[Installing Instructions 3](#_Toc512356507)

[**Install JDK 1.7.0\_60** 4](#_Toc512356508)

[**Set JAVA\_HOME** 4](#_Toc512356509)

[Disable FirewalID 5](#_Toc512356510)

[Linux commands 5](#_Toc512356511)

[Changing the Default Database 6](#_Toc512356512)

[Setting up MS Sql 7](#_Toc512356513)

[Followings are the steps to setup database and users 7](#_Toc512356514)

[Create the database and user 7](#_Toc512356515)

[Setting up the JDBC driver 8](#_Toc512356516)

[Changing the default database in ESB5.0.0 datasource 8](#_Toc512356517)

[WSO2 API MANAGER 2.1.0 10](#_Toc512356518)

[Not Certificate Found in Chrome 15](#_Toc512356519)

[renewing CA issued certificate chain in jks 16](#_Toc512356520)

[Add New .jks using .pfx file 16](#_Toc512356521)

[Create .pem File for haproxy 19](#_Toc512356522)

[Add New User Store 20](#_Toc512356523)

[Writing up custom user store jar file 21](#_Toc512356524)

[Create Crontab Services to touch custom userstore db connection 21](#_Toc512356525)

[Create daemon services 21](#_Toc512356526)

[Create Service Provider 26](#_Toc512356527)

[Inbound Authentication Configuration/ OAuth/OpenID Connect Configuration 26](#_Toc512356528)

[Create Identity Provider 26](#_Toc512356529)

[Federated Authenticators 26](#_Toc512356530)

[SAML2 Web SSO Configuration 26](#_Toc512356531)

[CREATE PROXY SERVICE FOR MAIN API SERVICE 27](#_Toc512356532)

[API CREATION 28](#_Toc512356533)

[IN AND OUT SEQUENCE 31](#_Toc512356534)

[Test Services 36](#_Toc512356535)

[Final Result 39](#_Toc512356536)

[Test ADFS Login with SAML Assertion 40](#_Toc512356537)

[Tools 40](#_Toc512356538)

[Reference: 41](#_Toc512356539)

[Sample code to read header token values using C#.NET 41](#_Toc512356540)

[Haproxy.cfg file 42](#_Toc512356541)

[Automatically Start HAProxy Service 43](#_Toc512356542)

[Configuring the Identity Server 5.1.0 as a Key Manager ... 45](#_Toc512356543)

[**Configure Ids to get the token for APIM** 46](#_Toc512356544)

[APIM 1.10 Refresh token Issue 46](#_Toc512356545)

[Revoke token 46](#_Toc512356546)

[Generating a new access token and refresh token 47](#_Toc512356547)

[Monitoring WSO2 ESB using JConsole 47](#_Toc512356548)

[Monitor Carbon log using a UNIX Cron job 47](#_Toc512356549)

[**Automatic shutdown azure VMs to save money** 48](#_Toc512356550)

[Understand Scheduler job Details 49](#_Toc512356551)

[**How to run a script before system shutdown/reboot with Systemd** 51](#_Toc512356552)

[**Manage Log Files** 52](#_Toc512356553)

[Crontab to delete log files (file older than specific days) 52](#_Toc512356554)

[{HowTo} WSO2 Carbon Server logs to be stored in a database 52](#_Toc512356555)

[Working with wso2 DATA ANALYTICS SERVER (DAS 3.0.1) 54](#_Toc512356556)

[Azure vm Auto Scaling 54](#_Toc512356557)

[Create azure vm auto scaling 55](#_Toc512356558)

[deploy application on vmss 55](#_Toc512356559)

[Osura’s blog 55](#_Toc512356560)

[Schedule Auto Scaling 55](#_Toc512356561)

[Deploy a VM Scale Set based on a Linux Custom Image and a script to deploy updates 55](#_Toc512356562)

[VMSS Application Deployment 66](#_Toc512356563)

[Storage account type scaling 67](#_Toc512356564)

[Limits 67](#_Toc512356565)

[POC VMSS LoadTesting 67](#_Toc512356566)

[Create Swap file in linux (cent os 7 ) 67](#_Toc512356567)

[Verify Log file 68](#_Toc512356568)

[Change Host Name 68](#_Toc512356569)

[Api Manager OS Level Tune up 68](#_Toc512356570)

[Ssh to another vm 69](#_Toc512356571)

[Pass 69](#_Toc512356572)

[Live HAPROXY.cfg 72](#_Toc512356573)

[To Check Live VM haproxy.cfg change like this 74](#_Toc512356574)

[How To Handle Schema Changes In Azure SQL Data Sync 76](#_Toc512356575)

[Sync data across multiple cloud and on-premises databases with SQL Data Sync (Preview) 76](#_Toc512356576)

[**Oasys Live shift from classic vm to ARM** 77](#_Toc512356577)

[Steps to Create a New Sudo User 77](#_Toc512356578)

[WSO2 IS 5.1 PATCH 79](#_Toc512356579)

[IDP ALLIAS 79](#_Toc512356580)

[Configure apim 1.10 & IDS 5.1 as Key Manager 80](#_Toc512356581)

[Step 1 - Download WSO2 APIM and WSO2 IS 80](#_Toc512356582)

[Step 2 - Optionally, configure port offset for WSO2 APIM or WSO2 IS 80](#_Toc512356583)

[Step 3 - Configure the Identity Server 80](#_Toc512356584)

[Step 4 - Configure the API Manager 87](#_Toc512356585)

[NO CERTIFICATE ERROR IN CHROME 92](#_Toc512356586)

***Api-poc***

***Apim – mgt console 9643***

***Service port : 8443***

***Api-Dev***

***Apim – mgt console 9643***

***Service port :443***

|  |
| --- |
| *DEV IP: 52.230.27.53*  *Ip address : devoasys.southeastasia.cloudapp.azure.com*  User Name: *devvmadmin password: DevVmAdmin@123* |

***Versions****: WSO2 ESB 5.0.0 /WSO2 API MANAGER 2.1.0*

*JDK 1.7.0\_60*

Installing Instructions**:**

## **Install JDK 1.7.0\_60**

* 1. **Unzip tar file**

**Linux Command: tar xzf jdk-7u60-linux-x64.ta**

* 1. **Install java using alternative command**

**Linux Command:**

**alternatives --install/usr/bin/java java home/devvmadmin/jdk1.7.0\_60/bin/java 2**

**alternatives –config java**

*(We can check java version by using command:* ***java -version****)*

## **Set JAVA\_HOME**

**Linux Command: export JAVA\_HOME="/home/devvmadmin/jdk1.7.0\_60"**

Also we can edit .bash\_profile for setting up JAVA\_HOME refer the link http://legendans.blogspot.com/2015/03/how-to-install-oracle-java-7-in.html

After setting up java we can install both 2 carbon products on Linux.

First we have to download ESB & API Manager. Then unzip by using following Linux command

(**unzip wso2esb-5.0.0.zip /unzip wso2am-2.1.0.zip)**

Then move to the <ESBHome>/repository/conf/carbon.xml and set <offset> value as 100 as same as change offset value as 200 of the Api manager.

After setting up offset value move both <ESBHome>/<APIHome>/bin folder to install. We can use this command ***./wso2server.sh***

***<Note>***

***Before login to the management console of carbon product you have to enable two main port and also service ports***

***9543 : esb server***

***Service ports : 8380(http)/8343 (https)***

***9643 : api manager***

***Service ports : 8480(http)/ 8443 (https)***

http & https port numbers are depending on the number which are denoted in </home/devvmadmin/wso2am-2.1.0/repository/conf/axis2> axis2.xml file. Under the transport listener

## Disable FirewalID

Check the status of the firewalld service

**systemctl status firewall.service**

If the firewall is active / running, enter this command to stop it:

**systemctl stop firewalld.service**

To completely disable the firewalld service, so it does not reload when you restart the host machine:

**systemctl disable firewalld.service**

Verify the status of the firewalld service:

**systemctl status firewalld.service**

The status should display as disabled and inactive (dead).

**firewalld.service - firewalld - dynamic firewall daemon**

**Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)**

**Active: inactive (dead)**

**Docs: man:firewalld(1)**

**Link:**

*[https://docs.eucalyptus.com/eucalyptus/4.3/install-guide/rhel7\_disable\_firewalld.html]*

## Linux commands

|  |  |  |
| --- | --- | --- |
| # | Desc | Command |
| 1 | **Unzip** | **unzip wso2am-2.1.0.zip** |
|  |  | **tar xzf jdk-7u60-linux-x64.ta** |
| 2 | **Move** | **mv /home/devvmadmin/rpmPackages/wso2am-2.1.0.zip .(dot)**  *(dot denoted current location you are in)* |
| 3 | **Remove folder** | **rm -rf wso2am-2.1.0** |
| 4 | **Find drives** | **df -h** |
| 5 | **find running Instances** | **ps -ef|grep wso2** |
| 6 | **Kill runing process !** | **kill -9 <99795> process ID** |
| 7 | **Start Server** | **sh wso2server.sh start** |
| 8 | **chown command** | chown is used to change the owner and group of files, directories and links. By default, the owner of a filesystem object is the user that created it  ex: **chown -R devvmadmin:devvmadmin /u01/** |
| 9 | **View all certificate** | **keytool -list -v -keystore client-truststore.jks** |
| 10 | **Create Certificate** | 1. keytool -genkey -alias wso2apim -keyalg RSA -keystore wso2carbon.jks -keysize 2048 -storepass wso2apim -keypass wso2apim -validity 7300 2. keytool -exportcert -alias wso2esb -file wso2carbon.crt -keystore wso2carbon.jks -storepass wso2esb 3. keytool -importcert -alias wso2esb -keystore client-truststore.jks -file wso2carbon.crt |
| 11 | **Grant permission** | **chmod 755 <filename>,chmod +x <filename>** |
| 12 | **Date/ time zone** | sudo timedatectl list-timezones  sudo timedatectl set-timezone Asia/Colombo  sudo timedatectl |
| 13 | **View Mem Stats** | **cat /proc/meminfo** |
| 14 | **View used port** | **lsof -i:443** |
| 15 | **View mem info** | **cat /proc/meminfo** |
| 16 | **View run processes** | **top** |

# [Changing the Default Database](https://docs.wso2.com/display/AM210/Changing+the+Default+API-M+Databases)

***Database Name: OASYS\_DEV\_APIM***

1. <API\_HOME\_2.1>/ dbscripts/ mssql.sql
2. <API\_HOME\_2.1>/ dbscripts/apimgt/mssql.sql

***Database Name: OASYS\_DEV\_GREG***

1. ***<GREG\_HOME\_5.1.0>/*** dbscripts/ mssql.sql
2. ***<GREG\_HOME\_5.1.0>/*** dbscripts/ identity/mssql.sql
3. ***<GREG\_HOME\_5.1.0>/*** dbscripts/ social/mssql/resource.sql

***Database Name: OASYS\_DEV\_ESB***

1. ***<ESB\_HOME5.0.0>/*** dbscripts/ mssql.sql

**Step 1**

## Setting up MS Sql

### Followings are the steps to setup database and users

**Enable TCP/IP**

1. In the start menu, click **Programs** and launch **Microsoft SQL Server.**
2. Click**Configuration Tools**, and then click**SQL Server Configuration Manager**.
3. Enable **TCP/IP** and disable **Named Pipes** from protocols of your Microsoft SQL server.
4. Double click **TCP/IP** to open the TCP/IP properties window and set **Listen All** to Yes on the **Protocol** tab.
5. On the **IP Address** tab, disable **TCP Dynamic Ports** by leaving it blank and give a valid TCP port, so that Microsoft SQL server will listen on that port.**<1433>**
6. Similarly, enable TCP/IP from **SQL Native Client Configuration** and disable **Named Pipes**. Also, check whether the port is set correctly to 1433.

## Create the database and user

1. Open the Microsoft SQL Management Studio to create a database and user.
2. Click **New Database** from the **Database** menu and specify all the options to create a new database.
3. Click **New Login** from the **Logins** menu, and specify all the necessary options.

##### ***Grant permissions***

*Assign newly created users the required grants/permissions to log in and create tables, to insert, index, select, update and delete data in tables in the newly created database. These are the minimum set of SQL server permissions.*

## Setting up the JDBC driver

Download and copy the sqljdbc4 Microsoft SQL JDBC driver file to the WSO2 product's <PRODUCT\_HOME>/repository/components/lib/ directory. Use com.microsoft.sqlserver.jdbc.SQLServerDriver as the <driverClassName> in your datasource configuration in <PRODUCT\_HOME>/repository/conf/datasources/master-datasources.xml file

Note*: if you want to change version or something else for JDBC driver file make sure to delete driver from <PRODUCT\_HOME>\repository\components\dropins.*

## Changing the default database in ESB5.0.0 datasource

*Link:* [*https://docs.wso2.com/display/ADMIN44x/Changing+to+MSSQL*](https://docs.wso2.com/display/ADMIN44x/Changing+to+MSSQL)

*db credentials*

[oasysportal-dev.database.windows.net](http://oasysportal-dev.database.windows.net/)

OASYsDEVUsr

OASYsDEVUsr

0a5ys@usr

***WSO2 ESB 5.0 port: 9543***

Following text files should be modified as follows

1. **<esb home>/repository/conf/axis2/axis2.xml**

Add

<!--JSON Message Formatters-->

<messageFormatter contentType="application/jsonp"

class="org.apache.synapse.commons.json.JsonStreamFormatter"/>

<!--JSON Message Builders-->

<messageBuilder contentType="application/jsonp"

class="org.apache.synapse.commons.json.JsonStreamBuilder"/>

Edit Keystore password

<!-- ================================================= -->

<!-- Transport Ins (Listeners) -->

<!-- ================================================= -->

<Password>wso2esb</Password>

<KeyPassword>wso2esb</KeyPassword>

<!-- ================================================= -->

Transport Outs (Senders)

<!-- ================================================= -->

<Password>wso2esb</Password>

<KeyPassword>wso2esb</KeyPassword>

1. **<esb home>/repository/conf/carbon.xml**

<HostName>DEV-Middleware</HostName>

<MgtHostName>DEV-Middleware</MgtHostName>

<HideAdminServiceWSDLs>false</HideAdminServiceWSDLs>

<!-- Keystore password-->

<Password>wso2esb</Password>

<!-- Private Key alias-->

<KeyAlias>wso2esb</KeyAlias>

<!-- Private Key password-->

<KeyPassword>wso2esb</KeyPassword>

1. **<wso2esb-5.0.0>\wso2esb-5.0.0\repository\conf\tomcat\catalina-server.xml**

keystorePass="wso2esb"

1. **<wso2esb-5.0.0>\wso2esb-5.0.0\repository\conf\datasources\master-datasources.xml**

**Data source should be change to match with SQL database which you are created.**

1. **<wso2esb-5.0.0>\wso2esb-5.0.0\repository\conf\user-mgt.xml**

<AdminUser>

<UserName>OasysUser</UserName>

<Password>oasysUsr@123$</Password>

</AdminUser>

1. **<wso2esb-5.0.0>\wso2esb-5.0.0\repository\conf\mgt-transports.xml**

<parameter name="keystorePass">xxxxx</parameter>

WSO2 API MANAGER 2.1.0 **port: 9443 LB:9643**

1. **<api home>/repository/conf/api-manager.xml - Pending**

<GatewayEndpoint>[http://api.oasys.lk:${http.nio.port},https://api.oasys.lk:${https.nio.port}</GatewayEndpoint](http://api.oasys.lk:$%7bhttp.nio.port%7d,https://api.oasys.lk:$%7bhttps.nio.port%7d%3c/GatewayEndpoint)>

<!-- Admin username for API key manager. -->

<Username>OasysUser</Username>

<!-- Admin password for API key manager. -->

<Password>0asysUsr@123$</Password>

<URL>[https://api.oasys.lk:${mgt.transport.https.port}/store</URL](https://api.oasys.lk:$%7bmgt.transport.https.port%7d/store%3c/URL)>

1. **<api home>/repository/conf/axis2/axis2.xml**

<!--JSON Message Formatters-->

<messageFormatter

contentType="application/jsonp"class="org.apache.synapse.commons.json.

