Guia de Configuração HTTPS – Datasul 11.5.12

28/07/2014

Sumário

Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©
Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©

1. Introdução

Este documento tem como objetivo documentar a instalação e configuração do apache em HTTP sobre SSL para funcionamento com o jboss-4.2.3-GA do produto Datasul. Apesar das configurações serem semelhantes, seguiremos o escopo limitado ao Windows 32bits.

Para finalizar tem um tópico sobre as configurações de Tuning do Apache para melhorar o desempenho no tratamento das requisições. Esse artigo foi baseado na versão 2.2.xx para windows, no entanto, as configurações são semelhantes aos outros SOs em que o apache Server-pool está disponível.

2. Configuração HTTPS com FrontEnd Apache – Windows 32Bits

2.1. Apache HTTP Server

A documentação foi realizada com o arquivo .msi de instalação para Windows, porém todas as configurações podem ser reutilizadas com exceção do item "b" abaixo.

a. É altamente recomendado baixar a versão apache 2.2 ou superior (a documentação foi baseada na 2.2.XX) que possui várias melhorias e correções para o melhor funcionamento com o produto TomCat disponibilizado com o JBOSS.

b. Baixar os arquivos para configurar o ambiente:

Apache	
Documentação	Link
Instalador	Link
Arquivo	httpd-2.2.25-win32-x86-openssl-0.9.8y.msi

2.2. Iniciar a instalação do Apache.

a. Duplo clique no arquivo httpd-2.2.25-win32-x86-openssl-0.9.8y.msi

b. Escolher Instalação Customizada (Custom)

c. Instalar todos os pacotes no disco local.

d. Definir a porta 80 como padrão do apache. Caso essa porta não esteja disponível defina outra nesse momento. Lembrando que, essa porta precisa estar liberada em regras de firewalls para correto funcionamento.

e. Finalizar a instalação.

f. Verificar se na bandeja de ícones, ao lado do relógio do sistema aparece o ícone do apache. Caso esteja verde está online caso contrário deve-se clicar com o botão esquerdo e iniciar o serviço.

g. Abrir o navegador de sua preferência e digitar http://<server> e o apache deve mostrar uma página com a mensagem "It works!"

2.3. Configurar mod_jk para loadbalance

a. Parar o apache.

b. Baixar o pacote do mod_jk:

TomCat Connector	
Instalador	Link

c. Descompactar o arquivo e copiar o arquivo mod_jk.so para a pasta APACHE_HOME\modules\

d. Criar o arquivo APACHE_HOME\conf\extra\httpd-mod-jk.conf e configurar o modulo mod_jk.so como mostrado abaixo:

d1. Adicionar o conteúdo abaixo no arquivo e salvar.

```
# Configuration Example for mod_jk
# used in combination with Apache 2.2.x
# Change the path and file name of the module, in case
# you have installed it outside of httpd, or using
# a versioned file name.
LoadModule jk_module modules/mod_jk.so
<IfModule jk_module>
#copy configurations to all virtual hosts and servers
JkMountCopy All
# We need a workers file exactly once
# and in the global server
JkWorkersFile conf/workers.properties
# Our JK error log
# You can (and should) use rotatelogs here
JkLogFile logs/mod_jk.log
# Our JK log level (trace,debug,info,warn,error)
JkLogLevel info
# Our JK shared memory file
JkShmFile logs/mod_jk.shm
# Define a new log format you can use in any CustomLog in order
# to add mod_jk specific information to your access log.
# LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" \"%{Cookie}i\" \"%{Set-Cookie}o\" %{pid}P %{tid}P
%{JK_LB_FIRST_NAME}n %{JK_LB_LAST_NAME}n ACC %{JK_LB_LAST_ACCESSED}n ERR %{JK_LB_LAST_ERRORS}n BSY
%{JK_LB_LAST_BUSY}n %{JK_LB_LAST_STATE}n %D" extended_jk
# This option will reject all requests, which contain an
# encoded percent sign (%25) or backslash (%5C) in the URL
# If you are sure, that your webapp doesn't use such
# URLs, enable the option to prevent double encoding attacks.
```


Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©
Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©

```
# Since: 1.2.24
# JkOptions +RejectUnsafeURI
# After setting JkStripSession to "On", mod_jk will
# strip all ";jsessionid=..." from request URLs it
# does *not* forward to a backend.
# This is useful, if all links in a webapp use
# URLencoded session IDs and parts of the static
# content should be delivered directly by Apache.
# Of course you can also do it with mod_rewrite.
# Since: 1.2.21
# JkStripSession On
# Start a separate thread for internal tasks like
# idle connection probing, connection pool resizing
# and load value decay.
# Run these tasks every JkWatchdogInterval seconds.
# Since: 1.2.27
JkWatchdogInterval 60
# Configure access to jk-status and jk-manager
# If you want to make this available in a virtual host,
# either move this block into the virtual host
# or copy it logically there by including "JkMountCopy On"
# in the virtual host.
# Add an appropriate authentication method here!
#<Location /jk-status>
# Inside Location we can omit the URL in JkMount
#    JkMount jk-status
#    Order deny,allow
#    Deny from all
#    Allow from 127.0.0.1
#</Location>
#<Location /jk-manager>
#    # Inside Location we can omit the URL in JkMount
#    JkMount jk-manager
#    Order deny,allow
#    Deny from all
#    Allow from 127.0.0.1
#</Location>
# If you want to put all mounts into an external file
# that gets reloaded automatically after changes
# (with a default latency of 1 minute),
# you can define the name of the file here.
JkMountFile conf/uriworkermap.properties
# Example for Mounting a context to the worker "balancer"
# The URL syntax "a|b" instantiates two mounts at once,
# the first one is "a", the second one is "ab".
# JkMount /myapp|/* balancer
# Example for UnMounting requests for all workers
# using a simple URL pattern
# Since: 1.2.26
# JkUnMount /myapp/static/* *
# Example for UnMounting requests for a named worker
# JkUnMount /myapp/images/* balancer
# Example for UnMounting requests using regexps
# SetEnvIf REQUEST_URI "\.(htm|html|css|gif|jpg|js)$" no-jk
# Example for setting a reply timeout depending on the request URL
# Since: 1.2.27
# SetEnvIf Request_URI "/transactions/" JK_REPLY_TIMEOUT=600000
# Example for disabling reply timeouts for certain request URLs
```

```
# Since: 1.2.27
# SetEnvIf Request_URI "/reports/" JK_REPLY_TIMEOUT=0
# IMPORTANT: Mounts and virtual hosts
# If you are using VirtualHost elements, you
# - can put mounts only used in some virtual host into its VirtualHost element
# - can copy all global mounts to it using "JkMountCopy On" inside the VirtualHost
# - can copy all global mounts to all virtual hosts by putting
#   "JkMountCopy All" into the global server
# Since: 1.2.26
</IfModule>
```

d2. Abrir o arquivo APACHE_HOME\conf\httpd.conf e adicionar no final do arquivo a linha abaixo fazendo referência ao arquivo criado no item "d1".

```
# Conf for mod_jk balance.
Include conf/extra/httpd-mod-jk.conf
```

e. Criar o arquivo APACHE_HOME/conf/workers.properties

e1. Abrir o arquivo com editor de sua preferência e colocar o conteúdo abaixo.

```
# 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 that the distributed version of this file requires modification
# before it is usable.
#
# Reference documentation: http://tomcat.apache.org/connectors-doc/reference/workers.html
#
# As a general note, the characters $( and ) are used internally to define
# macros. Do not use them in your own configuration!!!
#
# Whenever you see a set of lines such as:
# x=value
# y=$(x)\something
#
# the final value for y will be value\something

# Define two status worker:
# - jk-status for read-only use
# - jk-manager for read/write use
#worker.list=jk-status
#worker.jk-status.type=status
#worker.jk-status.read_only=true
```


Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©
Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©

```
#worker.list=jk-manager
#worker.jk-manager.type=status

# We define a load balancer worker
# with name "balancer"
worker.list=balancer
worker.balancer.type=lb
# error_escalation_time: seconds, default = recover_time/2 (=30)
# Determines, how fast a detected error should switch from
# local error state to global error state
# Since: 1.2.28
worker.balancer.error_escalation_time=0

# - max_reply_timeouts: number, default=0
#   If there are to many reply timeouts, a worker
#   is put into the error state, i.e. it will become
#   unavailable for all sessions residing on the respective
#   Tomcat. The number of tolerated reply timeouts is
#   configured with max_reply_timeouts. The number of
#   timeouts occuring is divided by 2 once a minute and the
#   resulting counter is compared against max_reply_timeouts.
#   If you set max_reply_timeouts to N and the errors are
#   occuring equally distributed over time, you will
#   tolerate N/2 errors per minute. If they occur in a burst
#   you will tolerate N errors.
#   Since: 1.2.24
worker.balancer.max_reply_timeouts=10

# Now we add members to the load balancer
# First member is "node1", most
# attributes are inherited from the
# template "worker.template".
worker.balancer.balance_workers=node1
worker.node1.reference=worker.template
worker.node1.host=10.80.18.123
worker.node1.port=8109
# Activation allows to configure
# whether this node should actually be used
# A: active (use node fully)
# D: disabled (only use, if sticky session needs this node)
# S: stopped (do not use)
#   Since: 1.2.19
worker.node1.activation=A

# Second member is "node2", most
# attributes are inherited from the
# template "worker.template".
worker.balancer.balance_workers=node2
worker.node2.reference=worker.template
worker.node2.host=10.80.18.88
worker.node2.port=8009
# Activation allows to configure
# whether this node should actually be used
# A: active (use node fully)
# D: disabled (only use, if sticky session needs this node)
# S: stopped (do not use)
#   Since: 1.2.19
```

Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©

Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©

```
worker.node2.activation=A

# Finally we put the parameters
# which should apply to all our ajp13
# workers into the referenced template
# - Type is ajp13
worker.template.type=ajp13

# - socket_connect_timeout: milliseconds, default=0
#   Since: 1.2.27
worker.template.socket_connect_timeout=5000

# - socket_keepalive: boolean, default=false
#   Should we send TCP keepalive packets
#   when connection is idle (socket option)?
worker.template.socket_keepalive=true

# - ping_mode: Character, default=none
#   When should we use cping/cpong connection probing?
#   C = directly after establishing a new connection
#   P = directly before sending each request
#   I = in regular intervals for idle connections
#       using the watchdog thread
#   A = all of the above
#   Since: 1.2.27
worker.template.ping_mode=A

# - ping_timeout: milliseconds, default=10000
#   Wait timeout for cpong after cping
#   Can be overwritten for modes C and P
#   Using connect_timeout and prepost_timeout.
#   Since: 1.2.27
worker.template.ping_timeout=10000

# - connection_pool_minsize: number, default=connection_pool_size
#   Lower pool size when shrinking pool due
#   to idle connections
#   We want all connections to be closed when
#   idle for a long time in order to prevent
#   firewall problems.
#   Since: 1.2.16
worker.template.connection_pool_minsize=0

# - connection_pool_timeout: seconds, default=0
#   Idle time, before a connection is eligible
#   for being closed (pool shrinking).
#   This should be the same value as connectionTimeout
#   in the Tomcat AJP connector, but there it is
#   milliseconds, here seconds.
worker.template.connection_pool_timeout=600

# - reply_timeout: milliseconds, default=0
#   Any pause longer than this timeout during waiting
#   for a part of the reply will abort handling the request
#   in mod_jk. The request will proceed running in
#   Tomcat, but the web server resources will be freed
```

```
#   and an error is send to the client.
#   For individual requests, the timeout can be overwritten
#   by the Apache environment variable JK_REPLY_TIMEOUT.
#   JK_REPLY_TIMEOUT since: 1.2.27
worker.template.reply_timeout=300000

# - recovery_options: number, default=0
#   Bit mask to configure, if a request, which was send
#   to a backend successfully, should be retried on another backend
#   in case there's a problem with the response.
#   Value "3" disables retries, whenever a part of the request was
#   successfully send to the backend.
worker.template.recovery_options=3
```

