Apache:MPM的引入带来性能改善 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/252/2021_2022_Apache_MPM_c103_252901.htm Apache 2.0在性能上的改善最吸引人。在支持POSIX线程的Unix系统上，Apache可以通过不同的MPM运行在一种多进程与多线程相混合的模式下，增强部分配置的可扩充性能。相比于Apache 1.3，2.0版本做了大量的优化来提升处理能力和可伸缩性，并且大多数改进在默认状态下即可生效。但是在编译和运行时刻，2.0也有许多可以显著提高性能的选择。本文不想叙述那些以功能换取速度的指令，如HostnameLookups等，而只是说明在2.0中影响性能的最核心特性：MPM（Multi-Processing Modules，多道处理模块）的基本工作原理和配置指令。 毫不夸张地说，MPM的引入是Apache 2.0最重要的变化。大家知道，Apache是基于模块化的设计，而Apache 2.0更扩展了模块化设计到Web服务器的最基本功能。服务器装载了一种多道处理模块，负责绑定本机网络端口、接受请求，并调度子进程来处理请求。扩展模块化设计有两个重要好处：◆ Apache可以更简洁、有效地支持多种操作系统；◆ 服务器可以按站点的特殊需要进行自定制。 在用户级，MPM看起来和其它Apache模块非常类似。主要区别是在任意时刻只能有一种MPM被装载到服务器中。 指定MPM的方法 下面以Red Hat Linux 9为平台，说明在Apache 2.0中如何指定MPM (Apache采用2.0.45)。先解压缩源代码包httpd-2.0.45.tar.gz，生成httpd-2.0.45目录（Apache 1.3源代码包的命名规则是apache_1.3.NN.tar.gz，而2.0版则是httpd-2.0.NN.tar.gz，其中NN是次版本号）。 进

入httpd-2.0.45目录，运行以下代码： $ ./configure --help|grep mpm 显示如下：--with-mpm=MPMChoose the process model for Apache to use.MPM={beos|worker|prefork|mpmt_os2|perchild|leader|threadpool} 上述操作用来选择要使用的进程模型，即哪种MPM模块。Beos、mpmt_os2分别是BeOS和OS/2上缺省的MPM，perchild主要设计目的是以不同的用户和组的身份来运行不同的子进程。这在运行多个需要CGI的虚拟主机时特别有用，会比1.3版中的SuExec机制做得更好。leader和threadpool都是基于worker的变体，还处于实验性阶段，某些情况下并不会按照预期设想的那样工作，所以Apache官方也并不推荐使用。因此，我们主要阐述prefork和worker这两种和性能关系最大的产品级MPM (有关其它的MPM详细说明，请参见Apache官方文档：http://httpd.apache.org/docs-2.0/mod/)。 prefork的工作原理及配置 如果不用“ --with-mpm ”显式指定某种MPM，prefork就是Unix平台上缺省的MPM。它所采用的预派生子进程方式也是Apache 1.3中采用的模式。prefork本身并没有使用到线程，2.0版使用它是为了与1.3版保持兼容性；另一方面，prefork用单独的子进程来处理不同的请求，进程之间是彼此独立的，这也使其成为最稳定的MPM之一。 若使用prefork，在make编译和make install安装后，使用“ httpd -l ”来确定当前使用的MPM，应该会看到prefork.c（如果看到worker.c说明使用的是worker MPM，依此类推）。再查看缺省生成的httpd.conf配置文件，里面包含如下配置段： StartServers 5 MinSpareServers 5 MaxSpareServers 10 MaxClients 150 MaxRequestsPerChild 0 prefork的工作原理是，控制进程在最初建立“ StartServers ”个

子进程后，为了满足MinSpareServers设置的需要创建一个进程，等待一秒钟，继续创建两个，再等待一秒钟，继续创建四个……如此按指数级增加创建的进程数，最多达到每秒32个，直到满足MinSpareServers设置的值为止。这就是预派生（prefork）的由来。这种模式可以不必在请求到来时再产生新的进程，从而减小了系统开销以增加性能。
MaxSpareServers设置了最大的空闲进程数，如果空闲进程数大于这个值，Apache会自动kill掉一些多余进程。这个值不要设得过大，但如果设的值比MinSpareServers小，Apache会自动把其调整为MinSpareServers 1。如果站点负载较大，可考虑同时加大MinSpareServers和MaxSpareServers。
MaxRequestsPerChild设置的是每个子进程可处理的请求数。每个子进程在处理了“ MaxRequestsPerChild ”个请求后将自动销毁。0意味着无限，即子进程永不销毁。虽然缺省设为0可以使每个子进程处理更多的请求，但如果设成非零值也有两点重要的好处：◆ 可防止意外的内存泄漏；◆ 在服务器负载下降的时侯会自动减少子进程数。 因此，可根据服务器的负载来调整这个值。笔者认为10000左右比较合适。
MaxClients是这些指令中最为重要的一个，设定的是Apache可以同时处理的请求，是对Apache性能影响最大的参数。其缺省值150是远远不够的，如果请求总数已达到这个值（可通过ps -ef|grep http|wc -l来确认），那么后面的请求就要排队，直到某个已处理请求完毕。这就是系统资源还剩下很多而HTTP访问却很慢的主要原因。系统管理员可以根据硬件配置和负载情况来动态调整这个值。虽然理论上这个值越大，可以处理的请求就越多，但Apache默认的限制不能大于256。

如果把这个值设为大于256，那么Apache将无法起动。事实上，256对于负载稍重的站点也是不够的。在Apache 1.3中，这是个硬限制。如果要加大这个值，必须在“ configure ”前手工修改的源代码树下的src/include/httpd.h中查找256，就会发现“ #define HARD_SERVER_LIMIT 256 ”这行。把256改为要增大的值（如4000），然后重新编译Apache即可。在Apache 2.0中新加入了ServerLimit指令，使得无须重编译Apache就可以加大MaxClients。下面是笔者的prefork配置段： StartServers 10 MinSpareServers 10 MaxSpareServers 15 ServerLimit 2000 MaxClients 1000 MaxRequestsPerChild 10000 上述配置中，ServerLimit的最大值是20000，对于大多数站点已经足够。如果一定要再加大这个数值，对位于源代码树下server/mpm/prefork/prefork.c中以下两行做相应修改即可：#define DEFAULT_SERVER_LIMIT 256#define MAX_SERVER_LIMIT 20000