Масштабируемая отказоустойчивая архитектура на основе серверов Apache HTTPD и Tomcat

Оглавление

[Масштабируемая отказоустойчивая архитектура на основе серверов Apache HTTPD и Tomcat 1](#_Toc452641068)

[1. Термины и сокращения 3](#_Toc452641069)

[2. Введение 3](#_Toc452641070)

[3. Установка HTTPD 4](#_Toc452641071)

[3.1 Проверка работы 4](#_Toc452641072)

[3.2 Создание Windows-сервиса для HTTPD 5](#_Toc452641073)

[3.3 Удаление Windows-сервиса HTTPD 5](#_Toc452641074)

[4. Установка Tomcat 5](#_Toc452641075)

[4.1 Настройки Tomcat 5](#_Toc452641076)

[4.1.1 Базовые настройки 5](#_Toc452641077)

[4.2 Создание Windows-сервиса для Tomcat 6](#_Toc452641078)

[4.3 Удаление Windows-сервиса Tomcat 8](#_Toc452641079)

[5. Настройка взаимодействия HTTPD-Tomcat 8](#_Toc452641080)

[5.1 Настройка на стороне HTTPD 9](#_Toc452641081)

[5.1.1 Коннектор mod\_jk 9](#_Toc452641082)

[5.2 Настройка на стороне Tomcat 9](#_Toc452641083)

[5.2.1 Настройка server.xml 10](#_Toc452641084)

[5.2.2 Коннектор tomcat-jk2 10](#_Toc452641085)

[5.3 Проверка работоспособности 10](#_Toc452641086)

[6. Настройки для приложений 10](#_Toc452641087)

[7. Установка приложений JepRia на Tomcat 12](#_Toc452641088)

[7.1 Автоматизированная установка приложений с помощью Ant 12](#_Toc452641089)

[7.2 Используемые библиотеки 12](#_Toc452641090)

[7.3 Используемые переменные 12](#_Toc452641091)

[7.4 Создание администратора Tomcat 12](#_Toc452641092)

[8. Конфигурирование SSO в экземплярах Tomcat 13](#_Toc452641093)

[*8.1* Включение режима SSO в экземпляре Tomcat 13](#_Toc452641094)

[8.2 Настройки Windows-сервиса Tomcat 13](#_Toc452641095)

[9. Дистрибутивы, библиотеки 14](#_Toc452641096)

[9.1 Дистрибутив Apache Tomcat 14](#_Toc452641097)

[9.2 Коннектор Tomcat-JK2 14](#_Toc452641098)

[9.3 Дистрибутив Apache HTTPD 14](#_Toc452641099)

[9.4 Коннектор MOD\_JK 14](#_Toc452641100)

[10. Приложение 1. Настройка HTTPD. 14](#_Toc452641101)

[10.1 Настройка кэширования 14](#_Toc452641102)

[10.2 Настройки, связанные с безопасностью сервера приложений 15](#_Toc452641103)

[10.2.1 Скрытие названия и версии сервера приложений 15](#_Toc452641104)

[11. Приложение 3.Пример настройки файла *server.xml* в Tomcat. 15](#_Toc452641105)

[12. Приложение 4.Пример настройки файла mod\_jk.conf 18](#_Toc452641106)

[13. Приложение 5 19](#_Toc452641107)

[14. Ссылки 20](#_Toc452641108)

# Термины и сокращения

*HTTPD (Hyper Text Transfer Protocol Daemon)* – Apache HTTP Server;

*Вертикальное масштабирование* – увеличение производительности системы при увеличении ресурсов вычислительного узла;

*Горизонтальное масштабирование* – увеличение производительности системы при увеличении количества вычислительных узлов;

*Reverse Proxy (Обратный прокси)* - тип прокси-сервера, который ретранслирует запросы клиентов из внешней сети на один или несколько серверов, логически расположенных во внутренней сети. При этом для клиента это выглядит так, будто запрашиваемые ресурсы находятся непосредственно на прокси-сервере.

*Realm* представляет собой «базу данных» пользовательских имён, паролей и ролей, используемых системами безопасности.

# Введение

Документ содержит основные положения по настройке системы серверов Apache HTTP Server (HTTPD) + Tomcat.



Рисунок 1.Общая схема взаимодействия серверов Apache HTTPD, Tomcat и СУБД

Архитектура предполагает простую и удобную возможность вертикального и горизонтального масштабирования функционала приложений, разрабатываемых в стандарте J2EE, при увеличении интенсивности запросов к системе и (или) увеличении одновременно работающего функционала.

Apache HTTPD выполняет функции:

* Reverse proxy для экземляров серверов Tomcat, обеспечивая обработку внешних запросов по стандартному порту 80;
* Балансировку нагрузки на экземпляры серверов Tomcat;
* Эффективный доступ к статическим ресурсам
* TODO…

# Установка HTTPD

Используемый дистрибутив HTTPD [см. Дистрибутив Apache HTTPD] предназначен для работы в Windows.

