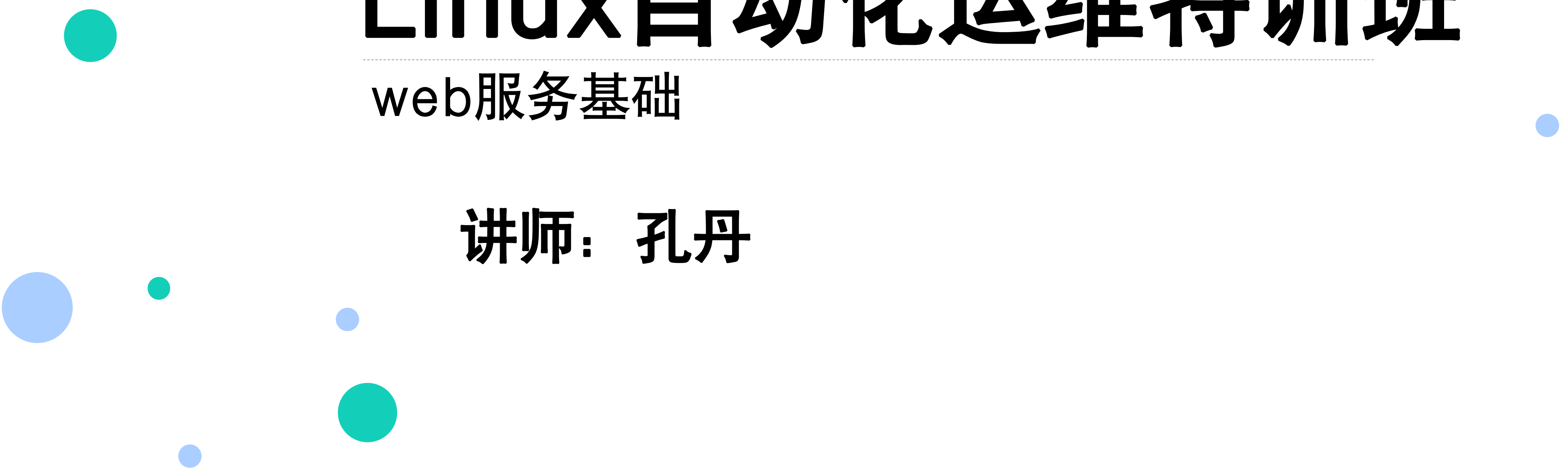




Linux自动化运维特训班

Tomcat实战

讲师：孔丹



大纲

- Tomcat简介
- Tomcat目录结构
- Tomcat配置文件
- Tomcat多实例
- Tomcat安全优化

Tomcat简介

- ❑ Tomcat是Apache软件基金会（Apache Software Foundation）的Jakarta 项目中的一个核心项目，由Apache、Sun和其他一些公司及个人共同开发而成。
- ❑ Tomcat服务器是一个免费的开放源代码的Web应用服务器，属于轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试JSP程序的首选。
- ❑ Tomcat和Nginx、Apache(httpd)、lighttpd等Web服务器一样，具有处理HTML页面的功能，另外它还是一个Servlet和JSP容器，独立的Servlet容器是Tomcat的默认模式。不过，Tomcat处理静态HTML的能力不如Nginx/Apache服务器。
- ❑ 目前Tomcat最新版本为9.0。Java容器还有resin、weblogic等。
- ❑ Tomcat官网：<http://tomcat.apache.org>

Tomcat简介

□ 安装Tomcat和JDK

JDK是 Java 语言的软件开发工具包，主要用于移动设备、嵌入式设备上的java应用程序。JDK是整个java开发的核心，它包含了JAVA的运行环境（JVM+Java系统类库）和JAVA工具。

□ 安装JDK

```
mkdir -p /usr/java
```

```
tar xf jdk-8u60-linux-x64.tar.gz -C /usr/java
```

