Apache电子商务解决方案之二 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/242/2021_2022_Apache_E7_94_B5_E5_c40_242492.htm 加速CRYPTO 一旦这些软件的问题被解决，并且我们有一个能够快速工作的会话高速缓冲存储器和支持会话恢复的浏览器，我们是否使用keep - alive连接就不再重要了。 现在，新的限制因素变成：我们每秒可以服务多少新的安全请求，概念上我们要知道建立一新的安全连接的延迟是多少（在非稳定缓慢增长的情况下，服务器可以支撑的负载极限的水平是多少）。这样在处理正常的http上我们有一个好处，就是安全地址很少需要被通告，因此也很少会出现在极短的时标上获取一个新的安全连接这样的现象（这类现象体现为负载峰值）。但是在最坏情况下，假设，有200个新的连接到你的站点，而你的站点每一秒仅能处理50签名，那么你潜在的客户为了新的连接最少要等待４秒，这就有可能就会导致这个潜在客户的流失。来自Zona研究一个报告表明，顾客在放弃发出他们的订单之前是没有耐心等待超过8秒钟的（而且这还必须包括处理全部请求的时间，而不仅仅只是处理SSL 连接的那部分）那么有什么解决方案能够解决现有的这种问题呢？ 高性能的机器部件 你可以购买一个更快的机器--一个每秒可以处理更多连接的机器。我们已经展示了一台采用廉价的Athlon处理器的机器，在进行RSA签名操作上比更为昂贵的Sparc Ultra 5机器要快上三倍！实际上，工程师们正在发展整个新的系列的处理器，这些将来的处理器特别用于加速诸如密钥签名等加密操作的处理。Ultrasparc III处理器将包含能够进行公共密钥加密操作的更有效率的指令。

美国英特尔公司也已经声称他们的新的Itanium 64位处理器可以以10倍于当前最快的32位Xeon处理器的速度来执行SSL操作。 密码加速方法 在处理器上处理密码操作的替换方法是使用一些"协处理器"，它们能够处理你的Web服务器上所有的和数学计算相关的任务。这是许多制造密码加速器硬件公司所乐于采用的方法。 现在大量的密码加速器主要采用包括Rainbow、DEC和nCipher加密算法。你可以通过购买加速硬件板来满足特定性能水平要求的加密功能，用于处理CPU密集型的耗时的RSA加密操作。当Web服务器获取一个新的连接时，它将相关的数学计算任务传递到加速板上，在板上执行计算后将结果送回Web服务器--有希望在极短的时间内完成计算并且可以继续处理Web服务器转发过来的计算任务。一旦初始的SSL会话被建立，加速板将不会在同一个SSL会话的后续连接中其作用。虽然这些板能作够做到对称加密，但事实上比起直接在CPU进行加密解密处理，单独将计算任务发送到加速硬件上进行计算，其花费的总体系统开销总是相对高的多（不管是在延迟还是系统资源的消耗上都是如此）。这导致密码加速器通常以附加硬件的形式，典型地通过PCI或SCSI总线进行连接，并且通常允许多个这样的设备串在一起为单台机器提供可伸缩性能的支持（包括容量的可伸缩性）。 你希望加速板有什么样的性能价格比以及如何测定它？这些公司通常引用加速板每秒能够支撑的1024位RSA签名操作的数目来判定，因此他们容易地同其他的加速器或我们的纯图形化软件做一比较。因为Web服务器子进程必须通过一些软件和硬件层与加速板通信以等待应答，因此在处理上显然会存在一点延迟。然而，这些块通讯对于CPU没有任何影响