Для установки его достаточно распаковать в одну из директорий, например, *с:\Apache24*.

Запуск сервера из командной строки выполняется командой:

>*httpd.exe*

## Проверка работы

При вводе в адресную строку браузера <http://localhost> он должен выдать такую страницу:



## Создание Windows-сервиса для HTTPD

Windows-сервис используемой версии HTTPD создаётся командой

***httpd -k install***,

которая запускается из директории *%APACHE\_HOME%/bin/*

## Удаление Windows-сервиса HTTPD

Для удаления сервиса Windows-сервиса HTTPD используется команда

***httpd -k uninstall***

# Установка Tomcat

Используемый дистрибутив Tomcat [см.Дистрибутив Tomcat] ориентирован на работу в 64-разрядной Windows, для её работы необходима соответствующая 64-разрядная версия Java 1.7+.

## Настройки Tomcat

Настройки Tomcat целесообразно разделить на базовые настройки, настройки для приложений и настройки взаимодействия с HTTPD (вынесены в раздел ).

### Базовые настройки

Базовые настройки включают настройки внешних переменных, настройки источников данных и пулов соединений.

#### Внешние переменные

Внешние переменные для Tomcat определяются в файле (его нужно создать) *%TOMCAT\_HOME%/bin/setEnv.bat* (здесь и далее используется нотация Windows), который автоматически вызывается в самом начале работы*.*

Для Tomcat используемой версии необходима версия Java 1.7+. Для неё в ***setEnv.bat*** задаётся переменная *JAVA\_HOME*, а директория *%JAVA\_HOME%/bin* прописывается в переменной *PATH*, например:

*set JAVA\_HOME=D:/Program Files/Java/jdk1.8.0\_45*

*set PATH=%JAVA\_HOME%/bin;% PATH%*

#### Проверка работы

После выполнения базовых настроек Tomcat можно проверить его работоспособность.

Запуск Tomcat из командной строки выполняется из директории *%TOMCAT\_HOME%/bin* командой

>*startup.*

После запуска в DOS-окно запуска должно выглядеть примерно так:



При вводе в адресную строку браузера <http://localhost:8080> (8080 – порт по умолчанию) он должен выдать такую страницу:



## Создание Windows-сервиса для Tomcat

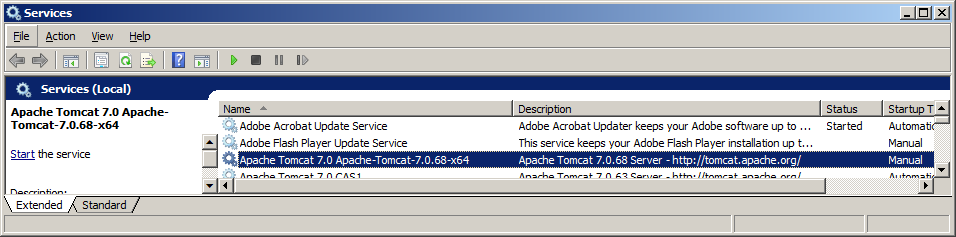
Для того, чтобы создать Windows-сервис для Tomcat, например, с именем *Tomcat\_1*, нужно перейти в директорию *<TomcatHome>/bin*,открыть в нём консольное окно и последовательно выполнить команды:

*set java\_home=<TomcatJavaHome>*

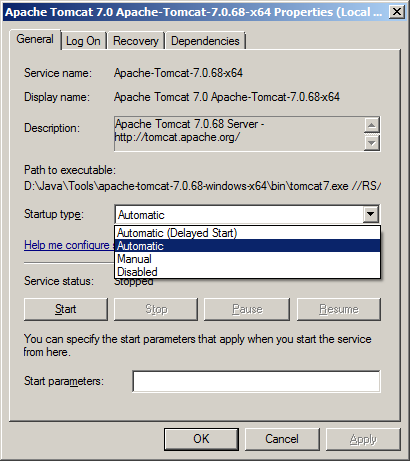
*service install Tomcat\_1*



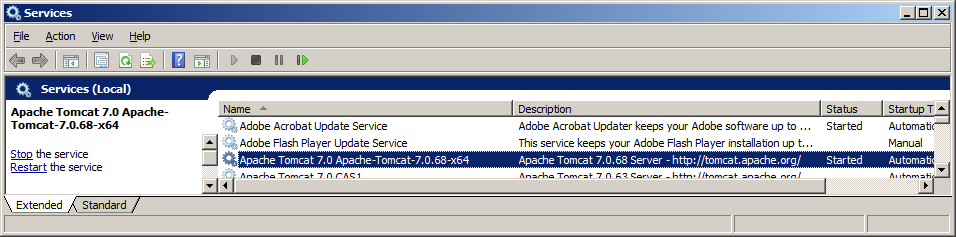
После этого в окне *Services* в списке сервисов должен появиться элемент с названием *Apache Tomcat ~~7.0 CAS~~* ~~(в некоторых версиях - просто~~ *~~CAS~~*~~)~~:



Далее необходимо настроить сервис на автоматический запуск:



После запуска сервиса в окне *Services* сервис должен иметь вид:



## Удаление Windows-сервиса Tomcat

Для удаления сервиса Tomcat используется команда

*service uninstall <serviceName>*

# Настройка взаимодействия HTTPD-Tomcat

Настройка взаимодействия HTTPD-Tomcat включает настройку на стороне HTTPD и настройку на стороне Tomcat.