配置环境变量：

```
vim /etc/profile.d/java.sh
```

```
JAVA_HOME=/usr/java/jdk1.8.0_60
```

```
PATH=$JAVA_HOME/bin:$PATH
```

```
CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
```

```
export JAVA_HOME CLASSPATH PATH
```

测试：

```
source /etc/profile.d/java.sh
```

```
java -version
```

Tomcat简介

□ 安装Tomcat

```
tar xf apache-tomcat-8.5.20.tar.gz -C /usr/local  
ln -s /usr/local/apache-tomcat-8.5.20 /usr/local/tomcat
```

配置环境变量:

```
echo 'export  
TOMCAT_HOME=/usr/local/tomcat'>>/etc/profile.d/tomcat.sh  
source /etc/profile.d/tomcat.sh
```

检查tomcat是否安装成功:

```
/usr/local/tomcat/bin/version.sh
```

附: 下载jdk(所有版本)

<http://www.oracle.com/technetwork/java/javase/archive-139210.html>

Tomcat目录结构

□ tomcat主目录介绍

```
[root@localhost ~]# cd /usr/local/tomcat/
```

```
[root@localhost tomcat]# tree -L 1
```

```
.
├── bin                #存放tomcat管理脚本
├── conf               # tomcat 配置文件存放目录
├── lib                # web应用调用的jar包存放路径
├── LICENSE
├── logs               # tomcat 日志存放目录, catalina.out 为
主要输出日志
├── NOTICE
├── RELEASE-NOTES
├── RUNNING.txt
├── temp               # 存放临时文件
├── webapps            # web程序存放目录
└── work               # 存放编译产生的.java 与 .class文件
```

7 directories, 4 files

Tomcat目录结构

□ webapps目录介绍

```
[root@localhost ~]# cd /usr/local/tomcat/webapps
```

```
[root@localhost webapps]# tree -L 1
```

```
.
├── docs                # tomcat 帮助文档
├── examples            # web应用实例
├── host-manager        # 主机管理
├── manager             # 管理
└── ROOT                # 默认站点根目录
```

5 directories, 0 files

Tomcat目录结构

□ Tomcat配置文件目录介绍

```
[root@localhost ~]# cd /usr/local/tomcat/conf
```

```
[root@localhost conf]# tree -L 1
```

```
.
├── Catalina
├── catalina.policy
├── catalina.properties
├── context.xml
├── logging.properties
├── logs
├── server.xml                # tomcat 主配置文件
├── server.xml.bak
├── server.xml.bak2
├── tomcat-users.xml         # tomcat 管理用户配置文件
├── tomcat-users.xsd
└── web.xml
```

2 directories, 10 files

Tomcat管理


□ Tomcat启动停止

启动程序/usr/local/tomcat/bin/startup.sh
关闭程序/usr/local/tomcat/bin/shutdown.sh


为tomcat提供服务启动脚本。
启动后访问页面：

[Home](#) [Documentation](#) [Configuration](#) [Examples](#) [Wiki](#) [Mailing Lists](#) [Find Help](#)

Apache Tomcat/8.5.20



If you're seeing this, you've successfully installed Tomcat. Congratulations!



Recommended Reading:

[Security Considerations HOW-TO](#)
[Manager Application HOW-TO](#)
[Clustering/Session Replication HOW-TO](#)

Server Status

Manager App

Host Manager

Developer Quick Start

[Tomcat Setup](#)
[First Web Application](#)

[Realms & AAA](#)
[JDBC DataSources](#)

[Examples](#)

[Servlet Specifications](#)
[Tomcat Versions](#)

Managing Tomcat

For security, access to the `manager.webapp` is restricted. Users are defined in:
`$CATALINA_HOME/conf/tomcat-users.xml`
In Tomcat 8.5 access to the manager application is split between different users.
[Read more...](#)

[Release Notes](#)
[Changelog](#)
[Migration Guide](#)
[Security Notices](#)

Documentation

[Tomcat 8.5 Documentation](#)
[Tomcat 8.5 Configuration](#)
[Tomcat Wiki](#)
Find additional important configuration information in:
`$CATALINA_HOME/RUNNING.txt`
Developers may be interested in:
[Tomcat 8.5 Bug Database](#)
[Tomcat 8.5 JavaDocs](#)
[Tomcat 8.5 SVN Repository](#)

Getting Help

[FAQ and Mailing Lists](#)
The following mailing lists are available:

[tomcat-announce](#)
Important announcements, releases, security vulnerability notifications. (Low volume).