JsonStreamFormatter"/>

<!--JSON Message Builders-->

<messageBuilder contentType="application/jsonp"class="org.apache.synapse.commons.json.JsonStreamBuilder"/>

<!-- ================================================= -->

<!-- Transport Ins (Listeners) -->

<!-- ================================================= -->

<Type>JKS</Type>

<Password>wso2apim</Password>

1

<KeyPassword>wso2apim</KeyPassword>

</KeyStore>

<Type>JKS</Type>

2

<Password>wso2apim</Password>

<KeyPassword>wso2apim</KeyPassword>

</KeyStore>

<Type>JKS</Type>

3

<Password>wso2apim</Password>

<KeyPassword>wso2apim</KeyPassword>

</KeyStore>

<Type>JKS</Type>

4

<Password>wso2apim</Password>

<KeyPassword>wso2apim</KeyPassword>

</KeyStore>

<parameter name="HostnameVerifier">AllowAll</parameter>

<Type>JKS</Type>

5

<Password>wso2apim</Password>

<KeyPassword>wso2apim</KeyPassword>

</KeyStore>

<parameter name="HostnameVerifier">AllowAll</parameter>

<Type>JKS</Type>

6

<Password>wso2apim</Password>

<KeyPassword>wso2apim</KeyPassword>

</KeyStore>

<parameter name="HostnameVerifier">AllowAll</parameter>

1. **<api home>/repository/conf/axis2/broker.xml**

<keyStore>

1 & 2

<password>wso2apim</password>

1. **<api home>/repository/conf/carbon.xml**

<HostName>api.oasys.lk</HostName>

<MgtHostName>api.oasys.lk</MgtHostName>

<HideAdminServiceWSDLs>false</HideAdminServiceWSDLs>

<!-- Keystore password-->

<Password>wso2apim</Password>

<!-- Private Key alias-->

<KeyAlias>wso2apim</KeyAlias>

<!-- Private Key password-->

<KeyPassword>wso2apim</KeyPassword>

1. **<wso2am-2.1.0>\wso2am-2.1.0\repository\conf\tomcat\catalina-server.xml**

**Edit this tag as follow**

<Connector protocol="org.apache.coyote.http11.Http11NioProtocol"

port="9763"

redirectPort="9443"

bindOnInit="false"

maxHttpHeaderSize="8192"

acceptorThreadCount="2"

maxThreads="750"

minSpareThreads="150"

disableUploadTimeout="false"

enableLookups="false"

connectionUploadTimeout="120000"

maxKeepAliveRequests="600"

acceptCount="600"

server="WSO2 Carbon Server"

compression="on"

compressionMinSize="2048"

noCompressionUserAgents="gozilla, traviata"

compressableMimeType="text/html,text/javascript,application/x-javascript,application/javascript,application/xml,text/css,application/xslt+xml,text/xsl,image/gif,image/jpg,image/jpeg"

URIEncoding="UTF-8"/>

2.-------------------------------------------------------------------------------

<Connector protocol="org.apache.coyote.http11.Http11NioProtocol"

port="9543"

proxyPort="8443"

bindOnInit="false"

sslProtocol="TLS"

sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"

maxHttpHeaderSize="8192"

acceptorThreadCount="2"

maxThreads="750"

minSpareThreads="150"

disableUploadTimeout="false"

enableLookups="false"

connectionUploadTimeout="120000"

maxKeepAliveRequests="600"

acceptCount="600"

server="WSO2 Carbon Server"

clientAuth="false"

compression="on"

scheme="https"

secure="true"

SSLEnabled="true"

compressionMinSize="2048"

noCompressionUserAgents="gozilla, traviata"

compressableMimeType="text/html,text/javascript,application/x-javascript,application/javascript,application/xml,text/css,application/xslt+xml,text/xsl,image/gif,image/jpg,image/jpeg"

keystoreFile="${carbon.home}/repository/resources/security/wso2carbon.jks"

keystorePass="wso2apim"

URIEncoding="UTF-8"/>

1. <wso2am-2.1.0>\wso2am-2.1.0\repository\conf\data-bridge\data-bridge-config.xml

<keyStorePassword>wso2apim</keyStorePassword>

1. **<wso2am-2.1.0>\wso2am-2.1.0\repository\conf\identity\EndpointConfig.properties**

**Carbon.Security.KeyStore.Password=wso2apim**

1. **<wso2am-2.1.0>\wso2am-2.1.0\repository\conf\identity\identity.xml**

**Uncomment & change as follows**

<SkipDBSchemaCreation>true</SkipDBSchemaCreation>

<KeyStore>

<Password>wso2apim</Password>

1. **<wso2am-2.1.0>\wso2am-2.1.0\repository\conf\datasources\** **master-datasources.xml**

Change default h2 database

1. **<wso2am-2.1.0>\wso2am-2.1.0\repository\conf\registry.xml**

**Add new tag to match with registry db newly added in master-datasource.xml**

****

1. **<wso2am-2.1.0>\wso2am-2.1.0\repository\conf\user-mgt.xml**

<AdminUser>

<UserName>OasysUser</UserName>

<Password>oasysUsr123$</Password>

</AdminUser>

# Not Certificate Found in Chrome

<https://security.stackexchange.com/questions/131684/new-ssl-cert-implementation-showing-errors-on-android-only>

|  |
| --- |
| We managed to solve this issue. Our webserver is Tomcat and the fix was to follow the steps below:  Backup \repository\conf\tomcat\catalina-server.xml  Shutdown the Tomcat Server  Modify the catalina-server.xml file by changing the setting clientAuth=want to clientAuth=false  Start the Tomcat Server.  Ensure the Tomcat server is started correctly. |

# renewing CA issued certificate chain in jks

|  |
| --- |
| This is only required when the certificate chain is checked in proxy ,firewalls.  keytool -importkeystore -srckeystore wso2carbon.jks -destkeystore mykeystore.p12 -deststoretype PKCS12 :convert the old jks to a p12 format  openssl pkcs12 -in mykeystore.p12 -nokeys -out cert.pem :export the certificate  --export the private key and the certificate from the pfx  --do the replacement in the cert.pem file with the new cert values    openssl pkcs12 -export -in cert.pem -inkey key.pem -certfile cert.pem -out testkeystore.p12 :pass the new private key    keytool -importkeystore -srckeystore testkeystore.p12 -srcstoretype pkcs12 -destkeystore wso2carbon.jks -deststoretype JKS :create a new jks with old credentials |

# Add New .jks using .pfx file

**Step 1.**

Download pfx from azure portal using shell script

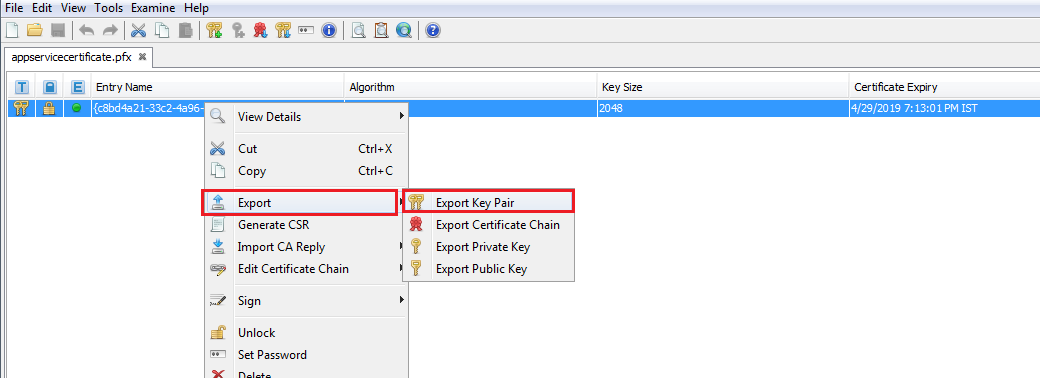
|  |
| --- |
| $appServiceCertificateName = "oasys-ssl"  $resourceGroupName = "SSL-Certificates"  $azureLoginEmailId = "OASys.Live.Host@outlook.com"  $subscriptionId = "66868cec-1509-428d-89e4-7f6ab79bcf4c"  Login-AzureRmAccount  Set-AzureRmContext -SubscriptionId $subscriptionId  $ascResource = Get-AzureRmResource -ResourceName $appServiceCertificateName -ResourceGroupName $resourceGroupName -ResourceType "Microsoft.CertificateRegistration/certificateOrders" -ApiVersion "2015-08-01"  $keyVaultId = ""  $keyVaultSecretName = ""  $certificateProperties=Get-Member -InputObject $ascResource.Properties.certificates[0] -MemberType NoteProperty  $certificateName = $certificateProperties[0].Name  $keyVaultId = $ascResource.Properties.certificates[0].$certificateName.KeyVaultId  $keyVaultSecretName = $ascResource.Properties.certificates[0].$certificateName.KeyVaultSecretName  $keyVaultIdParts = $keyVaultId.Split("/")  $keyVaultName = $keyVaultIdParts[$keyVaultIdParts.Length - 1]  $keyVaultResourceGroupName = $keyVaultIdParts[$keyVaultIdParts.Length - 5]  Set-AzureRmKeyVaultAccessPolicy -ResourceGroupName $keyVaultResourceGroupName -VaultName $keyVaultName -UserPrincipalName $azureLoginEmailId -PermissionsToSecrets get  $secret = Get-AzureKeyVaultSecret -VaultName $keyVaultName -Name $keyVaultSecretName  $pfxCertObject=New-Object System.Security.Cryptography.X509Certificates.X509Certificate2 -ArgumentList @([Convert]::FromBase64String($secret.SecretValueText),"", [System.Security.Cryptography.X509Certificates.X509KeyStorageFlags]::Exportable)  $pfxPassword = -join ((65..90) + (97..122) + (48..57) | Get-Random -Count 50 | % {[char]$\_})  $currentDirectory = (Get-Location -PSProvider FileSystem).ProviderPath  [Environment]::CurrentDirectory = (Get-Location -PSProvider FileSystem).ProviderPath  [io.file]::WriteAllBytes(".\appservicecertificate.pfx", $pfxCertObject.Export([System.Security.Cryptography.X509Certificates.X509ContentType]::Pkcs12, $pfxPassword))  Write-Host "Created an App Service Certificate copy at: $currentDirectory\appservicecertificate.pfx"  Write-Warning "For security reasons, do not store the PFX password. Use it directly from the console as required."  Write-Host "PFX password: $pfxPassword" |

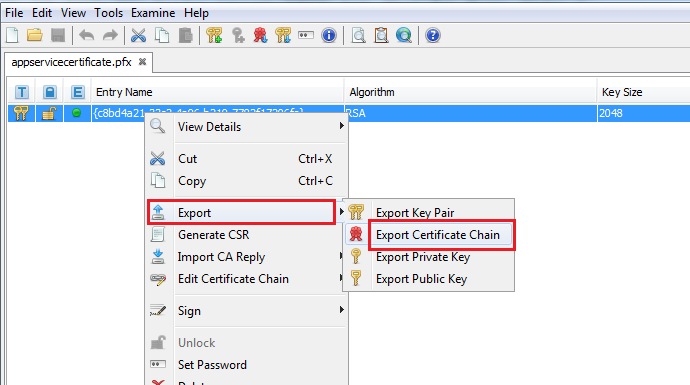
PFX **Password: *0wdva8YxWn41N3zbkKhE9MLIFUgmHsj2ZVPtfGicTqRoAJl7DB***

**Step 2.**

Using exported .pfx file we can get key pair and certificate that we need to create Keystore .jks .

By using keytool explorer

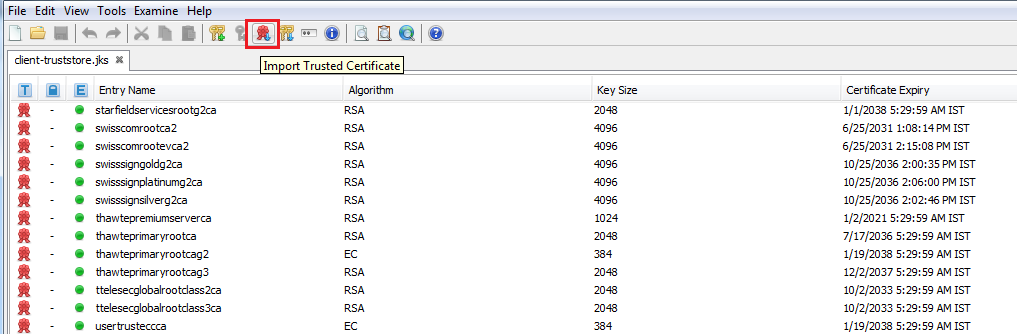




**Step 3**

Exported certificate in previous step import to **client-truststore**

**Note \*** If existing trusted certificate alias & newly created certificate alias same, it should be removed previous certificate before import new one.



Then Replace wso2carbon.jks & client-trustore with updated one.

**Note\*** Follow these steps to update new keystore password & allias.

|  |
| --- |
| If you want an easy way to locate all the configuration files that have references to keystores, you can use the grep command as follows:   1. Open a command prompt and navigate to the <PRODUCT\_HOME>/repository/conf/ directory where your product stores all configuration files. 2. Execute the following command: grep -nr ".jks" . |

**Finally you have to give new certificate to Update ADFS relying party.**

# Create .pem File for haproxy

**[pfx password:] *0wdva8YxWn41N3zbkKhE9MLIFUgmHsj2ZVPtfGicTqRoAJl7DB***

**Step 1**

Install [Win32 OpenSSL](http://www.slproweb.com/products/Win32OpenSSL.html).