## Настройка на стороне HTTPD

За настройку взаимодействия с серверами Tomcat на стороне HTTPD отвечают файлы *workers.properties* и *mod\_jk.conf*, расположенные в директории *%APACHE\_HOME%/conf/jk*.

Файл *workers.properties* описывает параметры экземпляров Tomcat, взаимодействующих с HTTPD. Параметры *port* и *host* каждого worker-а должны совпадать с соответствующими настройками Tomcat в *server.xml*. Ниже приводится пример файла *workers.properties*, в котором определяется два worker-а – ***worker1***и ***worker2***:

*# BEGIN workers.properties*

*worker.list=worker1,worker2*

*worker.worker1.port=8009*

*worker.worker1.host=localhost*

*worker.worker1.type=ajp13*

*worker.worker2.port=8109*

*worker.worker2.host=localhost*

*worker.worker2.type=ajp13*

*# END workers.properties*

Главное назначение файла *mod\_jk.conf* состоит в назначении группам ресурсов соответствующих им worker-ов.

Настройка сервера HTTPD для взаимодействия с Tomcat включает:

* Настройку файла *workers.properties*
* Настройку файла *mod\_jk.conf*
* Настройку файла *%APACHE\_HOME%/conf/httpd.conf*

Файл *mod\_jk.conf* создаётся в ручном режиме и размещается в директории *%APACHE\_HOME%/conf/jk*  (предлагаемый режим, когда *mod\_jk.conf* автоматически создаётся при запуске Tomcat, работает некорректно).

В п.*Приложение 4.Пример настройки файла mod\_jk.conf* приведён пример **ручной** настройки, обеспечивающей работу с встроенными примерами приложений Tomcat)

В конце файла *httpd.conf* необходимо подключить файл *mod\_jk.conf*:

…

*# Include mod\_jk's specific configuration file*

*Include ~~%APACHE\_HOME%/~~conf/jk/mod\_jk.conf*

…

При необходимости для каждого из экземпляров Tomcat могут быть выставлены дополнительные настройки.

???

### Коннектор mod\_jk

mod\_jk – коннектор, поддерживающий взаимодействие Apache HTTPD-Tomcat по протоколу AJP со стороны HTTPD.

Она добавляется в директорию %APACHE\_HOME%/modules.

Скачать можно здесь [см.Коннектор MOD\_JK].

## Настройка на стороне Tomcat

Настройки взаимодействия Tomcat с HTTPD включают настройки *server.xml* и настройки коннектора

Apache HTTPD-Tomcat.

### Настройка server.xml

См.пример настройки в

### Коннектор tomcat-jk2

Tomcat-jk2 – библиотека (jar), реализующая коннектор, поддерживающий взаимодействие Apache HTTPD-Tomcat по протоколу AJP со стороны Tomcat.

Она добавляется в директорию %TOMCAT\_HOME%/lib.

Скачать можно здесь [см. Библиотека Tomcat-JK2].

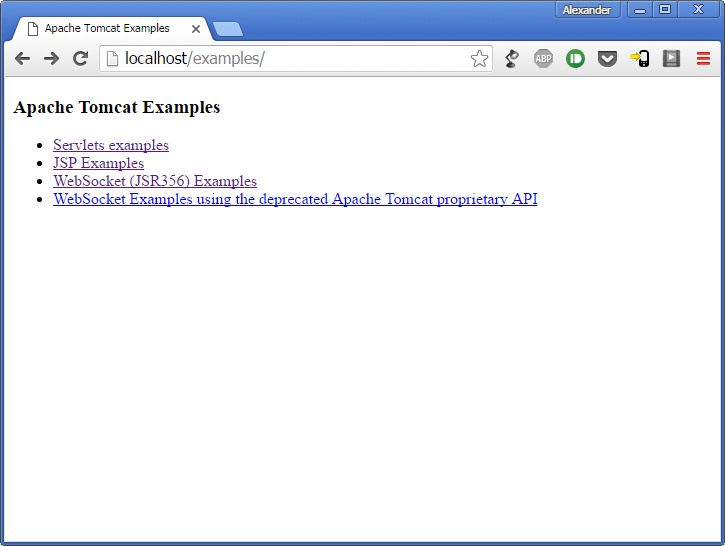
???