[tomcat-users](#)
User support and discussion

[taglibs-user](#)
User support and discussion for [Apache Taglibs](#)

[tomcat-dev](#)
Development mailing list, including commit messages

Tomcat日志

□ Tomcat日志

```
[root@localhost ~]# tailf /usr/local/tomcat/logs/catalina.out
```

```
01-Nov-2019 11:42:59.445 信息 [main]
```

```
org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler  
["ajp-nio-8009"]
```

```
01-Nov-2019 11:42:59.448 信息 [main]
```

```
org.apache.catalina.startup.Catalina.start Server startup in 124398  
ms
```

发现启动时间较长，其中有一项的启动时间占据了绝大多数

```
01-Nov-2019 11:42:58.864 警告 [localhost-startStop-1]
```

```
org.apache.catalina.util.SessionIdGeneratorBase.c  
reateSecureRandom Creation of SecureRandom instance for  
session ID generation using [SHA1PRNG] took [123,292]  
milliseconds.
```

```
01-Nov-2019 11:42:58.897 信息 [localhost-startStop-1]
```

```
org.apache.catalina.startup.HostConfig.deployDire  
ctory Deployment of web application directory [/usr/local/apache-  
tomcat-8.5.20/webapps/ROOT] has finish  
ed in [123,793] ms
```

Tomcat日志

□ Tomcat日志--解决启动慢

发现耗时在这里：是session引起的随机数问题导致的。Tomcat的Session ID是通过SHA1算法计算得到的，计算Session ID的时候必须有一个密钥。为了提高安全性Tomcat在启动的时候会通过随机生成一个密钥。

查看是否有足够的熵来用于产生随机数

```
[root@localhost ~]# cat /proc/sys/kernel/random/entropy_avail  
187
```

推荐解决方法：

```
yum install rng-tools # 安装rngd服务（熵服务，增大熵池）  
systemctl start rngd # 启动服务  
systemctl enable rngd # 开机自启
```

再次查看：

```
[root@localhost ~]# cat /proc/sys/kernel/random/entropy_avail  
2980
```

重新启动tomcat后时间：

```
01-Nov-2019 11:54:06.032 信息 [main]  
org.apache.catalina.startup.Catalina.start Server startup in 1158 ms
```


Tomcat管理

□ Tomcat web管理功能

注意：不要在生产环境使用。

Tomcat管理功能用于对Tomcat自身以及部署在Tomcat上的应用进行管理的web应用。在默认情况下是处于禁用状态的。如果需要开启这个功能，就需要配置管理用户。

1>配置tomcat-users.xml 文件。

```
<role rolename="admin-gui"/>
<role rolename="admin-script"/>
<role rolename="manager-gui"/>
<role rolename="manager-jmx"/>
<role rolename="manager-script"/>
<role rolename="manager-status"/>
<user username="tomcat" password="tomcat" roles="admin-
gui,manager-gui,manager-jmx,manager-script,manager-status,admin-
script"/>
```

</tomcat-users> #前面加上以上几行，注意，不要添加到注释里面去。

Tomcat管理

□ Tomcat web管理功能

注意：不在再生产环境使用。

2>允许方式Manager App

```
vim /usr/local/tomcat/webapps/manager/META-INF/context.xml
```

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve"
```

```
    allow="^.*$" /> #修改红色部分
```

3>允许访问Host Manager

```
vim /usr/local/tomcat/webapps/host-manager/META-INF/context.xml
```

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve"
```

