

Nginx的架构优势与功能简述

2016-01-27 23:33 feiying 0 阅读 81

Nginx 是现在比较常用的web服务器软件，它最初是一个叫做 IforSysoev的俄罗斯人为 Rambler.ru站点开发的，

因为其能够选择高效的epoll(基于Linux的2.6内核)，kqueue(仅仅在FreeBSD操作系统中运行的)，eventport(Solaris 10)，

这三个高效的网络IO模型，是高并发的前提，而Nginx在跨平台的方案中综合了上述的方案，在一段时期之内（Apache最新版本也对性能进行），

是Apache服务器不错的替代品，Nginx号称可以支撑50000个并发请求，相比之下，cpu和内存的消耗非常低，整体运行非常稳定。

Nginx的架构是典型的多进程模型：

master进程类似于应用服务器中的集群中的主监视Node节点，而work进程类似于应用服务器集群中的slave节点，

因为是进程模式，所以1个worker进程崩溃，不会导致整个Nginx服务器的崩溃，除非是Master进程崩溃，Nginx才崩溃，

但是Master进程却并不参与应用请求，所以从图中可以看到 client只与worker进程交互，大并发的压力都在worker进程中，

这种模式也是为什么Nginx稳定的原因，试想一下，在java服务器中，因为JVM仅仅就是一个进程，所以一旦出现什么风吹草动的话，

那么，JVM直接就会崩溃掉，而JVM崩溃掉，哪还有什么线程运行，服务器安在？

====》这就是Nginx为什么不推荐使用多线程的主要原因，为的就是一个**多进程稳定可靠**；
==》1

其次，思考一下多线程的问题，因为同一进程内是共享资源的，所以java API中搞出来很多的锁，什么synchronized锁，

什么轻量级锁，独占锁，又搞出来一个CAS指令等等。

下载《开发者大全》

下载 (/download/dev.apk)



因为进程中资源是有限的，所以这些锁互相钳制，一个等一个，因此效率是很低的，这就是第二个原因，**线程锁的消耗；===》2**

多线程编程非常复杂，在系统调用层级的mutex_lock写起来云山雾绕的，java还稍好一些，进程级就是基于pid进行编程，

遇到子进程，直接进行fork，在java语言中，直接通过Runtime进行运行子进程也是一个道理，总结下来，**进程编程肯定比线程容易；===》3**

上面说了多进程这么多得好，但是不得不说，从效率上来讲，多线程肯定要大大超过多进程，这个没的说，因为线程切换一下的代价，

没有进程转储的这个过程，线程共享进程中的资源嘛，因此切换是很快的。

但是，上述的前提是并没有加入更好的网络IO模型的，如果一旦使用了这种epoll模型（linux），java对应就是版本7以后的NIO或者Netty这种框架，

epoll的模式是基于事件的，也就是说我可以通过1个进程/线程来监听多路的传输包，因此从术语上讲解，这就叫做IO多路复用，

而epoll推荐的进程/线程数较少，一般会根据cpu的核数来选择，n+1（1为多出来的分页），

而目前服务器中的核数一般不多，4核、8核、16核居多，因此在这么少的进程/线程下，进程上述的劣势并不一定比线程要差，

这种优劣式取决于核数，还有你的epoll的进程/线程数的设置，而可以发现Nginx的Worker进程，1进程1线程，线程中运行的是一个epoll实例，

关注多路的网络IO的请求，因此如果是这种模式下，Nginx的Worker进程可谓是没有任何的线程切换消耗，而网络IO效率仍旧很高，

相比之下，如果你使用新版Nginx的多线程模式的话，那么当并发上到1000以上的时候，工作线程中就会有大量的线程切换；

因此，总结一下，**epoll这种IO多路复用机制下的1进程1线程的模式，比多线程的模式效率要高，没有大量的线程切换=====》4**


以上4点，就是Nginx使用多进程的原因，虽然现在新版本的Nginx也支持多线程，但多进程的Nginx的Worker的模式仍旧是主流；

讲述完成架构以后，其实还没有进入正题，需要给Nginx下一个定义：

Nginx是一个高性能的HttpServer，同时也是一个反向代理服务器。

HttpServer：总所周知，以tomcat为例，服务器前端务必要有http协议解析的部分，而Nginx从其http.conf的配置上就可以看到

```
http
{
    server
    {
        listen      80 default;
        server_name _ *;
        access_log   logs/default.access.log combined;
        location / {
            index index.html;
            root /data0/htdocs/htdocs;
        }
    }
}
```