## Проверка работоспособности

После того, как настройки на обеих сторонах выполнены, для проверки работоспособности нужно:

1. Перезапустить сервис Tomcat
2. Перезапустить сервис HTTPD
3. В адресной строке браузера набрать http:// <http://localhost/examples>,

После этого в окне браузера должна появиться страница:



# Настройки для приложений

Настройки для приложений включают настройки для источников данных и пулов соединений.

#### Настройка источников данных и пулов соединений

Настройка пулов соединений для источников данных Tomcat выполняется в файле *%TOMCAT\_HOME%/conf/context.xml*.

~~В общем случае для каждого realm-а используется отдельный источник данных, работающий в отдельной схеме БД~~ Ниже приводится пример настройки пула соединений для источника данных: *jdbc/JEPDS* , настроенного на Oracle:

…

*<Context>*

*…*

*<Resource*

*name="jdbc/ JEPDS "*

*auth="Container"*

*type="javax.sql.DataSource"*

*driverClassName="oracle.jdbc.OracleDriver"*

*url="jdbc:oracle:thin:@//host:1521/DB1"*

*username="dbschema1"*

*password=" welcome "*

*/>*

*…*

*<Context>*, где

* *auth* – способ управления безопасностью – в данном случае – Container Managed Security;
* *~~maxActive~~* ~~(int) – Максимальное число активных соединений одновременно взятых из пула. По умолчанию - 100;~~
* *~~maxIdle~~* ~~(int) – Максимальное число соединений, находящихся в пуле;~~
* *~~maxWait~~* ~~(int) – Максимальное число миллисекунд, в течение которых пул будет ожидать освобождения соединения перед тем, как выкинуть exception. Умолчательное значение - 30000 (30 секунд)~~
* *~~jmxEnabled~~* ~~(boolean) Указывает необходимость поддержки пулом JMX. По умолчанию - true.~~

#### Встраивание собственного Login-модуля

Для встраивания в Tomcat кастомизированного Login-модуля необходимо создать два конфигурационных файла: файл определения Realm и файл JAAS Login.

##### Файл определения Realm

Определение Realm находится в war-файле каждого приложения в директории *META-INF/сontext.xml* (не путать с файлом *%TOMCAT\_HOME%/conf/context.xml*) и имеет следующий вид:

*<?xml version="1.0" encoding="UTF-8"?>*

*<Context>*

*<Realm className="org.apache.catalina.realm.JAASRealm"*

*appName="ITMDS"*

*userClassNames="com.technology.jep.jepcommon.security.JepPrincipal"*

*roleClassNames="com.technology.jep.jepcommon.security.JepPrincipal" />*

*</Context>*, где

пользовательский и ролевой приципалы определяются классом

*com.technology.jep.jepcommon.security.JepPrincipal*.

##### Файл определения Login-модуля

##### Конфигурационный файл JAAS Login *jaas.conf* размещается в директории *%CATALINA\_BASE%/conf* и имеет вид:

*/\*\**

*\* Login Configuration for JAAS*

*\*/*

*ITMDS {*

*com.technology.jep.jepcommon.security.tomcat.JepLoginModule sufficient dataSource="jdbc/ITMDS";*

*};*

, где

встраиваиваемый Login-модуль определяется классом

*com.technology.jep.jepcommon.security.tomcat.JepLoginModule*.

# Установка приложений JepRia на Tomcat

Web-приложения, оформленные war-файлом или структурированные согласно стандарту J2EE, устанавливаются на Tomcat простым копированием в директорию *%TOMCAT\_HOME%/webapps*.

## Автоматизированная установка приложений с помощью Ant

Скрипт сборки приложений JepRia – *build.xml* – позволяет выполнять на Tomcat автоматизированную установку/удаление (deploy/undeploy) приложений средствами Ant при помощи целей (targets) *tomcat.deploy* и *tomcat.undeploy*.

## Используемые библиотеки

При установке/удалении приложений используются библиотеки:

*%TOMCAT\_HOME%/lib/catalina-ant.jar*

*%TOMCAT\_HOME%/lib/tomcat-coyote.jar*

*%TOMCAT\_HOME%/lib/tomcat-util.jar*

В процессе настройки автоматизированной установки необходимо убедиться, чтобы директория *%ANT\_HOME%/lib* не содержала иных версий этих библиотек.