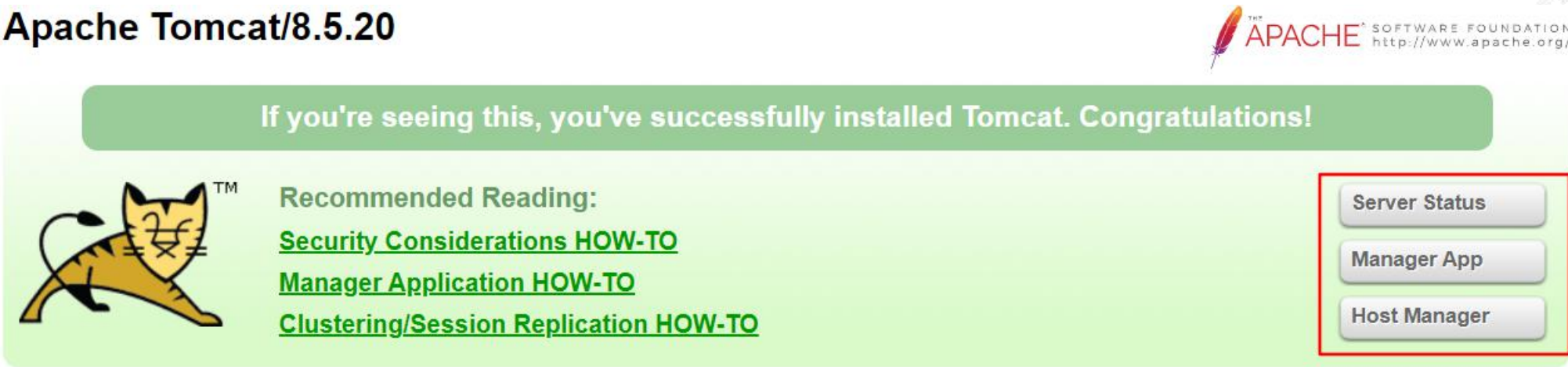
```
    allow="^.*$" />
```

```
    <!-- allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" / -->
```

修改完毕，重启tomcat。

Tomcat管理

- Tomcat web管理功能
- 访问管理页面



查看状态



Tomcat管理

Tomcat web管理功能 访问管理页面



Tomcat Web Application Manager

Message:	OK
----------	----

Manager			
List Applications	HTML Manager Help	Manager Help	Server Status

Applications					
Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Deploy	
Deploy directory or WAR file located on server	
Context Path (required): <input type="text"/>	
XML Configuration file URL: <input type="text"/>	

Tomcat配置文件

□ server.xml组件类别

顶级组件：位于整个配置的顶层，如server。

容器类组件：可以包含其它组件的组件，如service、engine、host、context。

连接器组件：连接用户请求至tomcat，如connector。

被嵌套类组件：位于一个容器当中，不能包含其他组件，如Valve、logger。

```
<server>
  <service>
    <connector />
    <engine>
      <host>
        <context></context>
      </host>
      <host>
        <context></context>
      </host>
    </engine>
  </service>
</server>
```

Tomcat配置文件

□ server.xml组件介绍

engine: 核心容器组件, catalina引擎, 负责通过connector接收用户请求, 并处理请求, 将请求转至对应的虚拟主机host。

host: 类似于httpd中的虚拟主机, 一般而言支持基于FQDN的虚拟主机。

context: 定义一个应用程序, 是一个最内层的容器类组件 (不能再嵌套)。配置context的主要目的指定对应对应的webapp的根目录, 类似于httpd的alias, 其还能为webapp指定额外的属性, 如部署方式等。

connector: 接收用户请求, 类似于httpd的listen配置监听端口的。

service (服务): 将connector关联至engine, 因此一个service内部可以有多个

connector, 但只能有一个引擎engine。service内部有两个connector, 一个engine。因此, 一般情况下一个server内部只有一个service, 一个service内部只有一个engine, 但一个service内部可以有多个connector。

server: 表示一个运行于JVM中的tomcat实例。

Valve: 阀门, 拦截请求并在将其转至对应的webapp前进行某种处理操作, 可以用于任何容器中, 比如记录日志(access log valve)、基于IP做访问控制(remote address filter valve)。

logger: 日志记录器, 用于记录组件内部的状态信息, 可以用于除context外的任何容器中。

realm: 可以用于任意容器类的组件中, 关联一个用户认证库, 实现认证和授权。可以关联的认证库有两种: UserDatabaseRealm、MemoryRealm和JDBCRealm。