Download Link : <http://slproweb.com/products/Win32OpenSSL.html>

Before Install OpenSSL you have to install [Microsoft Visual C++ 2008 Redistributable Package (x86)](http://www.microsoft.com/downloads/details.aspx?FamilyID=9B2DA534-3E03-4391-8A4D-074B9F2BC1BF&displaylang=en).

Link: <https://www.microsoft.com/en-us/download/details.aspx?id=29>

Document Link : <https://support.citrix.com/article/CTX136444>

**Step 2**

Use following commands to export key and certificate

|  |
| --- |
| source: http://www.markbrilman.nl/2011/08/howto-convert-a-pfx-to-a-seperate-key-crt-file/  `**openssl pkcs12 -in [yourfile.pfx] -nocerts -out [keyfile-encrypted.key]**`  What this command does is extract the private key from the .pfx file. Once entered you need to type in the importpassword of the .pfx file. This is the password that you used to protect your keypair when you created your .pfx file. If you cannot remember it anymore you can just throw your .pfx file away, cause you won’t be able to import it again, anywhere!. Once you entered the import password OpenSSL requests you to type in another password, twice!. This new password will protect your .key file.  Now let’s extract the certificate:  `**openssl pkcs12 -in [yourfile.pfx] -clcerts -nokeys -out [certificate.crt]**`  Just press enter and your certificate appears.  Now as I mentioned in the intro of this article you sometimes need to have an unencrypted .key file to import on some devices. I probably don’t need to mention that you should be carefully. If you store your unencrypted keypair somewhere on an unsafe location anyone can have a go with it and impersonate for instance a website or a person of your company. So always be extra careful when it comes to private keys! Just throw the unencrypted keyfile away when you’re done with it, saving just the encrypted one.  The command:  `**openssl rsa -in [keyfile-encrypted.key] -out [keyfile-decrypted.key]**`  Notes:  - When you first extract the key, apply a new password (probably the same as you used to extract it) and then create an unencrypted key with the rsa command above  - Use an encrypted key file for NGINX otherwise it'll ask for the password every time it is restarted.  - Check the top of the extract .crt file for extra bits above the ----BEING... line and remove if necessary  - This certificated needs to be concatenated with the full chain of certificate authorities `cat domain.crt CA\_bundle.crt > final.crt`  - test the cert with `openssl s\_client -showcerts -connect www.domain.com:443` |

Then combine both certificate & key file in text editor & save as [Yourname.pem]

# Add New User Store

1. Copy following jar file <apiHome>/repository/components/dropins folder

*com.wso2.carbon.custom.user.store.manager-2.0.0.jar*

*This jar file applicable for wso2 APIM 2.1*

*com.wso2.carbon.custom.user.store.manager-1.0.0.jar FILE APPLICABLE FOR*

*APIM 1.10*

1. Go to <apihome>/identity/User Stores and click on add new userStore

|  |  |
| --- | --- |
| User Store Manager Class |  |
| Domain Name\* | oasys |
| Driver Name \* | com.microsoft.sqlserver.jdbc.SQLServerDriver |
| Connection URL \* | jdbc:sqlserver://oasys-dev-services.azurewebsites.net:1433;database=OASys\_DEV\_GlobalServiceDesk;socketTimeout=0 |
| User Name \* | OASYsDEVUsr |
| Password \* | 0a5ys@usr |
| Advanced tab: | Change property value: view name as VIEW\_USERSTORE |

|  |  |
| --- | --- |
| |  | | --- | |  | |

1. *Go to api management console and navigate to*

*Home/identity/users and roles/list/roles*

***Then list down all user roles and click permission action on* Internal/everyone and make sure to give login permission.**

After adding to the server, you can edit the properties of the new secondary user store and enable/disable it in a dynamic manner.  
This will be saved to an XML file with the same name as the domain name,

under  <PRODUCT\_HOME>/repository/deployment/server/userstores directory for super tenant and <PRODUCT\_HOME>/repository/tenants/<tenantid>/userstores directory fortenant.

# Writing up custom user store jar file

https://docs.wso2.com/display/IS540/Writing+a+Custom+User+Store+Manager

# Create Crontab Services to touch custom userstore db connection

1. sudo crontab -e

\*/15 \* \* \* \* /home/VmAdmin/ref.sh

2. Ref.sh

touch /home/VmAdmin/wso2am-2.1.0/repository/deployment/server/userstores/\*.xml

Then we can use following command for giving touch for user store db connection

**>$ touch ref.sh**

# Create daemon services

<http://legendans.blogspot.com/2015/06/how-to-write-systemd-service-to-start.html>

<http://blog.lasindu.com/2014/10/how-to-install-wso2-server-as-linux.html>

<http://sumedhask.blogspot.com/2013/10/start-wso2-server-as-linux-service.html>

Create wso2apimService.sh file and move to **[/home/devvmadmin/wso2am-2.1.0/bin]** directory

|  |
| --- |
| #!/bin/bash  # chkconfig: 2345 95 20  # description: Deamon Service to Start up stopped services at server startup  # processname: wso2asService  # --------------------------------------------------------------  #  # 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.  #  # --------------------------------------------------------------  # This service script will be executed to start the servers.  # --------------------------------------------------------------  #  USER="devvmadmin" #eg: ubuntu  PRODUCT\_CODE="APIM" #eg: CEP  CARBON\_HOME="/home/devvmadmin/wso2am-2.1.0/" #eg: /mnt/10.0.0.1/wso2esb-4.9.0  LOCK\_FILE="${CARBON\_HOME}/wso2carbon.lck"  PID\_FILE="${CARBON\_HOME}/wso2carbon.pid"  CMD="${CARBON\_HOME}/bin/wso2server.sh" #eg: ${CARBON\_HOME}/bin/wso2server.sh  JAVA\_HOME=" /home/devvmadmin/jdk1.7.0\_60" #eg: /usr/java/default  export JAVA\_HOME=$JAVA\_HOME  # Status the service  status() {  if [ -f $PID\_FILE ]  then  PID=`cat $PID\_FILE`  ps -fp $PID > /dev/null 2>&1  PIDVAL=$?  else  PIDVAL=3  fi  if [ $PIDVAL -eq 0 ]  then  echo "WSO2 $PRODUCT\_CODE server is running ..."  else  echo "WSO2 $PRODUCT\_CODE server is stopped."  fi  return $PIDVAL  }  # Start the service  start() {  if [ -f $PID\_FILE ]  then  PID=`cat $PID\_FILE`  ps -fp $PID > /dev/null 2>&1  PIDVAL=$?  else  PIDVAL=3  fi  if [ $PIDVAL -eq 0 ]  then  echo "WSO2 $PRODUCT\_CODE server is running ..."  else  echo -n "Starting WSO2 $PRODUCT\_CODE server: "  touch $LOCK\_FILE  su - $USER -c "$CMD start > /dev/null 2>&1 &"  sleep 5  if [ -f $PID\_FILE ]  then  PID=`cat $PID\_FILE`  ps -fp $PID > /dev/null 2>&1  PIDVAL=$?  if [ $PIDVAL -eq 0 ]  then  echo "success"  else  echo "failure"  fi  else  echo "failure"  PIDVAL=2  fi  fi  echo  return $PIDVAL  }  # Restart the service  restart() {  echo -n "Restarting WSO2 $PRODUCT\_CODE server: "  touch $LOCK\_FILE  su - $USER -c "$CMD restart > /dev/null 2>&1 &"  sleep 15  if [ -f $PID\_FILE ]  then  PID=`cat $PID\_FILE`  ps -fp $PID > /dev/null 2>&1  PIDVAL=$?  if [ $PIDVAL -eq 0 ]  then  echo "success"  else  echo "failure"  fi  else  echo "failure"  PIDVAL=2  fi  echo  return $PIDVAL  }  # Stop the service  stop() {  if [ -f $PID\_FILE ]  then  PID=`cat $PID\_FILE`  ps -fp $PID > /dev/null 2>&1  PIDVAL=$?  if [ $PIDVAL -eq 0 ]  then  echo -n "Stopping WSO2 $PRODUCT\_CODE server: "  su - $USER -c "$CMD stop > /dev/null 2>&1 &"  rm -f $LOCK\_FILE  sleep 10  PID=`cat $PID\_FILE`  ps -fp $PID > /dev/null 2>&1  PIDVAL=$?  if [ $PIDVAL -eq 0 ]  then  echo "failure"  PIDVAL=2  else  echo "success"  PIDVAL=0  fi  else  echo "WSO2 $PRODUCT\_CODE server is not running."  PIDVAL=0  fi  else  echo "WSO2 $PRODUCT\_CODE server is not running."  PIDVAL=0  fi  echo  return $PIDVAL  }  ### main logic ###  case "$1" in  start)  start  ;;  stop)  stop  ;;  status)  status  ;;  restart|reload|condrestart)  restart  ;;  \*)  echo $"Usage: $0 {start|stop|restart|reload|status}"  exit 1  esac  exit $? |

**Create wso2apim.service file on linux and move to /etc/systemd/system directory**

|  |
| --- |
| [Unit]  Description=Wso2 API Manager  After=syslog.target network.target  [Service]  Type=oneshot  ExecStart= /home/devvmadmin/wso2am-2.1.0/bin/wso2apimService.sh start  ExecStop= /home/devvmadmin/wso2am-2.1.0/bin/wso2apimService.sh stop  RemainAfterExit=yes  StandardOutput=syslog  StandardError=syslog  [Install]  WantedBy=multi-user.target |

**After creating service**

**Create system link with service file**

**Command : sudo systemctl enable wso2esb.service -–now**

**sudo systemctl enable wso2apim.service –now**

**To list down all enabled system link service files**

**systemctl list-unit-files --type=service**

Go to /etc/systemd/system and check whether newly created service working or not

sudo systemctl status wso2apim.service

sudo systemctl start wso2apim.service

sudo systemctl stop wso2apim.service

Note: view log

Cd /home/devvmadmin/wso2am-2.1.0/repository/logs

tail 1000f wso2carbon.log

# Create Service Provider

|  |  |
| --- | --- |
| Service provider Name | LOLCAdfsSp |
| Description | Service provider for LOLC adf |
| Inbound Authentication Configuration/ [OAuth/OpenID Connect Configuration](https://apidev.oasys.lk:9443/carbon/application/configure-service-provider.jsp?spName=LOLCAdfsSp) Callback Url\* | <https://apidev.oasys.lk:9443/carbon/>  OAuth Client Key  uo7gBYHg\_qA7yAQdmh6otn5MUfwa |
|  |  |
|  |  |
|  |  |
|  |  |

# Create Identity Provider

https://omindu.wordpress.com/2015/06/19/setting-ad-fs-3-0-as-federated-authenticator-in-wso2-identity-server/

|  |  |
| --- | --- |
| Identity provider name | LOLCADFS |
| Identity Provider Public Certificate: | Add adfs certificate |
| [Federated Authenticators](https://apidev.oasys.lk:9443/carbon/idpmgt/idp-mgt-edit.jsp?idPName=LOLCADFS)[SAML2 Web SSO Configuration](https://apidev.oasys.lk:9443/carbon/idpmgt/idp-mgt-edit.jsp?idPName=LOLCADFS) |  |
| Enable SAML2 Web SSO | true |
| Service Provider Entity Id:\* | wso2km21AzureDev |
| Identity Provider Entity Id:\* | <http://sts.lolc.com/adfs/services/trust> |
| SSO [URL:\*](file:///C:\Users\Administrator\AppData\Roaming\Microsoft\Word\*) | <https://sts.lolc.com/adfs/ls> |
| Enable Logout | true |
| Logout Url: | <https://sts.lolc.com/adfs/ls> |
| Enable Logout Request Signing | true |
| Signature Algorithm: | DSA with SHA1 |
| Include Public Certificate | true |
| Include Protocol Binding | true |
| Include NameID Policy | true |
|  |  |

 create a new relying party for the below client\_id. Use below parameters and the attached certificate

|  |  |
| --- | --- |
| Client\_id | uo7gBYHg\_qA7yAQdmh6otn5MUfwa |
| Client\_id redirect URL | https://apidev.oasys.lk:9443/carbon/ |
| Commonauth endpoint | https://apidev.oasys.lk:9443/token |
| Relying party trust identifier | wso2km21AzureDev |
| Certificate |  |

# CREATE PROXY SERVICE FOR MAIN API SERVICE

|  |
| --- |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<proxy xmlns="http://ws.apache.org/ns/synapse"**  **name="mainOASYSService1\_0Rest"**  **startOnLoad="true"**  **statistics="enable"**  **trace="enable"**  **transports="https,http">**  **<target>**  **<inSequence>**  **<log level="full"/>**  **<filter regex="OPTIONS" source="get-property('axis2', 'HTTP\_METHOD')">**  **<then>**  **<property name="Access-Control-Request-Headers"**  **scope="transport"**  **value="authorization,content-type"/>**  **<property name="Access-Control-Allow-Headers"**  **scope="transport"**  **value="authorization,Access-Control-Allow-Origin,Content-Type,X-Requested-With,Accept"/>**  **<property name="Access-Control-Allow-Methods"**  **scope="transport"**  **value="GET,POST,PUT,DELETE,OPTIONS"/>**  **<property name="Access-Control-Allow-Origin" scope="transport" value="\*"/>**  **<property name="RESPONSE" scope="default" type="STRING" value="true"/>**  **<respond/>**  **</then>**  **</filter>**  **<send>**  **<endpoint>**  **<address uri="add end point url"/>**  **</endpoint>**  **</send>**  **</inSequence>**  **<outSequence>**  **<send/>**  **</outSequence>**  **</target>**  **<description/>**  **</proxy>** |

**Proxy end point uri :** <https://oasys-dev-services.azurewebsites.net/>

For testing purpose use: /Definitions/SecurityService.svc/validatelogin

# API CREATION

There are 3 admin services & main Oasys service available.