应用服务器技术讨论圈

如果是静态的文件的话，直接放到root属性中，index的属性就是主页，

对于Nginx来讲，默认支持静态的html网页，但是对于一些混杂有语言脚本的页面，例如php这种是需要解释执行的，有一个翻译的过程，

可以将这种翻译机制做成一个插件，安装在Nginx中即可，这就是PHP模块了，类似于apache。

对于jsp，你也可以同样以这种模式做一个插件进行解释，但通常tomcat在这一方面做得更好，而jsp也不是单纯的脚本语言混杂的页面，

jsp中有很多的标签，指令，需要进行编译成servlet，因此通常Nginx+tomcat这种模式，才是推荐的方式，而这种方式也就是反向代理的功能。

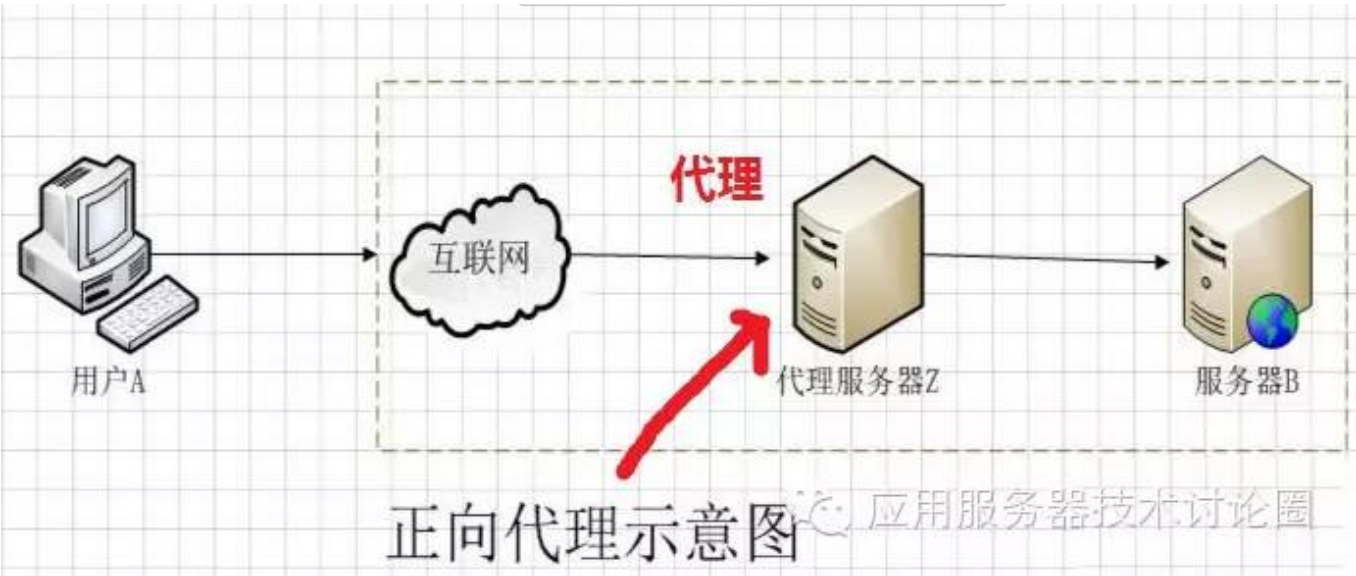
反向代理：反向代理这个概念有点陌生，不过说起正向代理肯定能明白，例如翻墙，当我们直接连不到互联网的时候，是被咱们国家的防火墙给qiang了，