UserDatabaseRealm: 使用JNDI自定义的用户认证库。

MemoryRealm: 认证信息定义在tomcat-users.xml中。

JDBCRealm: 认证信息定义在数据库中, 并通过JDBC连接至数据库中查找认证用户。

Tomcat配置文件

□ Connector主要参数说明

参数	参数说明
connector	接收用户请求，类似于httpd的listen配置监听端口。
port	指定服务器端要创建的端口号，并在这个端口监听来自客户端的请求。
address	指定连接器监听的地址，默认为所有地址（即0.0.0.0）
protocol	连接器使用的协议，支持HTTP和AJP。AJP（Apache Jserv Protocol）专用于tomcat与apache建立通信的，在httpd反向代理用户请求至tomcat时使用（可见Nginx反向代理时不可用AJP协议）。
minProcessors	服务器启动时创建的处理请求的线程数
maxProcessors	最大可以创建的处理请求的线程数
enableLookups	如果为true，则可以通过调用request.getRemoteHost()进行DNS查询来得到远程客户端的实际主机名，若为false则不进行DNS查询，而是返回其ip地址
redirectPort	指定服务器正在处理http请求时收到了一个SSL传输请求后重定向的端口号
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
connectionTimeout	指定超时的时间数(以毫秒为单位)

Tomcat配置文件

□ host参数详解

参数	参数说明
host	表示一个虚拟主机
name	指定主机名
appBase	应用程序基本目录，即存放应用程序的目录.一般为appBase="webapps"，相对于CATALINA_HOME而言的，也可以写绝对路径。
unpackWARs	如果为true，则tomcat会自动将WAR文件解压，否则不解压，直接从WAR文件中运行应用程序
autoDeploy	在tomcat启动时，是否自动部署。
xmlValidation	是否启动xml的校验功能，一般xmlValidation="false"。
xmlNamespaceAware	检测名称空间，一般xmlNamespaceAware="false"。

Tomcat配置文件

Context参数说明

参数	参数说明
Context	表示一个web应用程序，通常为WAR文件
docBase	应用程序的路径或者是WAR文件存放的路径,也可以使用相对路径，起始路径为此Context所属Host中appBase定义的路径。
path	表示此web应用程序的url的前缀，这样请求的url为http://localhost:8080/path/****
reloadable	这个属性非常重要，如果为true，则tomcat会自动检测应用程序的/WEB-INF/lib 和/WEB-INF/classes目录的变化，自动装载新的应用程序，可以在不重启tomcat的情况下改变应用程序

Web站点部署

□ 两种方式

第一种方式是直接将程序目录放在webapps目录下

第二种方式是使用开发工具将程序打包成war包，然后上传到webapps目录下。

□ 部署开源站点 (jpress)

jpress官网: <http://jpress.io>

下载地址: <https://github.com/JpressProjects/jpress>

Web站点部署

□ 安装配置数据库

```
yum -y install mariadb-server  
systemctl start mariadb.service
```

```
mysql -e "create database jpress DEFAULT CHARACTER  
SET utf8;"
```

```
mysql -e "grant all on jpress.* to jpress@'localhost'  
identified by '123456';"
```

□ jpress站点上线

```
# ls /usr/local/tomcat/webapps/jpress-web-newest.war  
/usr/local/tomcat/webapps/jpress-web-newest.war
```