，CPU专注于其他的任务，因此，这类延迟的增大并不会明显影响系统的吞吐量（除了那些需要运行大量子进程的任务，结果是消耗了大量的块通讯时间）。 我们已经对主要的一些加密硬件生产厂商所提供的产品价格进行了调查，并作为ICSA信息安全杂志一份报告公布于众。 我们隐藏了供应商和模型的名称，仅仅列出那些主要供应商多提供的产品中能够进行RSA密钥操作和密钥管理的产品，形成一个可以对照的产品特性列表。这些不同的单元能够显示不同产品之间的安全特性之差异，以解释各个产品之间价格差异的主要原因。 我们可以比较在通用的统一平台下进行同样加密操作时，不同产品之间性能和价格之间的差异。 我们发现如果借助一些负载均衡软件在几个CPU之间平衡负载，我们能在已有的三台AMD Athlon朱基上产生和高端加密硬件产品一样好的吞吐量，要知道这些专有的加密产品其价格至少是这些ADM机器的三倍。同时，我们也将拥有一个更易于升级和配置的系统结构，借助大量的系统级冗余，其实已经不再需要专有的控制系统来管理加密硬件设备了。但是目前的软件尚不能实现加密计算的集群构架，因此由单独的每一台机器处理各自的SSL连接，而在处理SSL连接之间则会先进行负载平衡的任务分配工作，这个负责分配的均衡器角色需要安置在这些加密处理机之前。 采用专有硬件密码加速器的另一个问题时，如果你的Web服务器超过一台，你就需要为每一台新的web服务器单独配置一个硬件密码加速器。如果，你仅仅是出于利用冗余的Web服务器保证容错能力而考虑的话，你就会为这些实际上并不需要的硬件加速器多支付额外的高昂费用。这就是目前大多数硬件加速器被设置为一对一服务和一对多服

务两种方式的主要原因（一个加速器可以连接到一个或者多个Web服务器上，防止亦然）。另外，许多的站点使用密码硬件加速器主要是为了密钥管理的需要而不是加速加密操作。这样，内部产生的私有密钥在使用上就会有严格的限制，可以有效地防止外部的干扰并且对于外部访问者而言是不可见的，因为加密器不会输出密钥的内容。甚至Web站点的系统管理员都无法访问这个密钥的具体内容，要知道站点管理员拥有多大的权限：启动和停止Web服务器！同时，对于站点里那些知晓某些可以解析私有密钥方法的系统操作者，这样的防范也是很有效的。假设，你拥有10个 Web服务器并且已经将他们用于处理大量的数据库查询任务和Web访问逻辑，但你的系统中SSL相关的处理任务有非常小；但是处于安全考虑，你不得不在每一台Web服务器上配置单独的密码加速器以管理好你的硬件密钥。实际上，这样的效果并不理想：由于SSL负载很低，你的大多数硬件加速器总是处理很低的利用率下，严重浪费了你的投资！ 移除性能瓶颈 对于普通的HTTP请求，存在着大量的软硬件解决方案来处理负载平衡的问题。处理负载平衡的一般方法是，监视进入的HTTP请求，然后根据某种规则在服务器节点列表中选择一台Web服务器，将这个请求传递给这台被选中的服务器节点。由于HTTP是一种无状态的协议，因此可以针对每一个网络连接进行传递操作，将不同的HTTP请求（包括在同一个网页中，单独的图片、媒体文件、HTML文本都是独立的发出连接请求）转发给不同的后段服务器。因此这种模型可以工作的很好不会出现任何问题。然而，在SSL环境下我们不希望出现这种情况，大多数不同的HTTP连接请求需要发往同一个后台主机，以

避免新的HTTP请求要同新的服务主机重新协商并建立新的SSL连接以获取密钥。我们应该采用必要的技术手段避免这种问题的出现，要知道，可能由于不断的计算新的SSL加密密钥，这个问题还可能演变成用户浏览网页时电击页面的响应性能问题。某些负载均衡技术可以通过判断请求包的源地址，以强制所有从同一个客户端发出的请求都会发向同一个后端服务主机。但是，这种做法本身在原理上就违背了负载均衡的基本理念，因为大量的客户可能通过同一台ISP拨号服务器或者是公司的代理服务起来访问站点，这些用户的地址最终将表现为同一个 IP地址，你怎么区分他们？！ 如果你需要硬件加速器来加速密钥计算任务，并且你拥有多个后端主机用于处理，你将不得不为每一个处理主机配置单独的硬件加速器（11）。这就意味着你需要大量的硬件加速器（和你的服务器数目相当），而你的加速器供应商会非常喜欢你！同时，这也意味着它将变成一笔高昂投资，而你实际上购买了多余的硬件加密授权来用于你那利用率并不高的SSL加密任务。那么，既然普通HTTP请求的负载均衡技术如此的易于应用，我们不妨来看一下现有几种流行的负载均衡技术能否适用于处理HTTPS下的SSL连接的负载均衡任务。 