## Используемые переменные

В файле *deploy.properties* необходимо определить переменные *DEPLOYMENT\_PATH, LOGIN, PASSWORD*, например:

…

*# Параметры установки для локального Tomcat*

*DEPLOYMENT\_PATH=http://localhost:8080/manager/text*

*LOGIN=admin*

*PASSWORD=admin*

…

Кроме этого должны быть определены переменные *TOMCAT\_HOME* и *JDBC\_HOME*, например:

*# Директория установки Tomcat*

*TOMCAT\_HOME=D:/Java/Tools/apache-tomcat-7.0.68-windows-x64*

*# Директория размещения библиотек JDBC*

*JDBC\_HOME=D:/Java/Tools/BinRepo/com/oracle*

## Создание администратора Tomcat

Автоматизированная установка выполняется от имени специального пользователя - администратора Tomcat, которого необходимо создать, определив его в файле *%TOMCAT\_HOME%/config/tomcat-users.xml* следующим образом (пароль, разумеется, вариабелен):

*<?xml version='1.0' encoding='utf-8'?>*

*<tomcat-users>*

*<role rolename="manager-gui"/>*

*<role rolename="manager-script"/>*

*<role rolename="manager-jmx"/>*

*<role rolename="manager-status"/>*

*<user username="admin" password="admin" roles="manager-gui,manager-script,manager-jmx,manager-status"/>*

*</tomcat-users>*

# Конфигурирование SSO в экземплярах Tomcat

Согласно инструкции Tomcat в режиме работы SSO один экземпляр Tomcat может поддерживать только один Realm (TODO в то же время, возможно, стоит проверить, может ли один Realm размещаться на нескольких экземплярах Tomcat).

Настоящий документ ориентирован на конфигурацию работы с несколькими Realm-ами, каждый из которых обслуживается отдельным экземпляром Tomcat.

## Включение режима SSO в экземпляре Tomcat

Режим SSO в экземпляре Tomcat включается расскомментариванием в файле *server.xml* (см.п.11)

строки

*<Valve className="org.apache.catalina.authenticator.SingleSignOn" />*

## Настройки Windows-сервиса Tomcat

Настройки Windows-сервиса Tomcat выполняются в скрипте установки сервиса *service.bat*.

Каждый экземпляр Tomcat устанавливается как сервис Windows, как выше было сказано командой

*service install <TomcatServiceName>*

При этом перед установкой каждого сервиса в скрипт *service.bat* вносится два изменения:

* Переменной *%JAVA\_HOME%* присваивается значение пути того JDK, на котором будет выполняться сервис,
* В опцию *–JvmOptions* добавляется пара параметров:

*-Dorg.apache.catalina.authenticator.Constants.SSO\_SESSION\_COOKIE\_NAME= REALM\_ID\_JSESSIONIDSSO* и

*-Djava.security.auth.login.config=%CATALINA\_BASE%/conf/jaas.conf*

*REALM\_ID\_JSESSIONIDSSO –* идентификатор SSO-сессии для Realm-а устанавливаемого сервиса Tomcat (устанавливается в cookie браузера при входе в первое из приложений Realm-а).Здесь *REALM\_ID* – идентификатор Realm-а, поддерживаемого устанавливаемым сервисом Tomcat.

Например:

*…*

*rem Make sure prerequisite environment variables are set*

***set JAVA\_HOME=D:/Program Files/Java/jdk1.8.0\_45***

*…*

*"%EXECUTABLE%" //IS//%SERVICE\_NAME% ^*

*--Description "Apache Tomcat 7.0.68 Server - http://tomcat.apache.org/" ^*

*--DisplayName "%DISPLAYNAME%" ^*

*--Install "%EXECUTABLE%" ^*

*--LogPath "%CATALINA\_BASE%\logs" ^*

*--StdOutput auto ^*

*--StdError auto ^*

*--Classpath "%CLASSPATH%" ^*

*--Jvm "%JVM%" ^*

*--StartMode jvm ^*

*--StopMode jvm ^*

*--StartPath "%CATALINA\_HOME%" ^*

*--StopPath "%CATALINA\_HOME%" ^*

*--StartClass org.apache.catalina.startup.Bootstrap ^*

*--StopClass org.apache.catalina.startup.Bootstrap ^*

*--StartParams start ^*

*--StopParams stop ^*

*--JvmOptions "****-Dorg.apache.catalina.authenticator.Constants.SSO\_SESSION\_COOKIE\_NAME=REALM\_ID\_JSESSIONIDSSO;-Djava.security.auth.login.config=%CATALINA\_BASE%/conf/jaas.conf****;-Dcatalina.home=%CATALINA\_HOME%;-Dcatalina.base=%CATALINA\_BASE%;-Djava.endorsed.dirs=%CATALINA\_HOME%\endorsed;-Djava.io.tmpdir=%CATALINA\_BASE%\temp;-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager;-Djava.util.logging.config.file=%CATALINA\_BASE%\conf\logging.properties" ^*

*--JvmMs 128 ^*

*--JvmMx 256*

*…*

# Дистрибутивы, библиотеки

В состав дистрибутивов системы входят:

* Дистрибутив *Apache Tomcat*
* Библиотека-коннектор *Tomcat\_JK2*
* Дистрибутив Apache HTTPD
* Коннектор *mod\_jk*