e2. Configurar as propriedades que identificam os nodes(jboss) que participaram do balance.
Para isso basta encontrar o bloco de configuração mostrado abaixo e alterar de acordo com as necessidades.

```
# Now we add members to the load balancer
# First member is "node1", most
# attributes are inherited from the
# template "worker.template".
worker.balancer.balance_workers=<nome do node>
worker.<nome do node>.reference=worker.template
worker.<nome do node>.host=<ip aonde está rodando o jboss>
worker.<nome do node>.port=<porta do connector ajp13, não é a porta da url do produto>
# Activation allows to configure
# whether this node should actually be used
# A: active (use node fully)
# D: disabled (only use, if sticky session needs this node)
# S: stopped (do not use)
#   Since: 1.2.19
worker.<nome do node>.activation=A
```

e3. Configurar os nodes (instancias do jboss) para jvmRoute afim de restringir a session do user a um único node.

- Acessar JBOSS_HOME/server/<instancia do cliente>/deploy/jboss-web.deployer/server.xml

- Localizar a tag abaixo:

```
<Engine name="jboss.web" defaultHost="localhost">
```

- Adicionar o atributo deixando a configuração como mostrado abaixo

```
<Engine name="jboss.web" defaultHost="localhost" jvmRoute="<nome do node>">
```

e4. Fazer a configuração "e3" para todos os nodes (JBOSS) que participam do balance.

f. Criar o arquivo APACHE_HOME/conf/uriworkermap.properties

f1. abrir o arquivo criado e adicionar o conteúdo abaixo:

```
# 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.

# uriworkermap.properties - IIS
#
# This file provides sample mappings for example wlb
# worker defined in workermap.properties.minimal
# The general syntax for this file is:
# [URL]=[Worker name]

/datasul=balancer
/datasul/*=balancer
/josso=balancer
/josso/*=balancer
/docs=balancer
/docs/*=balancer

# Optionally filter out all .jpeg files inside that context
# For no mapping the url has to start with exclamation (!)

#!/servlets-examples/*.jpeg=lb

#
# Mount jkstatus to /jkmanager
# For production servers you will need to
# secure the access to the /jkmanager url
#
#/jk-manager=jk-status
```

g. Iniciar os nodes (JBOSS)

h. Iniciar o Apache

2.4. Configurar SSL

a. Abrir o arquivo APACHE_HOME/conf/httpd.conf

b. Localizar e tirar o comentário da linha #Include conf/extra/httpd-ssl.conf e salvar.

c. Abrir o arquivo APACHE_HOME/conf/extra/httpd-ssl.conf e configurar como mostrado abaixo:

c1. Comentar as propriedades SSLSessionCache e SSLSessionCacheTimeout

c2. Criar a pasta APACHE_HOME/conf/data/cert e usar para colocar os certificados.

Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©
Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©

c3. Localizar a propriedade SSLCertificateFile e apontar para o arquivo .pem que contenha o certificado e a chave privada. Caso a chave privada esteja em outro arquivo é possível configurar o certificado no SSLCertificateFile e a chave privada na propriedade SSLCertificateKeyFile.

c4. Para produzir um "arquivo.pem" a partir de um "arquivo.jks" deve-se executar o seguinte comando.

Primeiro jks para pkcs12

```
keytool -importkeystore -srckeystore keystore.jks -destkeystore intermediate.p12 -deststoretype PKCS12
```

Segundo pkcs12 para pem

```
openssl pkcs12 -in intermediate.p12 -out extracted.pem –nodes
```

d. modificar os nodes(para https)

d1. abrir o arquivo JBOSS_HOME/server/<instancia do cliente>/deploy/jboss-web.deployer/server.xml

d2. localizar a tag Connector do ajp13 e configurar como abaixo:

O connector deve estar assim:

```
<Connector port="8009" address="${jboss.bind.address}" protocol="AJP/1.3"
 emptySessionPath="true" enableLookups="false" redirectPort="8443" />
```

Configurações do certificado:

```
  <Connector port="8050" address="${jboss.bind.address}" protocol="AJP/1.3"
          emptySessionPath="true" enableLookups="false" redirectPort="8443"
          maxThreads="150"
          SSLEnabled="true"
          scheme="https" secure="true"
          clientAuth="false" sslProtocol="TLS"
          keystoreFile="<caminho para o certificado jks>"
          keyAlias="<alias do certificado>"
          keystorePass="<pwd do certificado>"/>
```

d3. fazer a configuração de d2 para todas os nodes(jboss) do balance.

e. Reiniciar os nodes (jboss)

f. Reiniciar o apache.