Web站点部署

□ 浏览器访问

http://IP:8080/jpress-web-newest/install
安装完毕之后，重启tomcat即可访问。

访问后台：http://IP:8080/jpress-web-newest/admin/login

JPress安装向导

JPress安装第一步：安装必读。

欢迎使用JPress。在开始前，我们需要您数据库的一些信息。请准备好如下信息。

- 1、数据库名
- 2、数据库用户名
- 3、数据库密码
- 4、数据库主机
- 5、数据表前缀（table prefix，特别是当您要在一个数据库中安装多个JPress时）

我们会使用这些信息来创建一个名为db.properties的数据库信息文件。如果自动创建未能成功，不用担心，您可以在文本编辑器中打开db-sample.properties，填入数据库信息，并将其另存为db.properties。

绝大多数时候，您的网站服务商会给您这些信息。如果您没有这些信息，在继续安装JPress之前您将需要联系他们。如果您准备好了，那么，愉快的玩耍吧...

需要更多帮助？点击[这里](#)。

下一步

Tomcat虚拟主机

□ 修改server.xml

```
[root@localhost ~]# cd /usr/local/tomcat/conf/
```

```
[root@localhost conf]# cp server.xml{,.bak}
```

```
[root@localhost conf]# vim server.xml
```

增加虚拟主机配置:

```
<Host name="www.a.com" appBase="webapps"
    unpackWARs="true" autoDeploy="true">
    <Context docBase="/web/a" path="" reloadable="false" />
    <Valve className="org.apache.catalina.valves.AccessLogValve"
directory="logs"
        prefix="localhost_access_log" suffix=".txt"
        pattern="%h %l %u %t &quot;%r&quot; %s %b" />
    </Host>
<Host name="www.b.com" appBase="webapps"
    unpackWARs="true" autoDeploy="true">
    <Context docBase="/web/b" path="" reloadable="false" />
    <Valve className="org.apache.catalina.valves.AccessLogValve"
directory="logs"
        prefix="localhost_access_log" suffix=".txt"
        pattern="%h %l %u %t &quot;%r&quot; %s %b" />
    </Host>
```

Tomcat虚拟主机

❑ 虚拟主机目录及文件准备

```
mkdir -p /web/{a,b}
```

```
vim /web/a/index.jsp
```

```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
```

```
<html>
```

```
    <head>
```

```
        <title>JSP a page</title>
```

```
    </head>
```

```
    <body>
```

```
        <% out.println("Welocome to test site,http://www.a.com");%>
```

```
    </body>
```

```
</html>
```

```
vim /web/b/index.jsp
```

```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
```

```
<html>
```

```
    <head>
```

```
        <title>JSP a page</title>
```

```
    </head>
```

```
    <body>
```

```
        <% out.println("Welocome to test site,http://www.b.com");%>
```

```
    </body>
```

```
</html>
```

Tomcat虚拟主机

❑ 重启tomcat

```
[root@localhost ~]# systemctl restart tomcat
```

❑ 客户端测试

主机名解析，hosts文件添加：

```
tomcat_ip www.a.com www.b.com
```

测试：

← → ↻ ⓘ 不安全 | a.com:8080

Welocome to test site,http://www.a.com

← → ↻ ⓘ 不安全 | b.com:8080

Welocome to test site,http://www.b.com

Tomcat监控

□ 方法一：开发java监控页面

```
# cat /usr/local/tomcat/webapps/ROOT/meminfo.jsp
<%
```

```
Runtime rtm = Runtime.getRuntime();
long mm = rtm.maxMemory()/1024/1024;
long tm = rtm.totalMemory()/1024/1024;
long fm = rtm.freeMemory()/1024/1024;
```

```
out.println("JVM memory detail info :<br>");
out.println("Max memory:"+mm+"MB"<br>");
out.println("Total memory:"+tm+"MB"<br>");
out.println("Free memory:"+fm+"MB"<br>");
out.println("Available memory can be used is :"+(mm+fm-
tm)+"MB"<br>");
%>
```

JVM memory detail info :

