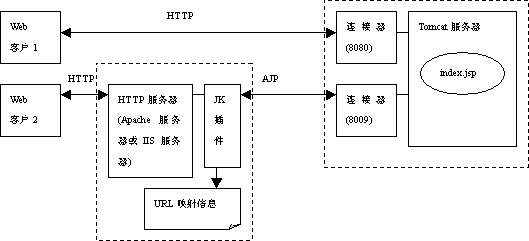
**tomcat的安全配置与优化**

参考博文：

<http://blog.csdn.net/our_sky/article/details/51362676>

<https://github.com/judasn/Linux-Tutorial/blob/master/Tomcat-Install-And-Settings.md>



图中间虚线框部分是 Apache基金下的服务器来做静态资源处理的，而这部分需要花费大量时间，当用nginx和tomcat做企业级集群的时候，需要禁用掉AJP协议。

一.安全设置

1.隐藏版本信息

1.1修改server.xml配置文件

# cd  $TOMCAT\_HOME/lib/ ; jar -xf catalina.jar

# cd  org/apache/catalina/util/

# vim  ServerInfo.properties

server.info=xixihaha  //改成你想要的名字

server.number=     //空掉

server.built=     //空掉

# cd  $TOMCAT\_HOME/lib/

# rm -rf catalina.jar  &&  jar -cvf  catalina.jar    META-INF    org

重启tomcat

备注：查看tomcat版本

# cd  $TOMCAT\_HOME/bin ;  sh  version.sh

...

Server version: xixihaha   //已经改成上面修改的版本名字

...

1.2.禁用tomcat管理界面

  生产环境一般不适用Tomcat默认的管理界面，这些页面存放在Tomcat 的webapps安装目录下，

把该目录下的所有文件删除即可：

# cd   $TOMCAT\_HOME/webapps/  ; rm -rf   \*

1.3.更改关闭tomcat的指令

# vim server.xml

       修改实例：

           <Server port="8005" shutdown="9SDKJ29jksjf23sjf0LSDF92JKS9DKkjsd">

        或者禁用8005端口

            <Server port="-1" shutdown="SHUTDOWN">

二.性能提升

2.1 链接优化

a. 配置链接池tomcatThreadPool：

在<Service name="catalina">下面添加

<Executor name="tomcatThreadPool" namePrefix="catalina-exec-"

        maxThreads="150" minSpareThreads="100"

        prestartminSpareThreads="true" maxQueueSize="100"/>

参数讲解：

* name: 线程名称，后面connector模块会引用到
* namePrefix: 线程前缀
* maxThreads : 最大并发连接数，不配置时默认200，一般建议设置500~ 800 ，要根据自己的硬件设施条件和实际业务需求而定。
* minSpareThreads：Tomcat启动初始化的线程数，默认值25
* prestartminSpareThreads：在tomcat初始化的时候就初始化minSpareThreads的值， 不设置true时minSpareThreads
* maxQueueSize: 最大的等待队列数，超过则拒绝请求

b. **优化连接器**connector

<Connector port="8080" protocol="org.apache.coyote.http11.Http11Nio2Protocol"

        connectionTimeout="20000"

        redirectPort="8443"

       executor="tomcatThreadPool"          enableLookups="false"

        acceptCount="100"

        maxPostSize="10485760"

       compression="on"

        disableUploadTimeout="true"

        compressionMinSize="2048"

        noCompressionUserAgents="gozilla, traviata"

        acceptorThreadCount="2"

        compressableMimeType="text/html,text/xml,text/plain,text/css,text/javascript,application/javascript"

        URIEncoding="utf-8"/>

参数讲解：

* port：连接端口。
* protocol（通信模式）：连接器使用的传输方式。  Tomcat 8 设置 nio2 更好：org.apache.coyote.http11.Http11Nio2Protocol

            protocol， Tomcat 6、7 设置 nio 更好：org.apache.coyote.http11.Http11NioProtocol

备注：

           每个web客户端请求对于服务器端来说就一个单独的线程，客户端的请求数量增多将会导致线程数就上去了，CPU就忙着跟线程切换。 而NIO则是使用单线程(单个CPU)或者只使用少量的多线程(多CPU)来接受Socket，而由线程池来处理堵塞在pipe 或者队 列里的请求.这样的话，只要OS可以接受TCP的连接，web服务器就可以处理该请求。大大提高了web服务器的可伸缩性。