Com essas configurações o sistema rodará normalmente, porém vale relembrar que o sistema funcionará completamente em HTTP sobre SSL com a compilação do Flex com os endpoints em HTTPS. Sendo assim, é necessário que seja feita a compilação dos fontes com os endpoints alterados.

3. Tunning Apache

3.1. Sistema Operacional

O servidor apache possui um compilado para cada sistema operacional e de acordo com a compilação o modulo que atende as configurações é disponibilizada. Para saber o módulo que atende seu SO basta seguir a tabela abaixo:

Apache	
BeOS	Link
Netware	Link
OS/2	Link
Unix	Link
Windows	Link

Tabela 1: modules disponíveis por compilação de SOs

3.2. Configurações

As configurações descritas nesse artigo foram baseadas na versão 2.2.xx do apache server, sendo assim, rara configurar o apache basta abrir o arquivo $APACHE_HOME\conf\httpd.conf e encontrar o conteúdo abaixo e **tirar o comentário da segunda linha**:

De:
```
# Server-pool management (MPM specific)
# Include conf/extra/httpd-mpm.conf
```

Para:

```
# Server-pool management (MPM specific)
Include conf/extra/httpd-mpm.conf
```

Salvar e fechar o arquivo mencionado acima e abrir o arquivo $APACHE_HOME\conf\extra\httpd-mpm.conf. Dentro desse arquivo existem todas as parameterizações por SO de acordo com a ***Tabela 1***. Para o artigo em específico, como mencionando anteriormente utilizaremos a versão disponibilizada para Windows.

Sendo assim, basta encontrar a configuração abaixo e configurar de acordo o mpm_worker_module e o mpm_winnt_module:

De:
```
<IfModule mpm_worker_module>
    StartServers         2
    MaxClients         150
    MinSpareThreads      25
    MaxSpareThreads      75
    ThreadsPerChild      25
    MaxRequestsPerChild   0
</IfModule>
```

Para:

```
<IfModule mpm_worker_module>
   StartServers        15
   MaxClients         300
   MinSpareThreads     50
   MaxSpareThreads     100
   ThreadsPerChild     50
   MaxRequestsPerChild 500
</IfModule>
```

De:

```
<IfModule mpm_winnt_module>
   ThreadsPerChild     150
   MaxRequestsPerChild    0
</IfModule>
```

Para:

```
<IfModule mpm_winnt_module>
   ThreadsPerChild     600
   MaxRequestsPerChild  500
</IfModule>
```

Para finalizar deve-se adicionar no início do arquivo as configurações abaixo:

```
#
# Timeout: The number of seconds before receives and sends time out.
#
Timeout 300

#
# KeepAlive: Whether or not to allow persistent connections (more than
# one request per connection). Set to "Off" to deactivate.
#
KeepAlive On

#
# MaxKeepAliveRequests: The maximum number of requests to allow
# during a persistent connection. Set to 0 to allow an unlimited amount.
# We recommend you leave this number high, for maximum performance.
#
MaxKeepAliveRequests 10000

#
# KeepAliveTimeout: Number of seconds to wait for the next request from the
# same client on the same connection.
#
KeepAliveTimeout 15
```

Salvar o o arquivo e iniciar o apache. Desta forma o apache estará apto a executar dentro de um limite considerável de performance, no entanto, caso seja necessário, é possível aumentar as configurações.

Obs.: Deve-se ter cuidado com os recursos utilizados no server para não consumir o SO por completo e deixar o sistema ainda mais lento.

4. Acerto Ambiente

Para utilizar o produto com HTTPS, o arquivo index-flex-https.swf (presente na raiz do WAR: [..]\datasul-byyou-11.5.9-SNAPSHOT.ear\datasul-framework-ui.war) deve ser renomeado para index-flex.swf, e o arquivo index-flex.swf original (que corresponde ao index para http normal) deve ser renomeado com outro nome, para que seja considerado o .swf para https.

Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©
Este documento é de propriedade da TOTVS. Todos os direitos reservados. ©