

网购秒杀系统架构设计

案例分析

秒杀是电子商务网站常见的一种营销手段：将少量商品（通常只有一件）以极低的价格，在特定的时间点开始出售。比如一元钱的手机，五元钱的电脑，十元钱的汽车等。因为商品价格诱人，而且数量有限，所以很多人趋之若鹜，在秒杀活动开始前涌入网站，等到秒杀活动开始的一瞬间，点下购买按钮（在此之前购买按钮为灰色，不可以点击），抢购商品。这些商品因为在活动开始的一秒内就被卖光了，所以被称作秒杀。

网站通过这种营销手段，制造某种轰动效应，从而达到网站推广的目的。而最终能够被幸运之神眷顾，秒到商品的只有一两个人而已。很多电子商务网站已经把秒杀活动常态化了，经常性地举行秒杀活动。

秒杀虽然对网站推广有很多好处，也能给消费者带来利益（虽然是很少的几个人），但是对网站技术却是极大的挑战：网站是为正常运营设计的，而秒杀活动带来的并发访问用户却是平时的数百倍甚至上千倍。网站如果为秒杀时的最高并发访问量进行设计部署，就需要比正常运营多得多的服务器，而这些服务器在绝大部分时候都是用不着的，浪费惊人。所以网站的秒杀业务不能使用正常的网站业务流程，也不能和正常的网站交

易业务共用服务器，必须设计部署专门的秒杀系统，进行专门应对。

12.1 秒杀活动的技术挑战

假设某网站秒杀活动只推出一件商品，预计会吸引 1 万人参加活动，也就是说最大并发请求数是 10,000，秒杀系统需要面对的技术挑战有如下几点。

1. 对现有网站业务造成冲击

秒杀活动只是网站营销的一个附加活动，这个活动具有时间短，并发访问量大的特点，如果和网站原有应用部署在一起，必然会对现有业务造成冲击，稍有不慎可能导致整个网站瘫痪。

2. 高并发下的应用、数据库负载

用户在秒杀开始前，通过不停刷新浏览器页面以保证不会错过秒杀，这些请求如果按照一般的网站应用架构，访问应用服务器、连接数据库，会对应用服务器和数据库服务器造成极大的负载压力。

3. 突然增加的网络及服务器带宽

假设商品页面大小 200K（主要是商品图片大小），那么需要的网络和服务带宽是 2G（ $200K \times 10,000$ ），这些网络带宽是因为秒杀活动新增的，超过网站平时使用的带宽。

4. 直接下单

秒杀的游戏规则是到了秒杀时间才能开始对商品下单购买，在此时间点之前，只能浏览商品信息，不能下单。而下单页面也是一个普通的 URL，如果得到这个 URL，不用等到秒杀开始就可以下单了。

12.2 秒杀系统的应对策略

为了应对上述挑战，秒杀系统的应对策略有如下几点。

1. 秒杀系统独立部署

为了避免因为秒杀活动的高并发访问而拖垮整个网站，使整个网站不必面对蜂拥而来的用户访问，可将秒杀系统独立部署；如果需要，还可以使用独立的域名，使其与网站完全隔离，即使秒杀系统崩溃了，也不会对网站造成任何影响。

2. 秒杀商品页面静态化

重新设计秒杀商品页面，不使用网站原来的商品详情页面，页面内容静态化：将商品描述、商品参数、成交记录 and 用户评价全部写入一个静态页面，用户请求不需要经过应用服务器的业务逻辑处理，也不需要访问数据库。所以秒杀商品服务不需要部署动态的 Web 服务器和数据库服务器。

3. 租借秒杀活动网络带宽

因为秒杀新增的网络带宽，必须和运营商重新购买或者租借。为了减轻网站服务器的压力，需要将秒杀商品页面缓存在 CDN，同样需要和 CDN 服务商临时租借新增的出口带宽。

4. 动态生成随机下单页面 URL

为了避免用户直接访问下单页面 URL，需要将该 URL 动态化，即使秒杀系统的开发者也无法在秒杀开始前访问下单页面的 URL。办法是在下单页面 URL 加入由服务器端生成的随机数作为参数，在秒杀开始的时候才能得到。

12.3 秒杀系统架构设计

秒杀系统为秒杀而设计，不同于一般的网购行为，参与秒杀活动的用户更关心的是如何能快速刷新商品页面，在秒杀开始的时候抢先进入下单页面，而不是商品详情等用户体验细节，因此秒杀系统的页面设计应尽可能简单。如图 12.1 所示。