Admin Services

* mainOASYSOAuthTokenRemTime1\_0Rest{POST} :/{need access token}
* OasysIsExistingUID1\_0Rest {POST}:/ none [no need access token]
* utilityComnLoginIDS1\_0Rest {POST}:/ none [no need access token]

Main Service

* mainOASYSService1\_0Rest {POST}:/\*

Log on API PUBLISHER and create New rest apis as follows

Get token remaining time

|  |  |
| --- | --- |
| Visibility | Public |
| Context | /oAuthTokenRemTime/1.0 |
| Production URL | https://apidev.oasys.lk:9443/services/OAuth2TokenValidationService |
| Sandbox URL | https://apidev.oasys.lk:9443/services/OAuth2TokenValidationService |
| Date Last Updated | 12/7/2017, 10:35:41 AM |
| Tier Availability | Unlimited |
| Default API Version | None |
| Business Owner | OASYS |
| Published Environments | Production and Sandbox |

Description

Is Exists user

|  |  |
| --- | --- |
| Visibility | Public |
| Context | /oasysIsExistingUID/1.0 |
| Production URL | https://apidev.oasys.lk:9443/services/RemoteUserStoreManagerService |
| Sandbox URL | https://apidev.oasys.lk:9443/services/RemoteUserStoreManagerService |
| Date Last Updated | 12/7/2017, 10:53:05 AM |
| Tier Availability | Unlimited |
| Default API Version | None |
| Business Owner | OASYS |
| Published Environments | Production and Sandbox |

Description

utility common login

|  |  |
| --- | --- |
| Visibility | Public |
| Context | /utilityComnLoginIDS/1.0 |
| Production URL | https://apidev.oasys.lk:9443/services/AuthenticationAdmin |
| Sandbox URL | https://apidev.oasys.lk:9443/services/AuthenticationAdmin |
| Date Last Updated | 12/7/2017, 11:28:41 AM |
| Tier Availability | Unlimited |
| Default API Version | None |
| Business Owner | OASYS |
| Published Environments | Production and Sandbox |

Description

main oasys service

|  |  |
| --- | --- |
| Visibility | Public |
| Context | /mainOASYSService/1.0 |
| Production URL | http://DEV-Middleware:8380/services/mainOASYSService1\_0Rest |
| Sandbox URL | http://DEV-Middleware:8380/services/mainOASYSService1\_0Rest |
| Date Last Updated | 12/7/2017, 10:36:07 AM |
| Tier Availability | Unlimited |
| Default API Version | None |
| Business Owner | OASYS |
| Published Environments | Production and Sandbox |

IN AND OUT SEQUENCE **AS FOLLOWS**

* **mainOASYSOAuthTokenRemTime1\_0Rest**

**IN**

|  |
| --- |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<sequence xmlns="http://ws.apache.org/ns/synapse" name="in-endpoint">**  **<property name="token" expression="//token"></property>**  **<payloadFactory media-type="xml">**  **<format>**  **<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://org.apache.axis2/xsd" xmlns:xsd1="http://dto.oauth2.identity.carbon.wso2.org/xsd">**  **<soapenv:Header></soapenv:Header>**  **<soapenv:Body>**  **<xsd:findOAuthConsumerIfTokenIsValid>**  **<xsd:validationReqDTO>**  **<xsd1:accessToken>**  **<xsd1:identifier>$1</xsd1:identifier>**  **<xsd1:tokenType>Bearer</xsd1:tokenType>**  **</xsd1:accessToken>**  **</xsd:validationReqDTO>**  **</xsd:findOAuthConsumerIfTokenIsValid>**  **</soapenv:Body>**  **</soapenv:Envelope>**  **</format>**  **<args>**  **<arg evaluator="xml" expression="$ctx:token"></arg>**  **</args>**  **</payloadFactory>**  **<property name="Authorization" expression="fn:concat('Basic ',base64Encode('OasysUser:oasysUsr123$'))" scope="transport"></property>**  **</sequence>** |

**OUT**

|  |
| --- |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<sequence xmlns="http://ws.apache.org/ns/synapse" name="out-endpoint">**  **<payloadFactory media-type="json">**  **<format>{"Token": "$1","RemainingTime": $2} </format>**  **<args>**  **<arg evaluator="xml" expression="$ctx:token"></arg>**  **<arg xmlns:ax2385="http://dto.oauth2.identity.carbon.wso2.org/xsd" xmlns:ns="http://org.apache.axis2/xsd" evaluator="xml" expression="//ax2385:expiryTime"></arg>**  **<arg xmlns:ax2385="http://dto.oauth2.identity.carbon.wso2.org/xsd" xmlns:ns="http://org.apache.axis2/xsd" evaluator="xml" expression="//ax2385:authorizedUser"></arg>**  **</args>**  **</payloadFactory>**  **<property name="messageType" value="application/json" scope="axis2"></property>**  **</sequence>** |

* **OasysIsExistingUID1\_0Rest**

**IN**

|  |
| --- |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<sequence xmlns="http://ws.apache.org/ns/synapse" name="isExistingUIDIn">**  **<log level="full"></log>**  **<property name="userName" expression="//userName"></property>**  **<property name="authorization" expression="get-property('transport', 'Authorization')"></property>**  **<payloadFactory media-type="xml">**  **<format>**  **<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ser="http://service.ws.um.carbon.wso2.org">**  **<soapenv:Header></soapenv:Header>**  **<soapenv:Body>**  **<ser:isExistingUser>**  **<ser:userName>$1</ser:userName>**  **</ser:isExistingUser>**  **</soapenv:Body>**  **</soapenv:Envelope>**  **</format>**  **<args>**  **<arg evaluator="json" expression="$.userName"></arg>**  **</args>**  **</payloadFactory>**  **<property name="Authorization" expression="fn:concat('Basic ',base64Encode('OasysUser:oasysUsr123$'))" scope="transport"></property>**  **<log level="custom">**  **<property name="Authz" expression="get-property('transport','Authorization')"></property>**  **</log>**  **<send>**  **<endpoint>**  **<address uri="https://apidev.oasys.lk:9443/services/RemoteUserStoreManagerService" format="soap12"></address>**  **</endpoint>**  **</send>**  **</sequence>** |

**OUT**

|  |
| --- |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<sequence xmlns="http://ws.apache.org/ns/synapse" name="isExistingUIDOut">**  **<log level="full"></log>**  **<property name="messageType" value="application/json" scope="axis2"></property>**  **<send></send>**  **</sequence>** |

* **utilityComnLoginIDS1\_0Rest**

|  |
| --- |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<sequence xmlns="http://ws.apache.org/ns/synapse" name="in-endpoint">**  **<log level="full" category="DEBUG"></log>**  **<property name="userName" expression="//user/userName"></property>**  **<property name="passWord" expression="//user/passWord"></property>**  **<property name="authorization" expression="get-property('transport', 'Authorization')"></property>**  **<payloadFactory media-type="xml">**  **<format>**  **<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:aut="http://authentication.services.core.carbon.wso2.org">**  **<soapenv:Header></soapenv:Header>**  **<soapenv:Body>**  **<aut:login>**  **<!--Optional:-->**  **<aut:username>$1</aut:username>**  **<!--Optional:-->**  **<aut:password>$2</aut:password>**  **<!--Optional:-->**  **<aut:remoteAddress></aut:remoteAddress>**  **</aut:login>**  **</soapenv:Body>**  **</soapenv:Envelope>**  **</format>**  **<args>**  **<arg evaluator="xml" expression="$ctx:userName"></arg>**  **<arg evaluator="xml" expression="$ctx:passWord"></arg>**  **</args>**  **</payloadFactory>**  **<property name="Authorization" expression="fn:concat('Basic ',base64Encode('OasysUser:oasysUsr123$'))" scope="transport" type="STRING"></property>**  **<log level="custom">**  **<property name="Authz" expression="get-property('transport','Authorization')"></property>**  **<property name="userName" expression="$ctx:userName"></property>**  **<property name="passWord" expression="$ctx:passWord"></property>**  **</log>**  **</sequence>** |

**OUT**

|  |
| --- |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<sequence xmlns="http://ws.apache.org/ns/synapse" name="out-endpoint">**  **<log level="full"></log>**  **<property name="messageType" value="application/json" scope="axis2"></property>**  **</sequence>** |

* **mainOASYSService1\_0Rest**

**IN SEQUNCE ONLY [JSONP CONVERSION]**

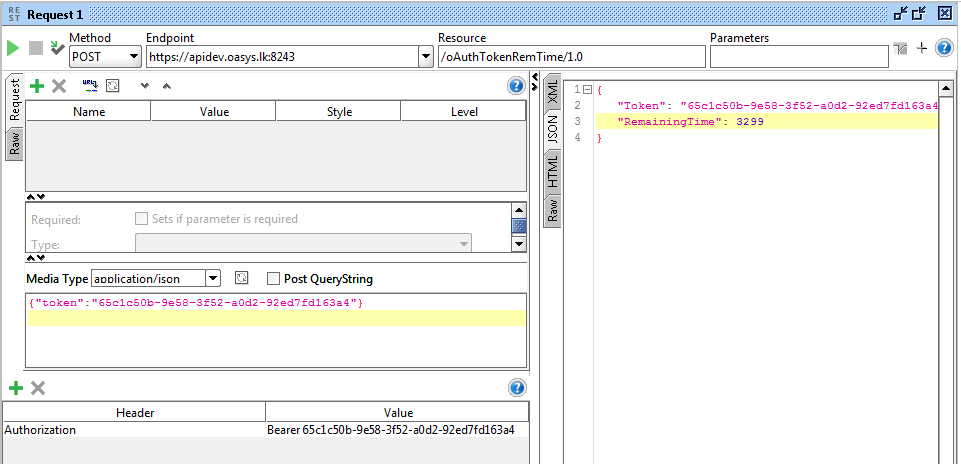
|  |
| --- |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<sequence xmlns="http://ws.apache.org/ns/synapse" name="default-endpoint">**  **<property name="ContentType" value="application/jsonp" scope="axis2"/>**  **<header name="To" expression="get-property('ContentType')"/>**  **</sequence>** |

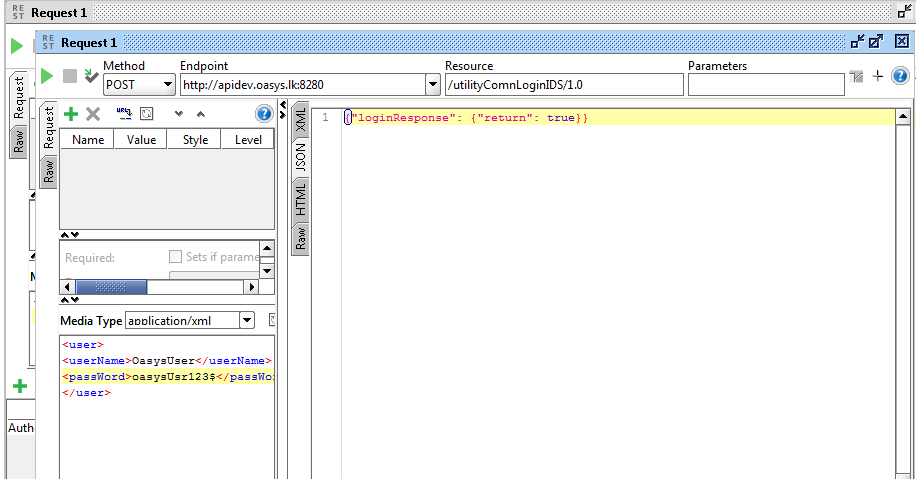
# Test Services

**Create 3 applications as mentioned follows & subscribe both 2 apis (*1.mainOASYSService1\_0Rest and 2.mainOASYSOAuthTokenRemTime1\_0Rest*)**

**For all applications**

* **OasysADApp**
* **OasysAdfsApp**
* **OasysDBApp**

****

****

**Token Requst**

**Uri :** [**https://apidev.oasys.lk:8243**](https://apidev.oasys.lk:8243)

**In Request Header add followings**

|  |  |
| --- | --- |
| **username** | **OasysUser** |
| **password** | **oasysUsr123$** |
| **client\_id** | **Enter <Consumer Key>** |
| **client\_secret** | **Enter <Consumer Secret>** |
| **grant\_type** | **password** |
| **scope** | **bearer** |

**Then result will give as follows**

|  |
| --- |
| {  "scope": "default",  "token\_type": "Bearer",  "expires\_in": 1557,  "refresh\_token": "e8f061bb-bf94-3756-899a-45bc87e661af",  "access\_token": "348b2683-c22f-36d5-964c-6f98a1f782d3"  } |

# Final Result

|  |  |
| --- | --- |
| API Service | https://apidev.oasys.lk:443/mainOASYSService/1.0 |
| Token URL | https://apidev.oasys.lk:443/token |
| API for Token remaining time | https://apidev.oasys.lk:443/oAuthTokenRemTime/1.0 |
| Utility Common Login | https://apidev.oasys.lk:443/utilityComnLoginIDS/1.0 |
| User Existence checking service | http://apidev.oasys.lk:443/oasysIsExistingUID/1.0 |
| Revoke URL | https://apidev.oasys.lk:443/revoke |

DB Login

|  |  |
| --- | --- |
| Client\_id | XwLO9JGkAtpi0DBJSsAz7uFgvD0a |
| Client\_secret | CEIGC836RCbZZaMC5x\_wohny4qwa |

AD Login

|  |  |
| --- | --- |
| Client\_id | r7fm\_bS588cn1Idbp9m8H6GMj\_Qa |
| Client\_secret | 1v0GW56gc4Ja7tOaJfGeWbwxOeca |

ADFS Login

|  |  |
| --- | --- |
| Client\_id | AwODTvIOEA7NKi6cSMB1wx\_VmLka |
| Client\_secret | ffboQRzSQgBYL36zkv5Fd6hwjxYa |

# Test ADFS Login with SAML Assertion

Using Postman

Uri : <https://apidev.oasys.lk:443/token>

**Add following Headers:**

|  |  |
| --- | --- |
| Content-Type | application/x-www-form-urlencoded |
| Authorization | Basic <base64 adfs clientID:clientSecret> |

Under body add following parameters

|  |  |
| --- | --- |
| grant\_type | urn:ietf:params:oauth:grant-type:saml2-bearer |
| assertion | Generated assertion text <base64> |

**Result**

|  |
| --- |
| {  "scope": "default",  "token\_type": "Bearer",  "expires\_in": 3600,  "refresh\_token": "ea14cef9-847a-3f5a-87ae-0ff8eb1a0252",  "access\_token": "62c99297-8ba8-3b0a-9361-e6e82b6fdd96"  } |

# Tools

**1. PuTTY -** OpenSSH, **PuTTY** is a very versatile tool for remote access to another computer. It's probably used more often by people who want secure remote shell access to a UNIX or Linux system than for any other purpose, though that is only one of its many uses. **PuTTY** is more than just an SSH client

**2. WinSCP -** (Windows Secure Copy) is a free and open-source SFTP, FTP, WebDAV and SCP client for Microsoft Windows. Its main function is secure file transfer between a local and a remote computer.

**3. SoapUI-5.3.0** - is an open-source web service testing application for service-oriented architectures (SOA) and representational state transfers (REST). Its functionality covers web service inspection, invoking, development, simulation and mocking, functional testing, load and compliance testing.

