设计一个秒杀系统，主要的挑战和问题有哪些？核心的架构方案或者思路有哪些？

1.秒杀系统的技术挑战：

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器带宽是 2G（200K×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。

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、成交记录和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。

3.秒杀系统的整体架构

秒杀过程：

1）秒杀开始前，用户刷新得到的是静态页面，存储在 CDN 服务器中；而秒杀页中的“抢购”定时点亮，是通过一个小的 javaScript 脚本控制的；

2）秒杀开始时，用户得到一个动态的 URL 下单地址进行秒杀（避免秒杀前的内幕操作）

3）强先到达服务器端的请求，通过分布式计数器获得锁，后到者则返回秒杀已结束的页面。