商品页面中的购买按钮只有在秒杀活动开始的时候才变亮，在此之前及秒杀商品卖出后，该按钮都是灰色的，不可以点击。

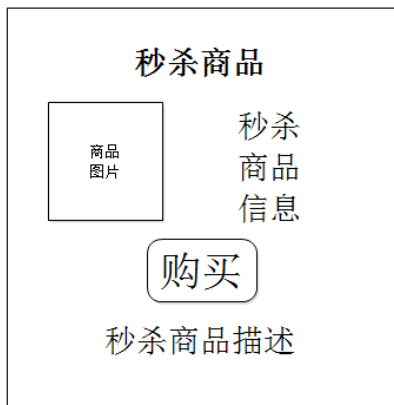


图 12.1 秒杀商品页面

下单表单也尽可能简单，购买数量只能是一个且不可以修改，送货地址和付款方式都使用用户默认设置，没有默认也可以不填，允许等订单提交后修改；只有第一个提交的订单发送给网站的订单子系统，其余用户提交订单后只能看到秒杀结束页面，如图 12.2 所示。

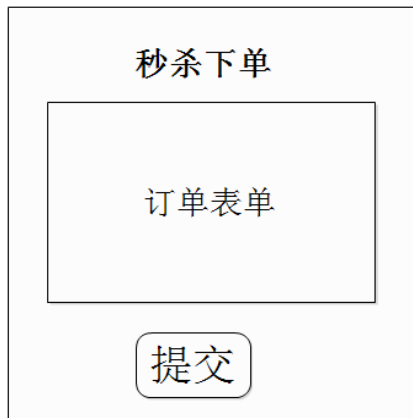


图 12.2 秒杀下单页面

除了上面提到的秒杀系统的技术挑战及应对策略，还有一些其他问题需要处理。

1. 如何控制秒杀商品页面购买按钮的点亮

购买按钮只有在秒杀活动开始的时候才能点亮，在此之前是灰色的。如果该页面是动态生成的，当然可以在服务器端构造响应页面输出，控制该按钮是灰色还是点亮，但是为了减轻服务器端负载压力，更好地利用 CDN、反向代理等性能优化手段，该页面被设计为静态页面，缓存在 CDN、反向代理服务器上，甚至用户浏览器上。秒杀开始时，用户刷新页面，请求根本不会到达应用服务器。

解决办法是使用 JavaScript 脚本控制，在秒杀商品静态页面中加入一个 JavaScript 文件引用，该 JavaScript 文件中加入秒杀是否开始的标志和下单页面 URL 的随机数参数，当秒杀开始的时候生成一个新的 JavaScript 文件并被用户浏览器加载，控制秒杀商品页面的展示。这个 JavaScript 文件使用随机版本号，并且不被浏览器、CDN 和反向代理服务器缓存。如图 12.3 所示。

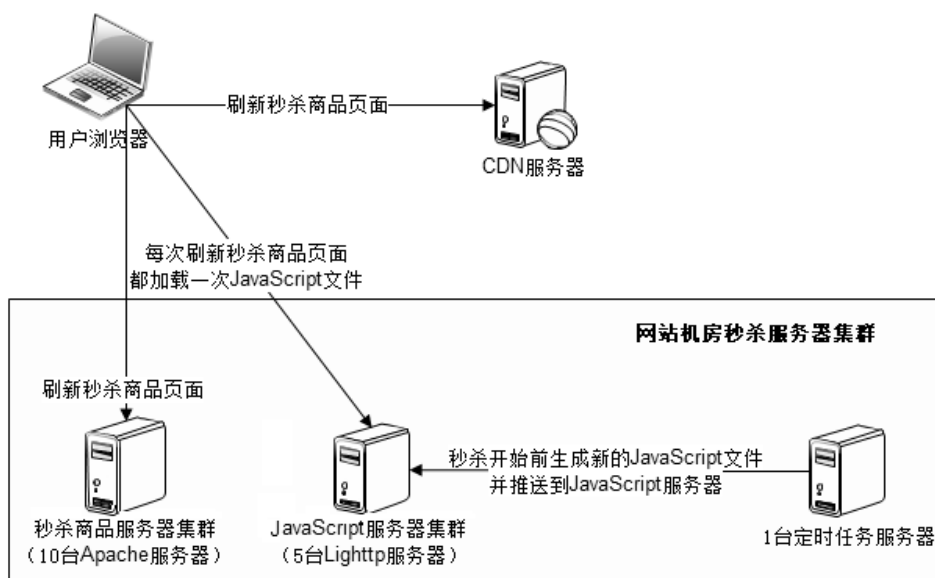


图 12.3 秒杀商品点亮过程

这个 JavaScript 文件非常小，即使每次浏览器刷新都访问 JavaScript 文件服务器也不会对服务器集群和网络带宽造成太大压力。

2. 如何只允许第一个提交的订单被发送到订单子系统

由于最终能够成功秒杀到商品的用户只有一个，因此需要在用户提交订单时，检查是否已经有订单提交。事实上，由于最终能够成功提交订单的用户只有一个，为了减轻下单页面服务器的负载压力，可以控制进入下单页面的入口，只有少数用户能进入下单页面，其他用户直接进入秒杀结束页面。假设下单服务器集群有 10 台服务器，每台服务器只接受最多 10 个下单请求，如图 12.4 所示。