* executor： 连接器使用的线程池名称
* enableLookups：禁用DNS  查询
* acceptCount：指定当所有可以使用的处理请求的线程数都被使用时，可以放到处理队列中的请求数，超过这个数的请求将不予处理，默认设置 100 。
* maxPostSize：限制 以FORM URL 参数方式的POST请求的内容大小，单位字节，默认是 2097152(2兆)，10485760 为 10M。如果要禁用限制，则可以设置为 -1。
* acceptorThreadCount： 用于接收连接的线程的数量，默认值是1。一般这个指需要改动的时候是因为该服务器是一个多核CPU，如果是多核 CPU 一般配置为 2。
* compression：传输时是压缩。
* compressionMinSize：压缩的大小
* noCompressionUserAgents：不启用压缩的浏览器

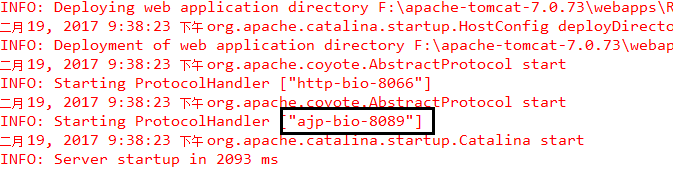
2.2 管理AJP端口

AJP是为Tomcat与HTTP服务器之间通信而定的协议，能提供较高的通信速率和效率，如果Tomcat前段放的是Apache，会用到AJP，默认是开启的，如果不是Apache，注释掉

<!--Connector port="8009" protocol="AJP/1.3" redirectPort="8443" /-->

如果不注释的话，启动的时候AJP协议是开启

# tail -f  $TOMCAT\_HOME/log/catalina.out



备注：

tail -f   $TOMCAT\_HOME/log/catalina.out  可以查看tomcat启动时间，做下前后对比

2.3   JVM优化  <https://github.com/judasn/Linux-Tutorial/blob/master/Tomcat-Install-And-Settings.md>

如果服务器只运行一个 Tomcat，机子内存如果是 16G：

* CATALINA\_OPTS="-Dfile.encoding=UTF-8 -server -Xms8192m -Xmx8192m -Xmn4096m -XX:PermSize=256m -XX:MaxPermSize=512m -XX:SurvivorRatio=10 -XX:MaxTenuringThreshold=15 -XX:NewRatio=2 -XX:+DisableExplicitGC"

参数说明：

Dfile.encoding：默认文件编码

-server：表示这是应用于服务器的配置，JVM 内部会有特殊处理的

-Xmx8192m：设置JVM最大可用内存为8192MB

-Xms8192m：设置JVM最小内存为8192m。此值可以设置与-Xmx相同，以避免每次垃圾回收完成后JVM重新分配内存。

-Xmn4096m：设置JVM新生代大小（JDK1.4之后版本）。一般-Xmn的大小是-Xms的1/2左右，不要设置的过大或过小，过大导致老年代变小，频繁Full GC，过小导致minor GC频繁。如果不设置-Xmn，可以采用-XX:NewRatio=2来设置，也是一样的效果

-XX:NewSize：设置新生代大小

-XX:MaxNewSize：设置最大的新生代大小

-XX:PermSize：设置永久代大小

-XX:MaxPermSize：设置最大永久代大小

-XX:NewRatio=4：设置年轻代（包括 Eden 和两个 Survivor 区）与终身代的比值（除去永久代）。设置为 4，则年轻代与终身代所占比值为 1：4，年轻代占整个堆栈的 1/5

-XX:MaxTenuringThreshold=10：设置垃圾最大年龄，默认为：15。如果设置为 0 的话，则年轻代对象不经过 Survivor 区，直接进入年老代。对于年老代比较多的应用，可以提高效率。如果将此值设置为一个较大值，则年轻代对象会在 Survivor 区进行多次复制，这样可以增加对象再年轻代的存活时间，增加在年轻代即被回收的概论。需要注意的是，设置了 -XX:MaxTenuringThreshold，并不代表着，对象一定在年轻代存活15次才被晋升进入老年代，它只是一个最大值，事实上，存在一个动态计算机制，计算每次晋入老年代的阈值，取阈值和MaxTenuringThreshold中较小的一个为准。

2.4 配置获取客户端真是IP

当nginx做tomcat的代理时，查看localhost\_acces.log，回发现客户端ip都变成了127.0.0.1

# vim server.xml

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="/opt/dfcp/logs/tomcat\_19001"

               prefix="localhost\_access\_log" suffix=".txt"

               pattern="**%{X-real-IP}i** %l %u %t &quot;%r&quot; %s %b" />     //将%h改为**%{X-real-IP}i**