## Дистрибутив Apache Tomcat

[apache-tomcat-7.0.68-windows-x64](http://apache-mirror.rbc.ru/pub/apache/tomcat/tomcat-7/v7.0.68/bin/apache-tomcat-7.0.68-windows-x64.zip)

## Коннектор Tomcat-JK2

Скачивается отсюда: [tomcat-jk2-5.0.28.jar](http://www.java2s.com/Code/JarDownload/tomcat-jk2/tomcat-jk2-5.0.18.jar.zip)

## Дистрибутив Apache HTTPD

[Дистрибутив httpd-2.2.31-x86-r3.zip](http://de.apachehaus.com/downloads/httpd-2.2.31-x86-r3.zip)

## Коннектор MOD\_JK

Скачивается отсюда: [tomcat-connectors-1.2.40-windows-i386-httpd-2.2.x.zip](http://apache-mirror.rbc.ru/pub/apach...s-1.2.40-windows-i386-httpd-2.2.x.zip)

# Приложение 1. Настройка HTTPD.

## Настройка кэширования

В приложении приведён перечень общих умолчательных настроек Apache HTTPD.

TODO Здесь приведено описание для Apache OC4J. Актуализировать

Для корректного кэширования и гарантированного обновления клиентских частей приложений при установке приложений на Tomcat необходимо установить следующие настройки кэширования в конфигурационном файле *httpd.conf* :

После правила запрета доступа к файлам, начинающихся с .ht:

*<Files ~ "^\.ht">*

*Order allow,deny*

*Deny from all*

*</Files>*

Добавить правила:

*#*

*# Accurate caching of JepRia applications.*

*#*

*<Files \*.nocache.\*>*

*ExpiresActive on*

*ExpiresDefault "now"*

*Header* ***merge*** *Cache-Control "public, max-age=0, must-revalidate"*

*</Files>*

*<Files \*.cache.\*>*

*ExpiresActive on*

*ExpiresDefault "now plus 1 year"*

*</Files>*

Полезные ссылки по настройке кэширования для GWT-приложений:

<http://www.gwtproject.org/doc/latest/DevGuideCompilingAndDebugging.html#perfect_caching>

<http://seewah.blogspot.ru/2009/02/gwt-tips-2-nocachejs-getting-cached-in.html>

## Настройки, связанные с безопасностью сервера приложений

### Скрытие названия и версии сервера приложений

TODO Здесь приведено описание для Apache OC4J. Актуализировать.

Для того, чтобы скрыть версию и наименование сервера приложения, необходимо выполнить следующие действия в конфигурационном файле httpd.conf:

* 1. **Загрузить модуль mod\_security:**

В секции Dynamic Shared Object (DSO) Support прописать:

LoadModule security\_module  modules/ApacheModuleSecurity.dll

В секции добавления модулей (ClearModuleList):

AddModule mod\_security.c

* 1. **Установить значение опции ServerSignature:**

ServerSignature Off

* 1. **Установить значение опции SecServerSignature**

SecServerSignature “Unknown”

*Примечание 1*: Вместо “Unknown” может быть любое не пустое значение – фактически данное значение будет возвращаться в заголовке ответа “Server”

*Примечание 2:*  В том случае, если значение SecurityTokens по какой-то причине перекрыто, необходимо установить его в Full

# Приложение 3.Пример настройки файла *server.xml* в Tomcat.

Ниже приводится пример файла server.xml Tomcat, настроенного на взаимодействие с Apache HTTPD. Изменённые/добавленные фрагменты выделены ***жирным шрифтом (одно изменение по кодировке)***:

*~~<?xml version='1.0' encoding='utf-8'?>~~*

*~~<!--~~*

*~~Licensed to the Apache Software Foundation (ASF) under one or more~~*

*~~contributor license agreements. See the NOTICE file distributed with~~*

*~~this work for additional information regarding copyright ownership.~~*

*~~The ASF licenses this file to You under the Apache License, Version 2.0~~*

*~~(the "License"); you may not use this file except in compliance with~~*

*~~the License. You may obtain a copy of the License at~~*

*~~http://www.apache.org/licenses/LICENSE-2.0~~*

*~~Unless required by applicable law or agreed to in writing, software~~*

*~~distributed under the License is distributed on an "AS IS" BASIS,~~*

*~~WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.~~*

*~~See the License for the specific language governing permissions and~~*

*~~limitations under the License.~~*

*~~-->~~*

*~~<!-- Note: A "Server" is not itself a "Container", so you may not~~*

*~~define subcomponents such as "Valves" at this level.~~*

*~~Documentation at /docs/config/server.html~~*

*~~-->~~*

*~~<Server port="8005" shutdown="SHUTDOWN">~~*

*~~<Listener className="org.apache.catalina.startup.VersionLoggerListener" />~~*

*~~<!-- Security listener. Documentation at /docs/config/listeners.html~~*

*~~<Listener className="org.apache.catalina.security.SecurityListener" />~~*

*~~-->~~*

*~~<!--APR library loader. Documentation at /docs/apr.html -->~~*

*~~<Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />~~*

*~~<!--Initialize Jasper prior to webapps are loaded. Documentation at /docs/jasper-howto.html -->~~*

