一个原子计数器（或信号量）来记录当前有多少个线程在运行，请求来先判断计数器的数值，若超过设置的最大线程个数则丢弃改类型的新请求，若不超过则执行计数操作请求来计数器+1，请求返回计数器-1。这种方式是严格的控制线程且立即返回模式，无法应对突发流量（流量洪峰来临时，处理的线程超过数量，其他的请求会直接返回，不继续去请求依赖的服务）

超时机制设计

（1）超时分两种，一种是请求的等待超时，一种是请求运行超时。

（2）等待超时：在任务入队列时设置任务入队列时间，并判断队头的任务入队列时间是否大于超时时间，超过则丢弃任务。

（3）运行超时：直接可使用线程池提供的get方法。

5. 如何提前发现雪崩

首先让系统不雪崩，然后通过监控发现请求正在接近或者超过阀值，然后再根据具体情况处理，这个接近或者超过阀值的过程，可以称为 “提前发现雪崩”。