秒杀系统的整体架构如图 12.5 所示。

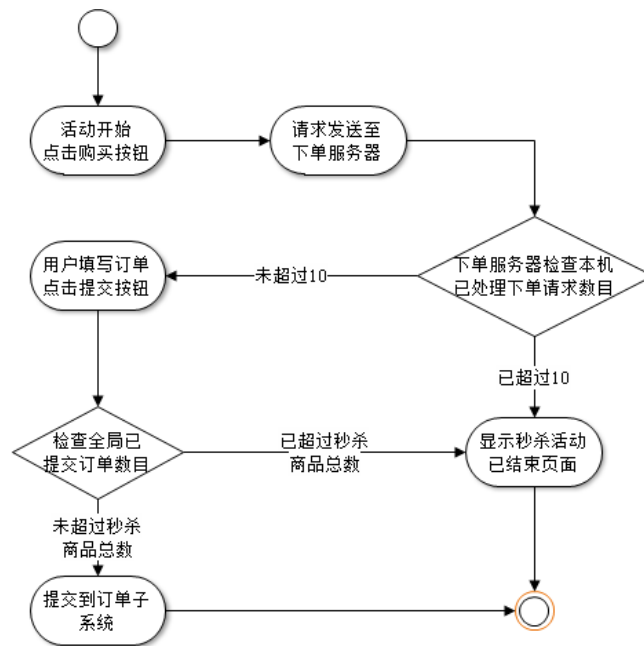


图 12.4 秒杀下单流程

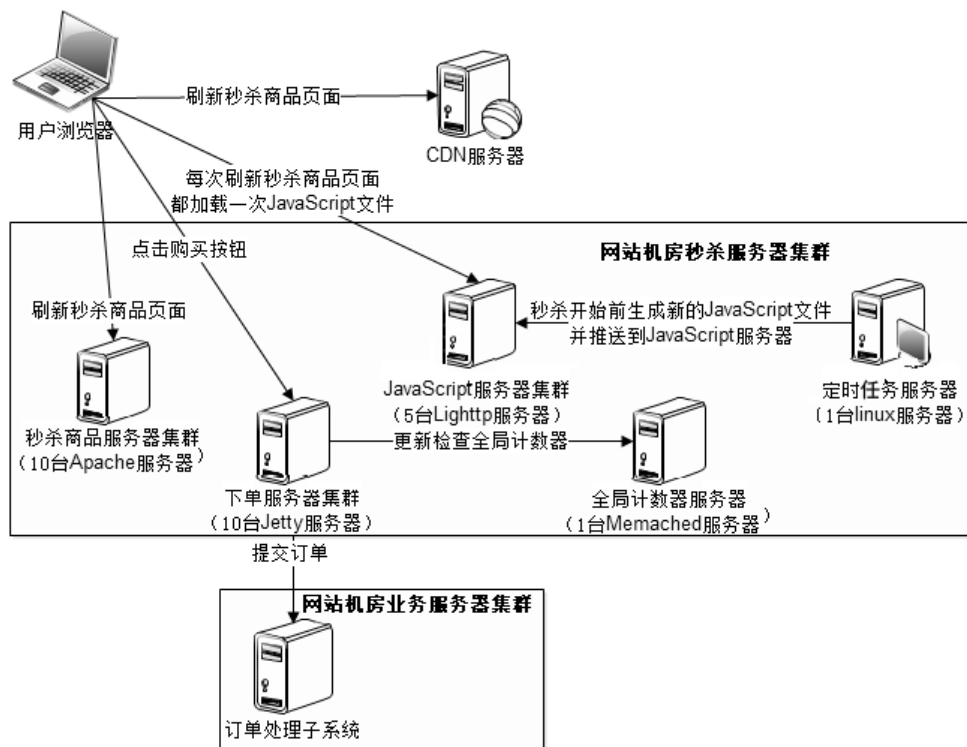


图 12.5 秒杀系统的整体架构

12.4 小结

秒杀是对网站架构的极大考验，在难以预计和控制的高并发访问的冲击下，稍有不慎，系统就会被用户秒杀，导致整个系统宕机，活动失败，构成重大事故。因此在遵循秒杀活动游戏规则的基础上，为了保证系统的安全，保持适度的公平公正即可。即使系统出了故障，也不应该给用户显示出错页面，而是显示秒杀活动结束页面，避免不必要的困扰。除了上面提到的一些针对秒杀活动进行的架构设计，在本书第 10 章中提到的许多性能优化设计都可以用于秒杀系统的优化。