# Reference:

|  |  |
| --- | --- |
|  |  |
| 01: | when user password have the "@" mark, JMS client's fail to create a connection with server  https://wso2.org/jira/browse/MB-258?focusedCommentId=81072&page=com.atlassian.jira.plugin.system.issuetabpanels%3Acomment-tabpanel#comment-81072 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Sample code to read header token values using C#.NET

|  |
| --- |
| namespace ABC { public partial class \_Default : System.Web.UI.Page { protected void Page\_Load(object sender, EventArgs e) {  try { if (!Page.IsPostBack) {  //store the value of claim :http://wso2.org/claims/lastname String lastName = "";  //check for X-JWT-Assertion parameter if (Request.Headers["X-JWT-Assertion"] != null) { reqHeader = Request.Headers["X-JWT-Assertion"].ToString(); string[] stringSeperator = new string[] { "=." }; //split the encoded string and send to decode string decodedHeader = base64Decode(reqHeader.Split(stringSeperator, StringSplitOptions.None)[1].ToString() + "="); decodedHeader = decodedHeader.Replace("\"", "'"); //format json string JObject obj = new JObject(); obj = (JObject)JsonConvert.DeserializeObject(decodedHeader); //Decode Object  if (obj["http://wso2.org/claims/lastname"] != null) { lastName = obj["http://wso2.org/claims/lastname"].ToString(); }  }  } } catch (Exception ex) { //lblErr.Text = "Error: " + ex.Message; }  }  //base64 decodes the given encoded string (data :encoded string) public string base64Decode(string data) { try { System.Text.UTF8Encoding encoder = new System.Text.UTF8Encoding(); System.Text.Decoder utf8Decode = encoder.GetDecoder(); byte[] todecode\_byte = Convert.FromBase64String(data); int charCount = utf8Decode.GetCharCount(todecode\_byte, 0, todecode\_byte.Length); char[] decoded\_char = new char[charCount]; utf8Decode.GetChars(todecode\_byte, 0, todecode\_byte.Length, decoded\_char, 0); string result = new String(decoded\_char); return result; } catch (Exception e) { throw new Exception("Error in base64Decode" + e.Message); } } } } |

# Haproxy.cfg file

Start Haproxy on boot : [https://tecadmin.net/install-and-configure-haproxy-on-centos/#](https://tecadmin.net/install-and-configure-haproxy-on-centos/)

Check haproxy.cfg file valid or not **/usr/local/sbin/haproxy -c -V -f /etc/haproxy/haproxy.cfg**

## Automatically Start HAProxy Service

# service haproxy start

# chkconfig haproxy on

[**https://tecadmin.net/install-and-configure-haproxy-on-centos/#**](https://tecadmin.net/install-and-configure-haproxy-on-centos/)

**how to install HAPROXY**

[**https://www.upcloud.com/support/haproxy-load-balancer-centos/**](https://www.upcloud.com/support/haproxy-load-balancer-centos/)

|  |
| --- |
| #---------------------------------------------------------------------  # Example configuration for a possible web application. See the  # full configuration options online.  #  # http://haproxy.1wt.eu/download/1.4/doc/configuration.txt  #  #---------------------------------------------------------------------  #---------------------------------------------------------------------  # Global settings  #---------------------------------------------------------------------  global  # to have these messages end up in /var/log/haproxy.log you will  # need to:  #  # 1) configure syslog to accept network log events. This is done  # by adding the '-r' option to the SYSLOGD\_OPTIONS in  # /etc/sysconfig/syslog  #  # 2) configure local2 events to go to the /var/log/haproxy.log  # file. A line like the following can be added to  # /etc/sysconfig/syslog  #  # local2.\* /var/log/haproxy.log  #  log 127.0.0.1 local2  tune.ssl.default-dh-param 4096  stats socket /var/lib/haproxy/stats  stats socket /var/run/haproxy.sock mode 600 level admin  stats timeout 2m  #---------------------------------------------------------------------  # common defaults that all the 'listen' and 'backend' sections will  # use if not designated in their block  #---------------------------------------------------------------------  defaults  mode http  log global  option httplog  option dontlognull  option http-server-close  option forwardfor except 127.0.0.0/8  option redispatch  retries 3  timeout http-request 5m  timeout queue 3m  timeout connect 5000  timeout client 24h  timeout server 24h  timeout http-keep-alive 5m  timeout check 10s  maxconn 6000  #---------------------------------------------------------------------  # main frontend which proxys to the backends  #---------------------------------------------------------------------  frontend https-mgt  bind \*:8543  option tcplog  mode tcp  default\_backend mgt\_backend  backend mgt\_backend  mode tcp  balance source  server smsnode1 localhost:9443 check  frontend https-api  bind \*:443  option tcplog  mode tcp  default\_backend api\_backend  backend api\_backend  mode tcp  balance source  server smsnode1 localhost:8243 check  frontend http\_statts  bind \*:8081  stats enable  stats scope https-mgt  stats scope mgt\_backend  stats scope https-api  stats scope api\_backend  stats uri /haproxy?stats  stats hide-version  stats auth haproxy:stats@lolc |

# [Configuring the Identity Server 5.1.0 as a Key Manager ...](https://docs.wso2.com/display/CLUSTER44x/Configuring+the+Identity+Server+5.1.0+as+a+Key+Manager+with+API+Manager+1.10.0)

Refer link: <https://docs.wso2.com/display/CLUSTER44x/Configuring+the+Identity+Server+5.1.0+as+a+Key+Manager+with+API+Manager+1.10.0>

IDS Changes

1.       Sql driver

2.       identity.xml from API

**3.       api-manager.xml from api**

a.       <GatewayType>None</GatewayType>

b.      <RevokeAPIURL> https://${GATEWAY\_SERVER\_HOST}:{port}/revoke </RevokeAPIURL>

c.       <ServerURL>https://${GATEWAY\_SERVER\_HOST}:{port}/services/</ServerURL>

d.      <EnableThriftServer>false</EnableThriftServer>

4.       master-datasources.xml

5.       registry.xml

**6.       user-mgt.xml**

**7.       Keystore**

**a.       ConfidentityenpointConfig**

**b.      Conftomcatcatalina**

**c.       Carbon**

d.      **Confaxis2**

# **Configure Ids to get the token for APIM**

o   Compile below script in IDS DB

  API\_HOMEdbscriptsapimgtmssql.sql-- Start of API-Mgt Tables --

|  |  |
| --- | --- |
| IDS |  |
| usr-mgt | remove UM  | <Property name="dataSource">jdbc/WSO2CarbonDB</Property> |
| registry | mount\_system config -->comment |
| identity | replace <Name>jdbc/WSO2AM\_DB</Name> withjdbc/WSO2CarbonDB     |   SkipDBSchemaCreation==true |
| master datasourse | remove WSO2UM\_DB | AM\_DB--> IDS |
| Api manager | datasourse name --> jdbc/WSO2AM\_DB |

|  |  |
| --- | --- |
| API |  |
| usr-mgt | remove UM  | <Property name="dataSource"> jdbc/WSO2AM\_DB</Property> |
| registry | mount\_system config -->comment |
| identity | replace <Name>jdbc/WSO2AM\_DB</Name> with jdbc/WSO2CarbonDB |
| master datasourse | remove WSO2UM\_DB |  WSO2CarbonDB -->IDS  | WSO2AM -->IDS |
| Api manager | Comment --> URL(AM\_DB) |  datasourse name --> jdbc/WSO2AM\_DB |

# APIM 1.10 Refresh token Issue

We have to add patch “org.wso2.carbon.identity.oauth\_5.0.7.jar”

Add ALTER IDS Table

You have to ALTER "IDN\_OAUTH2\_ACCESS\_TOKEN" Table

ADD Following Column

TOKEN\_SCOPE VARCHAR(25),

# Revoke token

|  |
| --- |
| curl -k -v -d "token=<refresh\_token\_to\_be\_revoked>" -H "Authorization: Basic <base64 encoded (clientId:clientSecret)>" -H "Content-Type: application/x-www-form-urlencoded" https://apidev.oasys.lk:443/revoke |

Result

|  |
| --- |
| HTTP/1.1 200 OK  RevokedAccessToken: e5c3d922-157f-3778-be72-6782f2e46f28  X-Frame-Options: DENY  AuthorizedUser: OasysUser@carbon.super  X-XSS-Protection: 1; mode=block  RevokedRefreshToken: 64552d22-8bd0-3437-b7af-72291af5650d  Content-Type: text/html  X-Content-Type-Options: nosniff  Pragma: no-cache  Cache-Control: no-store  Date: Wed, 13 Dec 2017 08:14:34 GMT  Transfer-Encoding: chunked  Connection: Keep-Alive |

# Generating a new access token and refresh token

|  |
| --- |
| curl -k -d "grant\_type=refresh\_token&refresh\_token=<retoken>" -H "Authorization: Basic SVpzSWk2SERiQjVlOFZLZFpBblVpX2ZaM2Y4YTpHbTBiSjZvV1Y4ZkM1T1FMTGxDNmpzbEFDVzhh" -H "Content-Type: application/x-www-form-urlencoded" https://apidev.oasys.lk:443/token |

Result will give you like this

|  |
| --- |
| {      "scope":"default",      "token\_type":"Bearer",      "expires\_in":3600,      "refresh\_token":"7ed6bae2b1d36c041787e8c8e2d6cbf8",      "access\_token":"b7882d23f1f8257f4bc6cf4a20633ab1"  } |

# Monitoring WSO2 ESB using JConsole

<https://wso2.com/library/knowledge-base/2011/04/monitoring-wso2-esb-using-jconsole/>

# Monitor Carbon log using a UNIX Cron job

<https://wso2.com/library/articles/2012/10/wso2-esb-examples-best-practises-error-handling-wso2-esb/>

manage log file using DAS

http://supunsetunga.blogspot.com/2016/03/creating-log-dashboard-with-wso2-das.html

# **Automatic shutdown azure VMs to save money**

Automatic shutdown using azure portal without scripting

<https://buildazure.com/2017/03/16/properly-shutdown-azure-vm-to-save-money/>

Schedule Azure VMs shutdown and Startup

<http://support.accops.com/support/solutions/articles/12000026158-scheduled-virtual-machine-shutdown-startup-microsoft-azure-cloud>

Auto Shutdown Azure linux VMs

<https://michaelblouin.ca/blog/2016/04/23/auto-shutdown-azure-linux-vm-inactivity/>

Schedule startup and shutdown of your virtual machines using Azure Automation

<https://ppolyzos.com/2017/08/15/schedule-startup-and-shutdown-of-your-virtual-machines-using-azure-automation/>

Excluding weekends

<http://blogs.objectsharp.com/post/2016/01/06/Excluding-weekends-in-Azure-Automation-Runbooks.aspx>

Using Azure Automation to run VMs during office hours only – using graphical runbooks

<https://blogs.endjin.com/2016/03/using-azure-automation-to-run-vms-during-office-hours-only-using-graphical-runbooks/>

How to run commands at shutdown on Linux

<https://opensource.com/life/16/11/running-commands-shutdown-linux>

How to run script before system shutdown reboot with system (from Riyas..)

<https://davidshen84.appspot.com/blog/2014/2014/ag1zfmRhdmlkc2hlbjg0ckgLEgRCbG9nIj5Ib3cgdG8gcnVuIGEgc2NyaXB0IGJlZm9yZSBzeXN0ZW0gc2h1dGRvd24vcmVib290IHdpdGggU3lzdGVtZAw>

How system works

<http://hackerpublicradio.org/eps.php?id=2134>

<https://community.dynamics.com/ax/b/axology/archive/2016/12/09/automated-startup-and-shutdown-of-azure-vms>

<https://linuxacademy.com/howtoguides/posts/show/topic/19376-leverage-azure-automation-and-automatically-shut-down-vms>

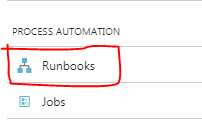
# Understand Scheduler job Details

<https://docs.microsoft.com/en-us/azure/scheduler/scheduler-concepts-terms>

Schedule Auto up down

1st Step

* Log on Azure portal & Create “Automation Account” on Azure
* Then click on Automation Account that you are created and select Runbooks and Add a Runbook .



Once you add run book go to the edit pane and paste following shell script.

|  |
| --- |
| # Input Parameters for  # - VmName: name of the vm to perform action to  # - ResourceGroupName: resource group where the vm belongs to  # - VmAction:action to perform (startup or shutdown)  Param(  [string]$VmName,  [string]$ResourceGroupName,  [ValidateSet("Startup", "Shutdown")]  [string]$VmAction  )    # Authenticate with your Automation Account  $Conn = Get-AutomationConnection -Name AzureRunAsConnection  Add-AzureRMAccount -ServicePrincipal -Tenant $Conn.TenantID `  -ApplicationID $Conn.ApplicationID -CertificateThumbprint $Conn.CertificateThumbprint    #week days normal schedule weekends shutdown  $UTCTime = (Get-Date).ToUniversalTime()  $TZ = [System.TimeZoneInfo]::FindSystemTimeZoneById("Sri Lanka Standard Time")  $LocalTime = [System.TimeZoneInfo]::ConvertTimeFromUtc($UTCTime, $TZ)  $day = $LocalTime.DayOfWeek    if ($day -eq 'Saturday' -or $day -eq 'Sunday')  {          #$vmStatus =Get-AzureRmVM -ResourceGroupName automationResourceGroup -Name 'AUTOVM' -Status  #$displayStatus = $vmStatus.Statuses.DisplayStatus            # Shutdown VM          #if($displayStatus -eq "VM running")          #{              Stop-AzureRmVM -Name $VmName -ResourceGroupName $ResourceGroupName -Force          #}  }  else {      # Startup VM  if($VmAction -eq "Startup")  {  Start-AzureRmVM -Name $VmName -ResourceGroupName $ResourceGroupName  }  # Shutdown VM  #if($VmAction -eq "Shutdown")  else  {  Stop-AzureRmVM -Name $VmName -ResourceGroupName $ResourceGroupName -Force  }  } |

* Next you can click on Schedule icon to add new schedules as you want.
* Here I added two jobs to vmStart and vmShutdown. Make sure to fill parameters correctly.

However there are other third party option to do above task. Please visit <https://vmpower.io/>

# **How to run a script before system shutdown/reboot with Systemd**

Finally you have to add new systemd service to shutdown wso2Esb nad wso2Apim Servers when your shut down schedule run.

First add separate sh.scripts to stop both esb and apim servers. As same as discussed under **Create daemon services**

Then you have to add create two service scripts (esb & apim) under ***[etc/system/system]***

|  |
| --- |
| [Unit]  Description=Wso2 API Manager  DefaultDependencies=no Before=reboot.target  [Service]  Type=oneshot  ExecStart=</home/dinesh1989/wso2esb-5.0.0/bin/shutdownservice.sh> stop  RemainAfterExit=yes  StandardOutput=syslog  StandardError=syslog  [Install]  WantedBy=reboot.target |

Referring link: <https://davidshen84.appspot.com/blog/2014/2014/ag1zfmRhdmlkc2hlbjg0ckgLEgRCbG9nIj5Ib3cgdG8gcnVuIGEgc2NyaXB0IGJlZm9yZSBzeXN0ZW0gc2h1dGRvd24vcmVib290IHdpdGggU3lzdGVtZAw>

# **Manage Log Files**

Refer Link [1]: <https://stackoverflow.com/questions/47335306/how-to-delete-logs-in-wso2-api-manager-periodically>

[2] <http://www.vitharana.org/2015/08/limit-size-of-wso2carbonlog-file.html>

[3] <https://nuwanbando.com/2010/09/25/howto-wso2-carbon-server-logs-to-be-stored-in-a-database/>

# Crontab to delete log files (file older than specific days)

Crontab –e

Then paste following and save

1 \* \* \* \* find /home/VmAdmin/wso2esb-5.0.0/repository/logs/ -mtime 180 -type f –delete

1 \* \* \* \* find /home/vmadmin/wso2am-2.1.0/repository/logs/ -mtime 90 -type f –delete

After that you will see crontab installing

Note that (-mtime 180) file older than 180days

# {HowTo} WSO2 Carbon Server logs to be stored in a database

Thought of documenting this for my own reference, hope this will help. I am also planing to make a knowledge base article out of it to be posted in[WSO2 Oxygen Tank](http://wso2.org/).