Max memory:237MB

Total memory:38MB

Free memory:8MB

Available memory can be used is :207MB

Tomcat监控

□ 方法二：使用jps命令进行监控

```
[root@localhost ~]# jps -lvm
```

```
3075 sun.tools.jps.Jps -lvm -
```

```
Denv.class.path=./usr/java/jdk1.8.0_60/lib/dt.jar:/usr/java/jdk1.8.0_60  
/lib/tools.jar -Dapplication.home=/usr/java/jdk1.8.0_60 -Xms8m
```

```
2980 org.apache.catalina.startup.Bootstrap start -
```

```
Djava.util.logging.config.file=/usr/local/tomcat/conf/logging.properties
```

```
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
```

```
-Djdk.tls.ephemeralDHKeySize=2048 -
```

```
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
```

```
Dcatalina.base=/usr/local/tomcat -Dcatalina.home=/usr/local/tomcat
```

```
-Djava.io.tmpdir=/usr/local/tomcat/temp
```

Tomcat监控

□ Tomcat远程监控功能

修改配置文件，开启远程监控

```
vim /usr/local/tomcat/bin/catalina.sh
```

```
CATALINA_OPTS="$CATALINA_OPTS  
-Dcom.sun.management.jmxremote  
-Dcom.sun.management.jmxremote.port=12345  
-Dcom.sun.management.jmxremote.authenticate=false  
-Dcom.sun.management.jmxremote.ssl=false  
-Djava.rmi.server.hostname=IP"
```

windows安装jdk环境，运行jconsole.exe或jvisualvm.exe进行监控

Tomcat安全优化

telnet管理端口保护（强制）

类别	配置内容及说明	标准配置	备注
telnet管理 端口保护	1.修改默认的8005管理端口为不易猜测的端口（大于1024）； 2.修改SHUTDOWN指令为其他字符串；	<Server port="8527" shutdown="dangerous">	1.以上配置项的配置内容只是建议配置，可以按照服务实际情况进行合理配置，但要求端口配置在8000~8999之间；

Tomcat安全优化

□ ajp连接端口保护（推荐）

类别	配置内容及说明	标准配置	备注
Ajp 连接端口保护	<p>1.修改默认的ajp 8009端口为不易冲突的大于1024端口；</p> <p>2.通过iptables规则限制ajp端口访问的权限仅为线上机器；</p>	<pre><Connector port="8528" protocol="AJP/1.3" /></pre>	<p>以上配置项的配置内容仅为建议配置，请按照服务实际情况进行合理配置，但要求端口配置在8000~8999之间；</p> <p>保护此端口的目的在于防止线下的测试流量被mod_jk转发至线上tomcat服务器；</p>

Tomcat安全优化

禁用管理端（强制）

类别	配置内容及说明	标准配置	备注
禁用管理端	<p>1. 删除默认的 {Tomcat安装目录}/conf/tomcat-users.xml文件，重启tomcat后将会自动生成新的文件；</p> <p>2. 删除{Tomcat安装目录}/webapps下默认的所有目录和文件；</p> <p>3.将tomcat 应用根目录配置为tomcat安装目录以外的目录；</p>	<pre><Context path="" docBase="/home/work/local/tomcat_webapps" debug="0" reloadable="false" crossContext="true"/></pre>	对于前段web模块，Tomcat管理端属于tomcat的高危安全隐患，一旦被攻破，黑客通过上传web shell的方式将会直接取得服务器的控制权，后果极其严重；

Tomcat安全优化

□ 降权启动

类别	配置内容及说明	标准配置	备注
降权启动	<p>1.tomcat启动用户权限必须为非root权限，尽量降低tomcat启动用户的目录访问权限；</p> <p>2.如需直接对外使用80端口，可通过普通账号启动后，配置iptables规则进行转发；</p>		避免一旦tomcat服务被入侵，黑客直接获取高级用户权限危害整个server的安全；