我们需要在国外找一个没被qiang的服务器，这个服务器有点特别，能帮着我们转发请求，然后将网页的内容返回到你的机器上，

这个模式就是正向代理，也就是说现在人们炒得很热的自由门啊，这种东西就是正向代理软件，当然这种方式上网极其不安全，毕竟你所走的内容全部

都要经过代理服务器，而这些代理服务器为什么免费都是有利益的，因而目前现在大家普遍翻墙的办法，就是在国外租一个专有的VPN账号，1年百十来块钱也就

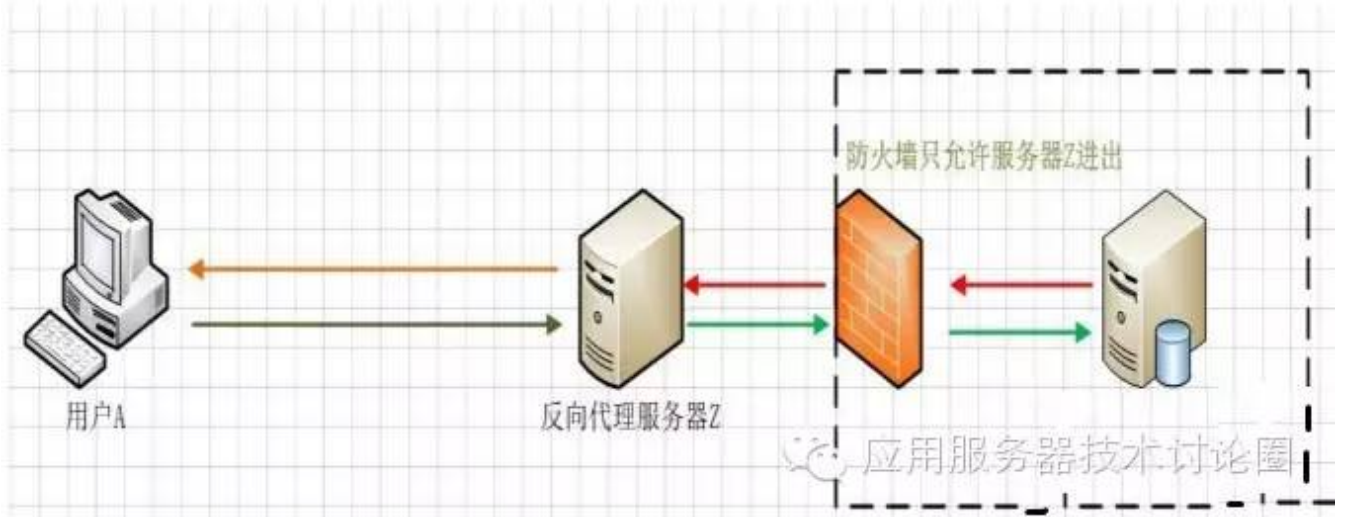
搞定了... 这就是正向代理，总结起来就是，内部用户访问不到外网，但是还有此需求，就通过一个代理机器进行访问，而这种代理模式，就是正向代理；



正向代理的整体示意图如上，而反向代理，就是方向是反着的，也就是外面的请求我想请求到内部的某一台服务器中，在外界做一个代理，

这种好处有2个，

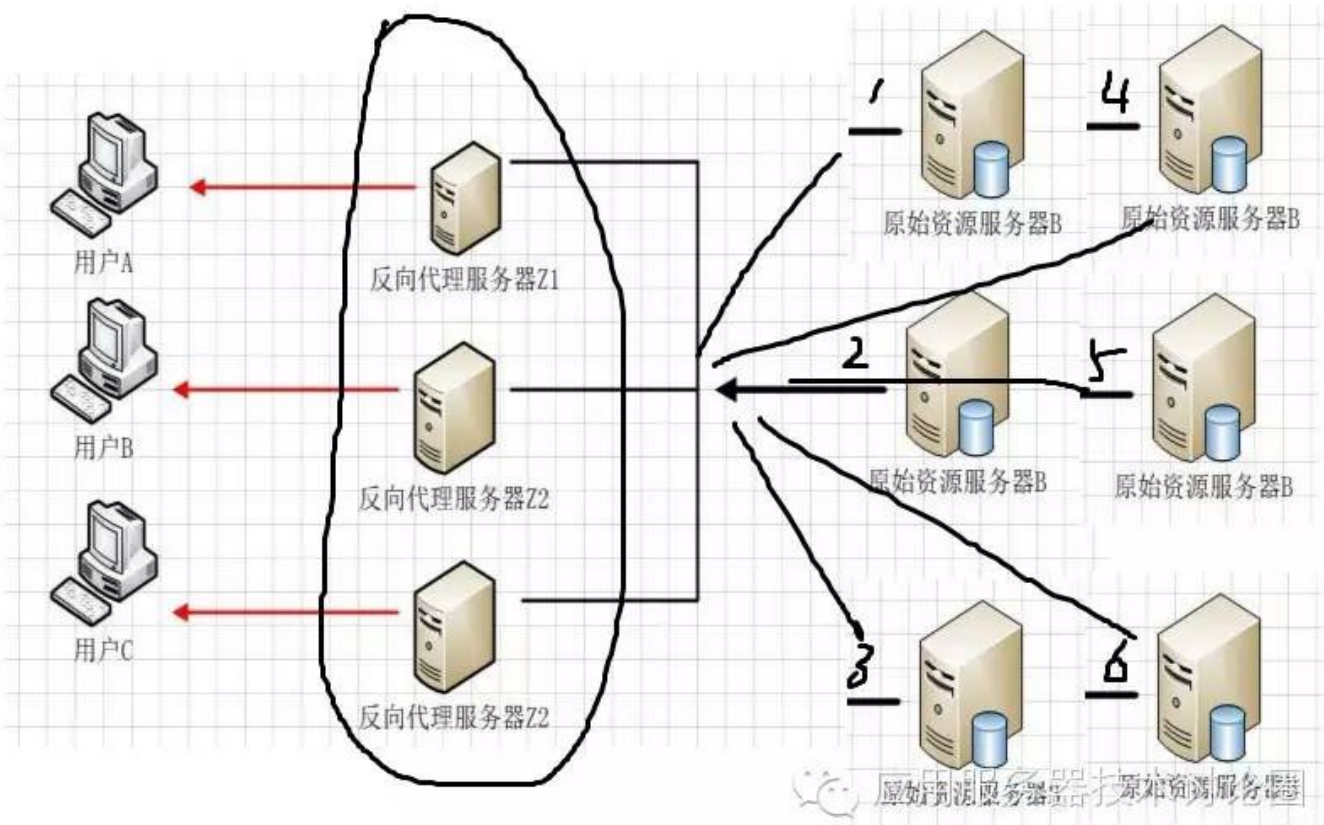
一种是通过一个代理机制来保护内部的服务器，这个服务器很有机密，需要保护，如下图所示：