As you may know, if your are familiar with [Log4j](http://logging.apache.org/log4j/1.2/), that the logs created with it, can be stored in a database instead of a file. In order to do that you need to add few configuration values to the log4j.properties file.

In-terms of a WSO2 Carbon based server this properties file resides in CARBON\_HOME/lib directory. before adding these configurations there are few steps to visit.

[1] Create database {LOG\_DB}  
[2] Create the log table  {LOGGING} with the following fields

create table LOGGING (

id decimal NOT NULL,

prio varchar(15),

cat varchar(255),

thread varchar(30),

msg varchar(255),

layout\_msg varchar(255),

throwable varchar(2000),

the\_timestamp timestamp);

[3] Create a user and assign to the DB

{user: logger / pwd: logger}

[4] Copy jdbcappender.jar to {CARBON\_HOME}/lib and {CARBON\_HOME}/repository/components/lib along with the database driver

{jdbcappender.jar can be found at http://www.dankomannhaupt.de/projects/jdbcappender/lib/jdbcappender.jar}

Thats it. All preconditions are set, Now open up the log4j.properties file and add the following appender settings to the end.

#log to db appender

log4j.appender.DB\_APPENDER=org.apache.log4j.jdbcplus.JDBCAppender

log4j.appender.DB\_APPENDER.url=jdbc:mysql://localhost:3306/LOG\_DB

log4j.appender.DB\_APPENDER.dbclass=com.mysql.jdbc.Driver

log4j.appender.DB\_APPENDER.username=logger

log4j.appender.DB\_APPENDER.password=logger

log4j.appender.DB\_APPENDER.sql=INSERT INTO LOGGING (id, prio, cat, thread, msg, layout\_msg, throwable,

the\_timestamp) VALUES (@INC@, '@PRIO@', '@CAT@', '@THREAD@', '@MSG@', '@LAYOUT:1@', '@THROWABLE@', '@TIMESTAMP@')

log4j.appender.DB\_APPENDER.layout=org.apache.log4j.PatternLayout

Scroll to log4j.rootLogger and add the appender to the list,

log4j.rootLogger=ERROR, CARBON\_CONSOLE, CARBON\_LOGFILE, CARBON\_MEMORY, CARBON\_SYS\_LOG, DB\_APPENDER

[4] <http://vanjikumaran.blogspot.com/2014/07/write-logs-into-external-database-in.html>

[5] <http://www.vitharana.org/2015/08/limit-size-of-wso2carbonlog-file.html>

[6] Configuring log4j properties: <https://docs.wso2.com/display/ADMIN44x/Configuring+Log4j+Properties>

[7] Monitoring Log wso2 administrative guide: <https://docs.wso2.com/display/ADMIN44x/Monitoring+Logs>

# Working with wso2 DATA ANALYTICS SERVER (DAS 3.0.1)

[1] <https://docs.wso2.com/display/DAS310/Working+with+Product+Specific+Analytics+Profiles>

To do

Vertical vm scale set

Clustering

# Azure vm Auto Scaling

## Create azure vm auto scaling

<https://docs.microsoft.com/en-us/azure/virtual-machines/windows/capture-image-resource>

## deploy application on vmss

<https://docs.microsoft.com/en-us/azure/virtual-machine-scale-sets/virtual-machine-scale-sets-upgrade-scale-set>

## Osura’s blog

<http://osuran.blogspot.com/2016/07/auto-scaling-for-wso2-application.html>

Manage & unmanaged disks in azure

<https://buildwindows.wordpress.com/2017/05/31/azure-managed-vs-unmanaged-disks-the-choice/>

## Schedule Auto Scaling

https://docs.microsoft.com/en-us/azure/virtual-machine-scale-sets/virtual-machine-scale-sets-autoscale-portal

## Deploy a VM Scale Set based on a Linux Custom Image and a script to deploy updates

<https://github.com/dinesh1989/azure-quickstart-templates/tree/master/201-vmss-linux-customimage-autoscale>

Steps

1. Create Base Vm And install all the stuff that you need. Note that when you creating vm you have to select unmanaged disk.
2. Then you can generalize vm by using following commands (azure cli)

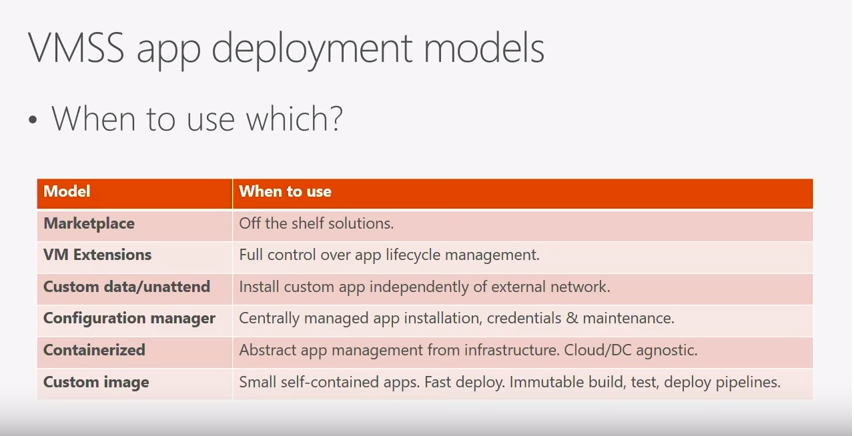
|  |
| --- |
| $vmName = "myVM"  $rgName = "myResourceGroup"  $location = "EastUS"  $imageName = "myImage"  $osVhdUri = "https://mystorageaccount.blob.core.windows.net/vhdcontainer/osdisk.vhd  Stop-AzureRmVM -ResourceGroupName $rgName -Name $vmName -Force  Set-AzureRmVm -ResourceGroupName $rgName -Name $vmName -Generalized  $imageConfig = New-AzureRmImageConfig -Location $location  $imageConfig = Set-AzureRmImageOsDisk -Image $imageConfig -OsType Windows -OsState Generalized -BlobUri $osVhdUri  $image = New-AzureRmImage -ImageName $imageName -ResourceGroupName $rgName -Image $imageConfig |

1. Also you we can use base vm without generalizing (it is really depend on unmanaged disk)
2. After creating base vm you can create vm autoscaling set by using following steps
   * Go to the portal and create new Template deployment
   * Then select build your own template and paste following vmss template and save.