Tomcat安全优化

□ 文件列表访问控制

类别	配置内容及说明	标准配置	备注
文件列表访问控制	1.conf/web.xml文件中default部分listings的配置必须为false;	<pre><init-param> <param- name>listings</p aram-name> <param- value>false</para m-value> </init-param></pre>	false为不列出目录文件，true为允许列出，默认为false;

Tomcat安全优化

□ 降权启动

类别	配置内容及说明	标准配置	备注
禁用管理端	<p>1.tomcat启动用户权限必须为非root权限，尽量降低tomcat启动用户的目录访问权限；</p> <p>2.如需直接对外使用80端口，可通过普通账号启动后，配置iptables规则进行转发；</p>		避免一旦tomcat服务被入侵，黑客直接获取高级用户权限危害整个server的安全；

Tomcat安全优化

□ 隐藏版本信息

类别	配置内容及说明	标准配置	备注
版本信息隐藏	<p>1.修改 conf/web.xml, 重定向403、404以及500等错误到指定的错误页面;</p> <p>2.也可以通过修改应用程序目录下的 WEB-INF/web.xml 下的配置进行错误页面的重定向;</p>	<pre><error-page> <error-code>403</error-code> <location>/forbidden.jsp</location> </error-page> <error-page> <error-code>404</error-code> <location>/notfound.jsp</location> </error-page> <error-page> <error-code>500</error-code> <location>/systembusy.jsp</location> </error-page></pre>	<p>在配置中对一些常见错误进行重定向, 避免当出现错误时tomcat默认显示的错误页面暴露服务器和版本信息;</p> <p>必须确保程序根目录下的错误页面已经存在;</p>

Tomcat性能优化

□ 增加tomcat处理线程数

编辑Tomcat目录下面的conf子目录下面的server.xml文件

maxThreads :Tomcat使用线程来处理接收的每个请求。这个值表示Tomcat可创建的最大的线程数。

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。

connectionTimeout:网络连接超时，单位：毫秒。设置为0表示永不超时，这样设置有隐患的。通常可设置为30000毫秒。

minSpareThreads:Tomcat初始化时创建的线程数。

maxSpareThreads:一旦创建的线程超过这个值，Tomcat就会关闭不再需要的socket线程。

配置示例：

```
<Connector port="8080" maxHttpHeaderSize="8192"
  maxThreads="150"      minSpareThreads="30"
  maxSpareThreads="75"
  enableLookups="false"      redirectPort="8443"
  acceptCount="100" c      disableUploadTimeout="true"
/>
```

Tomcat性能优化

□ tomcat内存空间设置

常见内存错误:

1、java.lang.OutOfMemoryError: Java heap space

整体意思是超出内存堆空间的错误

使用Java程序从数据库中查询大量的数据时出现异常

在JVM中如果98%的时间是用于GC(Garbage Collection)且可用的Heap size 不足2%的时候将抛出此异常信息。

2、java.lang.OutOfMemoryError: PermGen space

PermGen space的全称是Permanent Generation space,是指内存的永久保存区域,这块内存主要是被JVM存放Class和Meta信息的。

如果你的WEB APP下都用了大量的第三方jar, 其大小超过了jvm默认的大小(4M)那么就会产生此错误信息了。

Tomcat性能优化

□ tomcat内存空间设置

常见内存错误:

3、OutOfMemoryError: unable to create new native thread.

无法创建新的线程。

4、java "Too small initial heap" 错误
Xmx设置小了。

具体解决方法:

#生产环境设置, 假设内存空间为8G, 在catalina.sh在文件头部的注释下面, 加入下面的内容

```
export JAVA_OPTS='-server -Xms4096m -Xmx4096m -  
Xmn256m -XX:PermSize=256m -XX:MaxNewSize=512m  
-XX:MaxPermSize=512m '
```

提示:

使用64位tomcat版本。

总结

- Tomcat简介
- Tomcat目录结构
- Tomcat配置文件
- Tomcat多实例
- Tomcat安全优化

作业

- 安装部署Tomcat，并部署zrlog.war。



谢谢观看

更多好课，请关注[万门大学APP](#)