用户A仅仅知道反向代理服务器Z的地址，即使他想攻击内部服务器，但是他不知道内部服务器的地址，甚至如上图所示，就直接用一道防火墙给qiang了，

什么都看不见；

另一种好处就是增加一些域名，做成集群模式，如下图所示，



增加域名这种手段，其实是DNS的活，但是DNS有两大知名的缺陷，

一个是机器坏掉，DNS配置之后仍旧会请求坏掉的机器，这种应该类似有一个心跳检测的东西，但是不好意思，DNS没有；

其次，DNS负载分配不平衡，以上图为例，如果反向代理手段使用的是DNS，那么极有可能内部服务器有的压力很大，有的闲的无聊，

而上述的2点原因，都是Nginx能解决的，并且负载策略Nginx和Apache都有n种策略，而这也就是Nginx作为反向代理的作用。

HttpServer，反向代理是Nginx的两大最重要的功能，使用占比超过9成以上，但是不得不说的是，Nginx**也是一个邮件代理服务器**，

它支持IMAP/POP3/SMTP的功能：

__IMAP/POP3 代理服务功能：__

- ☞ 使用外部 HTTP 认证服务器重定向用户到 IMAP/POP3 后端；
- ☞ 使用外部 HTTP 认证服务器认证用户后连接重定向到内部的 SMTP 后端；
- ☞ 认证方法：
 - ☞ POP3: POP3 USER/PASS, APOP, AUTH LOGIN PLAIN CRAM-MD5;
 - ☞ IMAP: IMAP LOGIN;
 - ☞ SMTP: AUTH LOGIN PLAIN CRAM-MD5;
- ☞ SSL 支持；
- ☞ 在 IMAP 和 POP3 模式下的 STARTTLS 和 STLS 支持；

这个功能其实也就是1996年最初给Rambler.ru站点开发Nginx所使用的最初的功能；

除此之外，支持FastCGI脚本，SSL，基于PCRE的rewrite重写等等限制，这个我们后续再说。

分享：

阅读 81 0

应用服务器技术讨论圈 更多文章

东方通加码大数据业务 拟募资8亿收购微智信业 (/html/308/201504/206211355/1.html)

玩转Netty – 从Netty3升级到Netty4 (/html/308/201504/206233287/1.html)

金蝶中间件2015招聘来吧！Come on！ (/html/308/201505/206307460/1.html)

GlassFish 4.1 发布，J2EE 应用服务器 (/html/308/201505/206323120/1.html)

Tomcat对keep-alive的实现逻辑 (/html/308/201505/206357679/1.html)

猜您喜欢

史上最严的隐私条例出台，2018年开始执行 (/html/204/201605/2652761511/1.html)

Ceph中Bufferlist的设计与使用 (/html/362/201505/204950308/1.html)

比较全面的MySQL优化参考（上篇） (/html/251/201505/205739222/1.html)

我看锤子：极端的性格带来极端的产品 (/html/277/201506/207658372/1.html)

思想 | 意识上传电脑实现永生，机器之心觉得这是「天方夜谭」 (/html/162/201603/402925784/1.html)