|  |
| --- |
| {  "$schema": "http://schema.management.azure.com/schemas/2015-01-01-preview/deploymentTemplate.json",  "contentVersion": "1.0.0.0",  "parameters": {  "vmSku": {  "defaultValue": "Standard\_A1",  "type": "String",  "metadata": {  "description": "Size of VMs in the VM Scale Set."  }  },  "vmImage": {  "type": "String"  },  "vmssName": {  "maxLength": 61,  "type": "String",  "metadata": {  "description": "String used as a base for naming resources. Must be 3-61 characters in length and globally unique across Azure. A hash is prepended to this string for some resources, and resource-specific information is appended."  }  },  "instanceCount": {  "maxValue": 100,  "type": "Int",  "metadata": {  "description": "Number of VM instances (100 or less)."  }  },  "adminUsername": {  "type": "String",  "metadata": {  "description": "Admin username on all VMs."  }  },  "adminPassword": {  "type": "SecureString",  "metadata": {  "description": "Admin password on all VMs."  }  }  },  "variables": {  "storageAccountType": "Standard\_LRS",  "namingInfix": "[toLower(substring(concat(parameters('vmssName'), uniqueString(resourceGroup().id)), 0, 9))]",  "longNamingInfix": "[toLower(parameters('vmssName'))]",  "newStorageAccountSuffix": "[concat(variables('namingInfix'), 'sa')]",  "uniqueStringArray": [  "[concat(uniqueString(concat(resourceGroup().id, variables('newStorageAccountSuffix'), '0')))]",  "[concat(uniqueString(concat(resourceGroup().id, variables('newStorageAccountSuffix'), '1')))]",  "[concat(uniqueString(concat(resourceGroup().id, variables('newStorageAccountSuffix'), '2')))]",  "[concat(uniqueString(concat(resourceGroup().id, variables('newStorageAccountSuffix'), '3')))]",  "[concat(uniqueString(concat(resourceGroup().id, variables('newStorageAccountSuffix'), '4')))]"  ],  "vhdContainerName": "[concat(variables('namingInfix'), 'vhd')]",  "osDiskName": "[concat(variables('namingInfix'), 'osdisk')]",  "saCount": "[length(variables('uniqueStringArray'))]",  "addressPrefix": "10.0.0.0/16",  "subnetPrefix": "10.0.0.0/24",  "virtualNetworkName": "[concat(variables('namingInfix'), 'vnet')]",  "publicIPAddressName": "[concat(variables('namingInfix'), 'pip')]",  "subnetName": "[concat(variables('namingInfix'), 'subnet')]",  "loadBalancerName": "[concat(variables('namingInfix'), 'lb')]",  "publicIPAddressID": "[resourceId('Microsoft.Network/publicIPAddresses',variables('publicIPAddressName'))]",  "lbID": "[resourceId('Microsoft.Network/loadBalancers',variables('loadBalancerName'))]",  "natPoolName": "[concat(variables('namingInfix'), 'natpool')]",  "bePoolName": "[concat(variables('namingInfix'), 'bepool')]",  "natStartPort": 50000,  "natEndPort": 50119,  "natBackendPort": 22,  "nicName": "[concat(variables('namingInfix'), 'nic')]",  "ipConfigName": "[concat(variables('namingInfix'), 'ipconfig')]",  "frontEndIPConfigID": "[concat(variables('lbID'),'/frontendIPConfigurations/loadBalancerFrontEnd')]",  "diagnosticsStorageAccountName": "[concat(variables('uniqueStringArray')[0], variables('newStorageAccountSuffix'))]",  "diagnosticsStorageAccountResourceGroup": "[resourceGroup().name]",  "accountid": "[concat('/subscriptions/',subscription().subscriptionId,'/resourceGroups/',variables('diagnosticsStorageAccountResourceGroup'),'/providers/','Microsoft.Storage/storageAccounts/', variables('diagnosticsStorageAccountName'))]",  "wadlogs": "<WadCfg><DiagnosticMonitorConfiguration>",  "wadperfcounters1": "<PerformanceCounters scheduledTransferPeriod=\"PT1M\"><PerformanceCounterConfiguration counterSpecifier=\"\\Memory\\AvailableMemory\" sampleRate=\"PT15S\" unit=\"Bytes\"><annotation displayName=\"Memory available\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Memory\\PercentAvailableMemory\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"Mem. percent available\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Memory\\UsedMemory\" sampleRate=\"PT15S\" unit=\"Bytes\"><annotation displayName=\"Memory used\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Memory\\PercentUsedMemory\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"Memory percentage\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Memory\\PercentUsedByCache\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"Mem. used by cache\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Processor\\PercentIdleTime\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"CPU idle time\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Processor\\PercentUserTime\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"CPU user time\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Processor\\PercentProcessorTime\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"CPU percentage guest OS\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Processor\\PercentIOWaitTime\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"CPU IO wait time\" locale=\"en-us\"/></PerformanceCounterConfiguration>",  "wadperfcounters2": "<PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\BytesPerSecond\" sampleRate=\"PT15S\" unit=\"BytesPerSecond\"><annotation displayName=\"Disk total bytes\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\ReadBytesPerSecond\" sampleRate=\"PT15S\" unit=\"BytesPerSecond\"><annotation displayName=\"Disk read guest OS\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\WriteBytesPerSecond\" sampleRate=\"PT15S\" unit=\"BytesPerSecond\"><annotation displayName=\"Disk write guest OS\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\TransfersPerSecond\" sampleRate=\"PT15S\" unit=\"CountPerSecond\"><annotation displayName=\"Disk transfers\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\ReadsPerSecond\" sampleRate=\"PT15S\" unit=\"CountPerSecond\"><annotation displayName=\"Disk reads\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\WritesPerSecond\" sampleRate=\"PT15S\" unit=\"CountPerSecond\"><annotation displayName=\"Disk writes\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\AverageReadTime\" sampleRate=\"PT15S\" unit=\"Seconds\"><annotation displayName=\"Disk read time\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\AverageWriteTime\" sampleRate=\"PT15S\" unit=\"Seconds\"><annotation displayName=\"Disk write time\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\AverageTransferTime\" sampleRate=\"PT15S\" unit=\"Seconds\"><annotation displayName=\"Disk transfer time\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk\\AverageDiskQueueLength\" sampleRate=\"PT15S\" unit=\"Count\"><annotation displayName=\"Disk queue length\" locale=\"en-us\"/></PerformanceCounterConfiguration></PerformanceCounters>",  "wadcfgxstart": "[concat(variables('wadlogs'),variables('wadperfcounters1'),variables('wadperfcounters2'),'<Metrics resourceId=\"')]",  "wadmetricsresourceid": "[concat('/subscriptions/',subscription().subscriptionId,'/resourceGroups/',resourceGroup().name ,'/providers/','Microsoft.Compute/virtualMachineScaleSets/',variables('namingInfix'))]",  "wadcfgxend": "[concat('\"><MetricAggregation scheduledTransferPeriod=\"PT1H\"/><MetricAggregation scheduledTransferPeriod=\"PT1M\"/></Metrics></DiagnosticMonitorConfiguration></WadCfg>')]",  "computeApiVersion": "2016-03-30",  "networkApiVersion": "2016-03-30",  "storageApiVersion": "2015-06-15",  "insightsApiVersion": "2015-04-01"  },  "resources": [  {  "type": "Microsoft.Network/virtualNetworks",  "name": "[variables('virtualNetworkName')]",  "apiVersion": "[variables('networkApiVersion')]",  "location": "[resourceGroup().location]",  "properties": {  "addressSpace": {  "addressPrefixes": [  "[variables('addressPrefix')]"  ]  },  "subnets": [  {  "name": "[variables('subnetName')]",  "properties": {  "addressPrefix": "[variables('subnetPrefix')]"  }  }  ]  }  },  {  "type": "Microsoft.Storage/storageAccounts",  "name": "[concat(variables('uniqueStringArray')[copyIndex()], variables('newStorageAccountSuffix'))]",  "apiVersion": "[variables('storageApiVersion')]",  "location": "[resourceGroup().location]",  "copy": {  "name": "storageLoop",  "count": "[variables('saCount')]"  },  "properties": {  "accountType": "[variables('storageAccountType')]"  }  },  {  "type": "Microsoft.Network/publicIPAddresses",  "name": "[variables('publicIPAddressName')]",  "apiVersion": "[variables('networkApiVersion')]",  "location": "[resourceGroup().location]",  "properties": {  "publicIPAllocationMethod": "Dynamic",  "dnsSettings": {  "domainNameLabel": "[variables('longNamingInfix')]"  }  }  },  {  "type": "Microsoft.Network/loadBalancers",  "name": "[variables('loadBalancerName')]",  "apiVersion": "[variables('networkApiVersion')]",  "location": "[resourceGroup().location]",  "properties": {  "frontendIPConfigurations": [  {  "name": "LoadBalancerFrontEnd",  "properties": {  "publicIPAddress": {  "id": "[variables('publicIPAddressID')]"  }  }  }  ],  "backendAddressPools": [  {  "name": "[variables('bePoolName')]"  }  ],  "inboundNatPools": [  {  "name": "[variables('natPoolName')]",  "properties": {  "frontendIPConfiguration": {  "id": "[variables('frontEndIPConfigID')]"  },  "protocol": "tcp",  "frontendPortRangeStart": "[variables('natStartPort')]",  "frontendPortRangeEnd": "[variables('natEndPort')]",  "backendPort": "[variables('natBackendPort')]"  }  }  ]  },  "dependsOn": [  "[concat('Microsoft.Network/publicIPAddresses/', variables('publicIPAddressName'))]"  ]  },  {  "type": "Microsoft.Compute/virtualMachineScaleSets",  "sku": {  "name": "[parameters('vmSku')]",  "tier": "Standard",  "capacity": "[parameters('instanceCount')]"  },  "name": "[variables('namingInfix')]",  "apiVersion": "[variables('computeApiVersion')]",  "location": "[resourceGroup().location]",  "properties": {  "overprovision": "true",  "upgradePolicy": {  "mode": "Manual"  },  "virtualMachineProfile": {  "storageProfile": {  "osDisk": {  "name": "vmssosdisk",  "caching": "ReadOnly",  "createOption": "FromImage",  "osType": "Linux",  "image": {  "uri": "[parameters('vmImage')]"  }  }  },  "osProfile": {  "computerNamePrefix": "[variables('namingInfix')]",  "adminUsername": "[parameters('adminUsername')]",  "adminPassword": "[parameters('adminPassword')]"  },  "networkProfile": {  "networkInterfaceConfigurations": [  {  "name": "[variables('nicName')]",  "properties": {  "primary": "true",  "ipConfigurations": [  {  "name": "[variables('ipConfigName')]",  "properties": {  "subnet": {  "id": "[concat('/subscriptions/', subscription().subscriptionId,'/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/virtualNetworks/', variables('virtualNetworkName'), '/subnets/', variables('subnetName'))]"  },  "loadBalancerBackendAddressPools": [  {  "id": "[concat('/subscriptions/', subscription().subscriptionId,'/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/', variables('loadBalancerName'), '/backendAddressPools/', variables('bePoolName'))]"  }  ],  "loadBalancerInboundNatPools": [  {  "id": "[concat('/subscriptions/', subscription().subscriptionId,'/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/', variables('loadBalancerName'), '/inboundNatPools/', variables('natPoolName'))]"  }  ]  }  }  ]  }  }  ]  },  "extensionProfile": {  "extensions": [  {  "name": "LinuxDiagnostic",  "properties": {  "publisher": "Microsoft.OSTCExtensions",  "type": "LinuxDiagnostic",  "typeHandlerVersion": "2.3",  "autoUpgradeMinorVersion": true,  "settings": {  "xmlCfg": "[base64(concat(variables('wadcfgxstart'),variables('wadmetricsresourceid'),variables('wadcfgxend')))]",  "storageAccount": "[variables('diagnosticsStorageAccountName')]"  },  "protectedSettings": {  "storageAccountName": "[variables('diagnosticsStorageAccountName')]",  "storageAccountKey": "[listkeys(variables('accountid'), variables('storageApiVersion')).key1]",  "storageAccountEndPoint": "https://core.windows.net"  }  }  }  ]  }  }  },  "dependsOn": [  "[concat('Microsoft.Storage/storageAccounts/', variables('uniqueStringArray')[0], variables('newStorageAccountSuffix'))]",  "[concat('Microsoft.Storage/storageAccounts/', variables('uniqueStringArray')[1], variables('newStorageAccountSuffix'))]",  "[concat('Microsoft.Storage/storageAccounts/', variables('uniqueStringArray')[2], variables('newStorageAccountSuffix'))]",  "[concat('Microsoft.Storage/storageAccounts/', variables('uniqueStringArray')[3], variables('newStorageAccountSuffix'))]",  "[concat('Microsoft.Storage/storageAccounts/', variables('uniqueStringArray')[4], variables('newStorageAccountSuffix'))]",  "[concat('Microsoft.Network/loadBalancers/', variables('loadBalancerName'))]",  "[concat('Microsoft.Network/virtualNetworks/', variables('virtualNetworkName'))]"  ]  },  {  "type": "Microsoft.Insights/autoscaleSettings",  "name": "autoscalewad",  "apiVersion": "[variables('insightsApiVersion')]",  "location": "[resourceGroup().location]",  "properties": {  "name": "autoscalewad",  "targetResourceUri": "[concat('/subscriptions/',subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Compute/virtualMachineScaleSets/', variables('namingInfix'))]",  "enabled": true,  "profiles": [  {  "name": "Profile1",  "capacity": {  "minimum": "1",  "maximum": "10",  "default": "1"  },  "rules": [  {  "metricTrigger": {  "metricName": "\\Processor\\PercentProcessorTime",  "metricNamespace": "",  "metricResourceUri": "[concat('/subscriptions/',subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Compute/virtualMachineScaleSets/', variables('namingInfix'))]",  "timeGrain": "PT1M",  "statistic": "Average",  "timeWindow": "PT5M",  "timeAggregation": "Average",  "operator": "GreaterThan",  "threshold": 60  },  "scaleAction": {  "direction": "Increase",  "type": "ChangeCount",  "value": "1",  "cooldown": "PT1M"  }  },  {  "metricTrigger": {  "metricName": "\\Processor\\PercentProcessorTime",  "metricNamespace": "",  "metricResourceUri": "[concat('/subscriptions/',subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Compute/virtualMachineScaleSets/', variables('namingInfix'))]",  "timeGrain": "PT1M",  "statistic": "Average",  "timeWindow": "PT5M",  "timeAggregation": "Average",  "operator": "LessThan",  "threshold": 30  },  "scaleAction": {  "direction": "Decrease",  "type": "ChangeCount",  "value": "1",  "cooldown": "PT5M"  }  },  {  "metricTrigger": {  "metricName": "\\Memory\\PercentUsedMemory",  "metricNamespace": "",  "metricResourceUri": "[concat('/subscriptions/',subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Compute/virtualMachineScaleSets/', variables('namingInfix'))]",  "timeGrain": "PT1M",  "statistic": "Average",  "timeWindow": "PT5M",  "timeAggregation": "Average",  "operator": "GreaterThan",  "threshold": 75  },  "scaleAction": {  "direction": "Increase",  "type": "ChangeCount",  "value": "1",  "cooldown": "PT1M"  }  },  {  "metricTrigger": {  "metricName": "\\Memory\\PercentUsedMemory",  "metricNamespace": "",  "metricResourceUri": "[concat('/subscriptions/',subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Compute/virtualMachineScaleSets/', variables('namingInfix'))]",  "timeGrain": "PT1M",  "statistic": "Average",  "timeWindow": "PT5M",  "timeAggregation": "Average",  "operator": "LessThan",  "threshold": 25  },  "scaleAction": {  "direction": "Decrease",  "type": "ChangeCount",  "value": "1",  "cooldown": "PT5M"  }  }  ]  }  ]  },  "dependsOn": [  "[concat('Microsoft.Compute/virtualMachineScaleSets/', variables('namingInfix'))]"  ]  }  ]  } |

In Image text pane you can paste your vhd uri. Vhd uri can be found under storage account of base vm

## VMSS Application Deployment



## Storage account type scaling

|  |
| --- |
| Limits A scale set built on a custom image (one built by you) must create all OS disk VHDs within one storage account. As a result, the maximum recommended number of VMs in a scale set built on a custom image is 20. If you turn off overprovisioning, you can go up to 40.  A scale set built on a platform image is limited to 100 VMs (we recommend 5 storage accounts for this scale).  For more VMs than these limits allow, you will need to deploy multiple scale sets. [For an example of how to do this, please see this template.](https://github.com/Azure/azure-quickstart-templates/tree/master/301-custom-images-at-scale) |

<https://github.com/uglide/azure-content/blob/master/articles/virtual-machine-scale-sets/virtual-machine-scale-sets-design-overview.md>

## POC VMSS LoadTesting

* Error: Tue Jan 23 11:23:19 IST 2018:ERROR:javax.net.ssl.SSLHandshakeException: Remote host closed connection during handshake

Esb Errors

|  |
| --- |
| SynapseCallbackReceiver Synapse received a response for the request with message Id : urn:uuid:a1c625a2-6b51-4bde-b4c9-fb91b50c8085 But a callback is not registered (anymore) to process this response |

*This is due to callback timeout when response lands to ESB.*

<http://abeykoon.blogspot.com/2017/06/wso2-esb-internals-outgoing-message-path.html>

# Create Swap file in linux (cent os 7 )

<https://www.vembu.com/blog/increase-swap-memory-centos-7/>

Edit the file with sudo privileges in your text editor:

sudo nano /etc/fstab

At the bottom of the file, you need to add a line that will tell the operating system to automatically use the swap file that you created:

/swapfile swap swap sw 0 0

When you are finished adding the line, you can save and close the file. The server will check this file on each bootup, so the swap file will be ready for use from now on.

Referring link : <https://www.digitalocean.com/community/tutorials/how-to-add-swap-on-centos-7>

# Verify Log file

http://ajanthane.blogspot.com/2016/11/

Result Verification

By enabling the DEBUG logs at log4j.properties for

log4j.category.org.apache.synapse.transport=DEBUG

log4j.logger.org.apache.synapse.transport.http.wire=DEBUG

We can verify it by the below wso2carbon.log

# Change Host Name

https://www.cyberciti.biz/faq/rhel-redhat-centos-7-change-hostname-command/

# Api Manager OS Level Tune up

To optimize network and OS performance, configure the following settings in the /etc/sysctl.conf file of Linux. These settings specify a larger port range, a more effective TCP connection timeout value, and a number of other important parameters at the OS-level.

Oasys Live VM sysctl.conf configuration

|  |
| --- |
| net.ipv4.tcp\_fin\_timeout = 30  fs.file-max = 2097152  net.ipv4.tcp\_tw\_recycle = 1  net.ipv4.tcp\_tw\_reuse = 1  net.core.rmem\_default = 524288  net.core.wmem\_default = 524288  net.core.rmem\_max = 67108864  net.core.wmem\_max = 67108864  net.ipv4.tcp\_rmem = 4096 87380 16777216  net.ipv4.tcp\_wmem = 4096 65536 16777216  net.ipv4.ip\_local\_port\_range = 1024 65535  net.ipv6.conf.all.disable\_ipv6 = 1  net.ipv6.conf.disable.disable\_ipv6 = 1  net.ipv6.conf.lo.disable\_ipv6 = 1  net.ipv4.tcp\_timestamps = 0 |

# Ssh to another vm

Scp Copy Directory

scp -r wso2esb-5.0.0 [VmAdmin@52.230.80.202:/home/VmAdmin](mailto:VmAdmin@52.230.80.202:/home/VmAdmin)

scp File

scp wso2esb-5.0.0 [VmAdmin@52.230.80.202:/home/VmAdmin](mailto:VmAdmin@52.230.80.202:/home/VmAdmin)

# Pass

|  |
| --- |
| LOLCAdfsSp  Service provider for LOLC adfs  LOLCADFS  <https://api.oasys.lk:443/token>  Kenneth.teo@microsoft.com  <https://aka.ms/hostedlab>  key: lolcdevops  OasysLive  Service port : 8243  UserName: VmAdmin  Password:0asys@123$  <https://api.oasys.lk:8543/carbon/admin/login.jsp>  <https://api.oasys.lk:9543/carbon/admin/login.jsp>  Oasys Live MGConsole:  <AdminUser>  <UserName>OasysUser</UserName>  <Password>0asysUsr@123$</Password>  </AdminUser>  Oasys DEV APIM  Service port : 8243  VmUserName : devvmadmin  PW:DevVmAdmin@123  <https://apidev.oasys.lk:9443/carbon/admin/login.jsp>  <https://apidev.oasys.lk:9543/carbon/admin/login.jsp>  <AdminUser>  <UserName>OasysUser</UserName>  <Password>oasysUsr123$</Password>  </AdminUser>  Oasys POC  Service port : 8443  POC USERNAME (CLASSIC): VmAdmin (ARM): vmadmin (pw same)  Password POC(classic) : vmAdmin@12345  <https://apipoc.oasys.lk:9643/carbon/admin/login.jsp>  <https://apipoc.oasys.lk:9543/carbon/admin/login.jsp>  <AdminUser>  <UserName>PocUsr</UserName>  <Password>pocusr</Password>  </AdminUser>  New PROD VM PW 0Oasys@123$  OasysADApp  OasysAdfsApp  OasysDBApp  <https://support.microsoft.com/en-us/help/29289/receive-remote-assistance-support-from-microsoft>  Driver name : com.microsoft.sqlserver.jdbc.SQLServerDriver  jdbc:sqlserver://oasys-dev-services.azurewebsites.net:1433;database=OASys\_DEV\_GlobalServiceDesk;socketTimeout=0  VIEW\_USERSTORE  azure knowledge  svts  api.ai |
|  |