*~~<Listener className="org.apache.catalina.core.JasperListener" />~~*

*~~<!-- Prevent memory leaks due to use of particular java/javax APIs-->~~*

*~~<Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />~~*

*~~<Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />~~*

*~~<Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />~~*

*~~<!-- Global JNDI resources~~*

*~~Documentation at /docs/jndi-resources-howto.html~~*

*~~-->~~*

*~~<GlobalNamingResources>~~*

*~~<!-- Editable user database that can also be used by~~*

*~~UserDatabaseRealm to authenticate users~~*

*~~-->~~*

*~~<Resource name="UserDatabase" auth="Container"~~*

*~~type="org.apache.catalina.UserDatabase"~~*

*~~description="User database that can be updated and saved"~~*

*~~factory="org.apache.catalina.users.MemoryUserDatabaseFactory"~~*

*~~pathname="conf/tomcat-users.xml" />~~*

*~~</GlobalNamingResources>~~*

*~~<!-- A "Service" is a collection of one or more "Connectors" that share~~*

*~~a single "Container" Note: A "Service" is not itself a "Container",~~*

*~~so you may not define subcomponents such as "Valves" at this level.~~*

*~~Documentation at /docs/config/service.html~~*

*~~-->~~*

*~~<Service name="Catalina">~~*

*~~<!--The connectors can use a shared executor, you can define one or more named thread pools-->~~*

*~~<!--~~*

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

*~~maxThreads="150" minSpareThreads="4"/>~~*

*~~-->~~*

*~~<!-- A "Connector" represents an endpoint by which requests are received~~*

*~~and responses are returned. Documentation at :~~*

*~~Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)~~*

*~~Java AJP Connector: /docs/config/ajp.html~~*

*~~APR (HTTP/AJP) Connector: /docs/apr.html~~*

*~~Define a non-SSL HTTP/1.1 Connector on port 8080~~*

*~~-->~~*

*~~<Connector port="8080"~~* ***~~URIEncoding="UTF-8~~****~~" protocol="HTTP/1.1"~~*

*~~connectionTimeout="20000"~~*

*~~redirectPort="8443" />~~*

*~~<!-- A "Connector" using the shared thread pool-->~~*

*~~<!--~~*

*~~<Connector executor="tomcatThreadPool"~~*

*~~port="8080" protocol="HTTP/1.1"~~*

*~~connectionTimeout="20000"~~*

*~~redirectPort="8443" />~~*

*~~-->~~*

*~~<!-- Define a SSL HTTP/1.1 Connector on port 8443~~*

*~~This connector uses the BIO implementation that requires the JSSE~~*

*~~style configuration. When using the APR/native implementation, the~~*

*~~OpenSSL style configuration is required as described in the APR/native~~*

*~~documentation -->~~*

*~~<!--~~*

*~~<Connector port="8443" protocol="org.apache.coyote.http11.Http11Protocol"~~*

*~~maxThreads="150" SSLEnabled="true" scheme="https" secure="true"~~*

*~~clientAuth="false" sslProtocol="TLS" />~~*

*~~-->~~*

*~~<!-- Define an AJP 1.3 Connector on port 8009 -->~~*

*~~<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />~~*

*~~<!-- An Engine represents the entry point (within Catalina) that processes~~*

*~~every request. The Engine implementation for Tomcat stand alone~~*

*~~analyzes the HTTP headers included with the request, and passes them~~*

*~~on to the appropriate Host (virtual host).~~*

*~~Documentation at /docs/config/engine.html -->~~*

*~~<!-- You should set jvmRoute to support load-balancing via AJP ie :~~*

*~~<Engine name="Catalina" defaultHost="localhost" jvmRoute="jvm1">~~*

*~~-->~~*

*~~<Engine name="Catalina" defaultHost="localhost">~~*

*~~<!--For clustering, please take a look at documentation at:~~*

*~~/docs/cluster-howto.html (simple how to)~~*

*~~/docs/config/cluster.html (reference documentation) -->~~*

*~~<!--~~*

*~~<Cluster className="org.apache.catalina.ha.tcp.SimpleTcpCluster"/>~~*

*~~-->~~*

*~~<!-- Use the LockOutRealm to prevent attempts to guess user passwords~~*

*~~via a brute-force attack -->~~*

*~~<Realm className="org.apache.catalina.realm.LockOutRealm">~~*

*~~<!-- This Realm uses the UserDatabase configured in the global JNDI~~*

*~~resources under the key "UserDatabase". Any edits~~*

*~~that are performed against this UserDatabase are immediately~~*

*~~available for use by the Realm. -->~~*

*~~<Realm className="org.apache.catalina.realm.UserDatabaseRealm"~~*

*~~resourceName="UserDatabase"/>~~*

*~~</Realm>~~*

*~~<Host name="localhost" appBase="webapps" unpackWARs="true" autoDeploy="true">~~*

***~~<Listener className="org.apache.jk.config.ApacheConfig" modJk="D:/Apache24/modules/mod\_jk.so" workersConfig="D:/Java/Tools/apache-tomcat-7.0.68-windows-x64/conf/jk/workers.properties" append="true" forwardAll="false"/>~~***

*~~<!-- SingleSignOn valve, share authentication between web applications~~*

*~~Documentation at: /docs/config/valve.html -->~~*

*~~<!--~~*

*~~<Valve className="org.apache.catalina.authenticator.SingleSignOn" />~~*

*~~-->~~*

*~~<!-- Access log processes all example.~~*

*~~Documentation at: /docs/config/valve.html~~*

*~~Note: The pattern used is equivalent to using pattern="common" -->~~*

*~~<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>~~*

*~~</Engine>~~*

*~~</Service>~~*

*~~</Server>~~*

*<?xml version='1.0' encoding='utf-8'?>*

*<Server port="8105" shutdown="SHUTDOWN">*

*…*

*<Connector port="8180"* ***URIEncoding="UTF-8****" protocol="HTTP/1.1"*

*connectionTimeout="20000"*

*redirectPort="8443" />*

*…*

*<Engine name="Catalina" defaultHost="localhost">*

*<Host name="localhost" appBase="webapps" unpackWARs="true" autoDeploy="true">*

*…*

***<Valve className="org.apache.catalina.authenticator.SingleSignOn" />***

*…*

*</Host>*

*</Engine>*

*</Service>*

*</Server>*

# Приложение 4.Пример настройки файла mod\_jk.conf

Пример настройки приводится для обеспечения запуска примеров Tomcat:

*########## Modified Auto generated on Wed Mar 30 19:21:54 MSK 2016##########*

*<IfModule !mod\_jk.c>*

*LoadModule jk\_module "modules/mod\_jk.so"*

*</IfModule>*

*JkWorkersFile "conf/jk/workers.properties"*

*JkLogFile "logs/mod\_jk.log"*

*JkLogLevel emerg*

*# Select the log format*

*JkLogStampFormat "[%a %b %d %H:%M:%S %Y]"*

*# JkOptions indicate to send SSL KEY SIZE,*

*JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories*

*# JkRequestLogFormat set the request format*

*JkRequestLogFormat "%w %V %T"*

*# Send everything for context /ws to worker* ***worker1***

*JkMount /ws* ***worker1***

*JkMount /ws/\** ***worker1***

*# Send everything for context /examples to worker* ***worker1***

*JkMount /examples* ***worker1***

*JkMount /examples/\** ***worker1***

*JkMount /jepriashowcase-1-1* ***worker1***

*JkMount /jepriashowcase-1-1/\** ***worker1***

*JkMount /jepriashowcase-1-2* ***worker1***

*JkMount /jepriashowcase-1-2/\** ***worker1***

*JkMount /jepriashowcase-2-1* ***worker2***

*JkMount /jepriashowcase-2-1/\** ***worker2***

*JkMount /jepriashowcase-2-2* ***worker2***

*JkMount /jepriashowcase-2-2/\** ***worker2***

Замечание:

Имена worker-ов (в данном случае - ***worker1****,* ***worker2***) должны соответствовать именам, определённым в файле *workers.properties*.

# Приложение 5



Рисунок . Снижение нагрузки с Tomcat посредством переноса ответственности за статические ресурсы на HTTPD.



Рисунок .Схема балансировки нагрузки.

# Ссылки

1. [Apache HTTPD, mod\_jk, Apache Tomcat, Linux](http://programador.ru/apache-mod_jk-tomcat/)
2. [Using Apache HTTP Server on Microsoft Windows](http://httpd.apache.org/docs/current/platform/windows.html#down)
3. [Installing Tomcat 6 with Java 6 and Apache 2.2 on Windows Server 2003 x64](http://www.anchor.com.au/hosting/dedicated/tomcat_with_apache_on_windows_x64)
4. Полезный ресурс: [Installing Apache and Tomcat and Mysql on Windows 7](http://boldinventions.com/index.php?option=com_content&view=article&id=95%3Ainstalling-apache-and-tomcat-and-mysql-on-window-7&catid=34%3Acategory-electronics-articles&Itemid=53)
5. Помогло при настройке *mod\_jk.conf*: [How To Configure Tomcat to work with Apache](http://www.ntu.edu.sg/home/ehchua/programming/howto/apacheplustomcat_howto.html)