# Live HAPROXY.cfg

|  |
| --- |
| [VmAdmin@LIVE-Middleware haproxy]$ cat haproxy.cfg  #---------------------------------------------------------------------  # Example configuration for a possible web application. See the  # full configuration options online.  #  # http://haproxy.1wt.eu/download/1.4/doc/configuration.txt  #  #---------------------------------------------------------------------  #---------------------------------------------------------------------  # Global settings  #---------------------------------------------------------------------  global  # to have these messages end up in /var/log/haproxy.log you will  # need to:  #  # 1) configure syslog to accept network log events. This is done  # by adding the '-r' option to the SYSLOGD\_OPTIONS in  # /etc/sysconfig/syslog  #  # 2) configure local2 events to go to the /var/log/haproxy.log  # file. A line like the following can be added to  # /etc/sysconfig/syslog  #  # local2.\* /var/log/haproxy.log  #  log 127.0.0.1 local2  tune.ssl.default-dh-param 4096  stats socket /var/lib/haproxy/stats  stats socket /var/run/haproxy.sock mode 600 level admin  stats timeout 2m  #---------------------------------------------------------------------  # common defaults that all the 'listen' and 'backend' sections will  # use if not designated in their block  #---------------------------------------------------------------------  defaults  mode http  log global  option httplog  option dontlognull  option http-server-close  option forwardfor except 127.0.0.0/8  option redispatch  retries 3  timeout http-request 5m  timeout queue 3m  timeout connect 5000  timeout client 24h  timeout server 24h  timeout http-keep-alive 5m  timeout check 10s  maxconn 6000  #---------------------------------------------------------------------  # main frontend which proxys to the backends  #---------------------------------------------------------------------  frontend https-oasys  bind \*:443 ssl crt /etc/haproxy/certs/oasysliveazure.pem  mode http  option httplog  option forwardfor  default\_backend oasysapim\_backend  backend oasysapim\_backend  balance source  option forwardfor  option httpclose  server 59apihttp localhost:8280 check  frontend https-adfs  bind \*:8543 ssl crt /etc/haproxy/certs/oasysliveazure.pem  mode tcp  default\_backend adfs\_backend  backend adfs\_backend  mode tcp  balance source  server 37idsnode1 localhost:9443 ssl verify none check  frontend http\_statts  bind \*:8081  stats enable  stats scope https-oasys  stats scope oasysapim\_backend  stats scope https-adfs  stats scope adfs\_backend  stats uri /haproxy?stats  stats hide-version  stats auth haproxy:stats@lolc |

# To Check Live VM haproxy.cfg change like this

|  |
| --- |
| #---------------------------------------------------------------------  # Example configuration for a possible web application. See the  # full configuration options online.  #  # http://haproxy.1wt.eu/download/1.4/doc/configuration.txt  #  #---------------------------------------------------------------------  #---------------------------------------------------------------------  # Global settings  #---------------------------------------------------------------------  global  # to have these messages end up in /var/log/haproxy.log you will  # need to:  #  # 1) configure syslog to accept network log events. This is done  # by adding the '-r' option to the SYSLOGD\_OPTIONS in  # /etc/sysconfig/syslog  #  # 2) configure local2 events to go to the /var/log/haproxy.log  # file. A line like the following can be added to  # /etc/sysconfig/syslog  #  # local2.\* /var/log/haproxy.log  #  log 127.0.0.1 local2  tune.ssl.default-dh-param 4096  stats socket /var/lib/haproxy/stats  stats socket /var/run/haproxy.sock mode 600 level admin  stats timeout 2m  #---------------------------------------------------------------------  # common defaults that all the 'listen' and 'backend' sections will  # use if not designated in their block  #---------------------------------------------------------------------  defaults  mode http  log global  option httplog  option dontlognull  option http-server-close  option forwardfor except 127.0.0.0/8  option redispatch  retries 3  timeout http-request 5m  timeout queue 3m  timeout connect 5000  timeout client 24h  timeout server 24h  timeout http-keep-alive 5m  timeout check 10s  maxconn 6000  #---------------------------------------------------------------------  # main frontend which proxys to the backends  #---------------------------------------------------------------------  frontend https-oasys  bind \*:443 ssl crt /etc/haproxy/certs/oasysliveazure.pem  mode http  option httplog  option forwardfor  default\_backend oasysapim\_backend  backend oasysapim\_backend  balance source  option forwardfor  option httpclose  server 59apihttp prodmiddleware.southeastasia.cloudapp.azure.com:8280 check  frontend https-adfs  bind \*:8543 ssl crt /etc/haproxy/certs/oasysliveazure.pem  mode tcp  default\_backend adfs\_backend  backend adfs\_backend  mode tcp  balance source  server 37idsnode1 prodmiddleware.southeastasia.cloudapp.azure.com:9443 ssl verify none check  frontend http\_statts  bind \*:8081  stats enable  stats scope https-oasys  stats scope oasysapim\_backend  stats scope https-adfs  stats scope adfs\_backend  stats uri /haproxy?stats  stats hide-version  stats auth haproxy:stats@lolc |

# How To Handle Schema Changes In Azure SQL Data Sync

<https://www.francoisdelport.com/2017/07/how-to-handle-schema-changes-in-azure-sql-data-sync/>

# Sync data across multiple cloud and on-premises databases with SQL Data Sync (Preview)

<https://docs.microsoft.com/en-us/azure/sql-database/sql-database-sync-data#when-to-use-data-sync>

# **Oasys Live shift from classic vm to ARM**

|  |  |  |
| --- | --- | --- |
| # | Command | Desc |
| 01 | > telnet prodmiddleware.southeastasia.cloudapp.azure.com 8280 | Telnetting service port of api gateway from classic vm |
| 02 | > telnet prodmiddleware.southeastasia.cloudapp.azure.com 9443 | Telnetting API Management console |
| 03 | > cp haproxy.cfg haproxy.cfg.working.01022018 | Get Backup working haproxy.cfg |
| 04 | > sudo cp haproxy.cfg haproxy.cfg.working.01022018 | Make copy again as haproxy.cfg |
| 05 | > sudo vi haproxy.cfg | Edit haproxy.cfg as above highlighted |
| 06 | > history|grep start | To view history key as start |
| 07 | > sudo systemctl restart haproxy.service | Restart haproxy |
| 08 | > sudo cp haproxy.cfg haproxy.cfg.workingARMSERVERTEST.01022018 | Make copy of haproxy.cfg |
| 09 | > sudo cp haproxy.cfg.working.01022018 haproxy.cfg | Change back to original haproxy.cfg |
| 10 | > sudo systemctl restart haproxy.service | Restart haproxy |

Oasys Live (ESB)

Response count : 36502 2018/02/5 8:50 AM

Response count : 58343 2018/02/5 11:50 AM

Response Count : 70652 2018/02/5 1:15 PM

Response Count : 81280 2018/02/5 2:50 PM

Response Count : 98318 2018/02/5 4:50 PM

Response Count : 136180 2018/02/6 9:00 PM

# Steps to Create a New Sudo User

1. Log in to your server as the root user.
   * ssh root@server\_ip\_address
2. Use the adduser command to add a new user to your system.

Be sure to replace username with the user that you want to create.

* + adduser username
  + Use the passwd command to update the new user's password.
    - passwd username
  + Set and confirm the new user's password at the prompt. A strong password is highly recommended!

Set password prompts:

Changing password for user username.

New password:

Retype new password:

passwd: all authentication tokens updated successfully.

1. Use the usermod command to add the user to the wheel group.
   * usermod -aG wheel username

By default, on CentOS, members of the wheel group have sudo privileges.

1. Test sudo access on new user account
   * Use the su command to switch to the new user account.
     + su - username
   * As the new user, verify that you can use sudo by prepending "sudo" to the command that you want to run with superuser privileges.

# WSO2 IS 5.1 PATCH

# IDP ALLIAS

<https://github.com/wso2-attic/carbon-identity/commit/0244b00a006f2a261a267d1a93191127da240f0d>

is 5.1 refresh token issue

<https://wso2.org/jira/browse/IDENTITY-4633>

# Configure apim 1.10 & IDS 5.1 as Key Manager

### Step 1 - Download WSO2 APIM and WSO2 IS

* Download WSO2 Identity Server 5.1.0 from [Identity Server product page](http://wso2.com/products/identity-server/) and [install it](https://docs.wso2.com/display/IS510/Installing+the+Product). <IS\_HOME> will refer to the root folder of the unzipped WSO2 IS pack.
* Download WSO2 API Manager (WSO2 APIM) 1.10.0 from [here](http://wso2.com/products/api-manager/) and [install it](https://docs.wso2.com/display/AM1100/Installing+the+Product). <APIM\_HOME> will refer to the root folder of the unzipped WSO2 API-M pack.

### Step 2 - Optionally, configure port offset for WSO2 APIM or WSO2 IS

This is only required if you running WSO2 API Manager on the same Virtual Machine (VM) as the WSO2 Identity Server.

**What is port offset?**

The port offset feature allows you to run multiple WSO2 products, multiple instances of a WSO2 product, or multiple WSO2 product clusters on the same server or virtual machine (VM). The port offset defines the number by which all ports defined in the runtime such as the HTTP/S ports will be offset. For example, if the HTTP port is defined as 9763 and the portOffset is 1, the effective HTTP port will be 9764. Therefore, for each additional WSO2 product, instance, or cluster you add to a server, set the port offset to a unique value (the default is 0).

Open the <PRODUCT\_HOME>/repository/conf/carbon.xml file and change the offset to 1. This increments the product's default port by one. <PRODUCT\_HOME> refers to the product to which you are configuring a port offset and it can be either <IS\_HOME> or <APIM\_HOME>.

**carbon.xml**

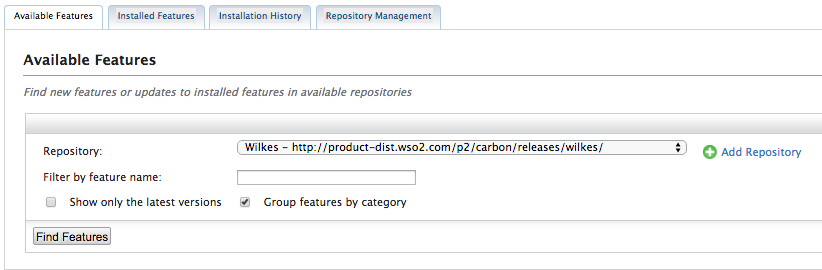
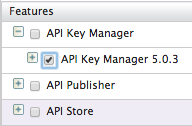
|  |
| --- |
| <Offset>1</Offset> |

### Step 3 - Configure the Identity Server

1. Log into the Identity Server and access the [Management Console](https://docs.wso2.com/display/IS510/Getting+Started+with+the+Management+Console).
2. After starting the Identity Server, [install the Key Manager feature](https://docs.wso2.com/display/IS510/Installing+Features).

**Warning**: Installing this feature will result in changes to some of the configuration files in Identity Server. The <IS\_HOME>/repository/conf/identity.xml file and the <IS\_HOME>/repository/conf/datasources/master-datasouces.xml file will lose their current configurations.

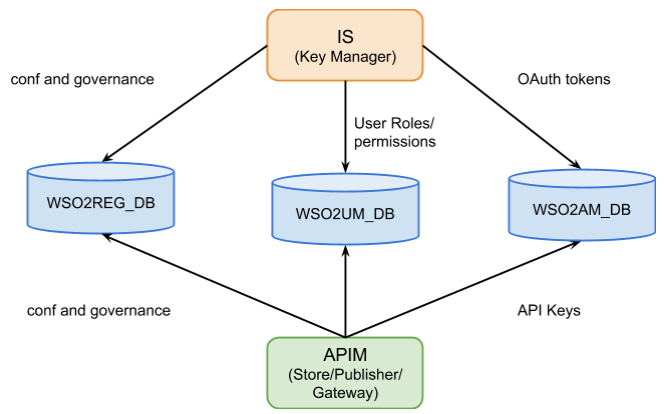
To install the feature:

* 1. Navigate to the **Features** section in the **Configure** menu of the management console.
  2. Add the following feature repository in the **Feature Management** section in the Identity Server. See [here](https://docs.wso2.com/display/IS510/Repository+Management) for information on how to do this.  
     P2 Repo: <http://product-dist.wso2.com/p2/carbon/releases/wilkes/>
  3. After adding the repository, navigate to the **Available Features** tab and find the feature in that repository by clicking the **Find Features** button. The list of available features appear.  
     
     1. Expand the **API** **Key Manager** feature from the **Features** category and select **API Key Manager 5.0.3**.  
        
     2. Click on the **Install** button and go through the wizard to complete the installation. See [here](https://docs.wso2.com/display/IS510/Installing+Features) if you require more information on how to do this.

1. Download WSO2 API Manager from [here](http://wso2.com/products/api-manager/)  and [install it](https://docs.wso2.com/display/AM1100/Installing+the+Product) .
2. Copy the  identity.xml  file from the <APIM\_HOME>/repository/conf/identity  directory and paste it into the <IS\_HOME>/repository/conf/identity directory. You must replace the file that is already available there. This is because you just installed the key manager feature, and you need to use a new identity.xml file that contains some additional changes.
3. Copy the **api-manager.xml** file from the <APIM\_HOME>/repository/conf directory and paste it into the <IS\_HOME>/repository/conf directory.
4. Make the following changes in the **api-manager.xml** file you just copied.
   1. Change the GatewayType property to the following. This is done because the default value here is Synapse. Synapse runtime is used for various ESB related functionality that is not available in the Identity Server, so this must be changed to None.  
      <GatewayType>None</GatewayType>
   2. Change the <RevokeAPIURL> so that it points to the API Manager server. Note that if API Manager is running in distributed mode (has a separate node for the Gateway), you need to point this URL to the Gateway worker node. This is done so that when the token is revoked, the Gateway cache is updated as well. The port value you enter here must be the NIO port. See [Default Ports of WSO2 Products](https://docs.wso2.com/display/Carbon443/Default+Ports+of+WSO2+Products) for more information.  
      <RevokeAPIURL> https://${GATEWAY\_SERVER\_HOST}:{port}/revoke </RevokeAPIURL>
   3. Change the <ServerURL> occurring under the <APIGateway> section so that it points to the API Manager server. If you are using distributed mode, this needs to point to the Gateway manager node. This is done so that when the token is regenerated, the Gateway cache is updated as well. The port value you enter here must be the management transport port. See [Default Ports of WSO2 Products](https://docs.wso2.com/display/Carbon443/Default+Ports+of+WSO2+Products) for more information.  
      <ServerURL>https://${GATEWAY\_SERVER\_HOST}:{port}/services/</ServerURL>
   4. Change EnableThriftServer to false. The Identity Server does not come with a thrift server and this causes issues at runtime if not disabled.  
      <EnableThriftServer>false</EnableThriftServer>
5. Open the <IS\_HOME>/repository/conf/datasources/master-datasources.xml file and add the following datasources.

Ensure that you keep the 'WSO2\_CARBON\_DB' datasource the way it is and simply add the following datasources in the master-datasources.xml file. Also note that the WSO2AM\_DB is already added in the master-datasources.xml file so you do not need to add it again. However, you must edit this datasource to point to your new database as this still points to the default H2 database. The following code block includes a sample of the WSO2AM\_DB datasource as a sample configuration when pointing to the new database.

|  |
| --- |
| <datasource>      <name>WSO2AM\_DB</name>      <description>The datasource used for API Manager database</description>      <jndiConfig>          <name>jdbc/WSO2AM\_DB</name>      </jndiConfig>      <definition type="RDBMS">          <configuration>  <url>jdbc:mysql://localhost:3306/apimgt?autoReconnect=true&amp;relaxAutoCommit=true&amp;</url>              <username>apiuser</username>              <password>apimanager</password>              <driverClassName>com.mysql.jdbc.Driver</driverClassName>              <maxActive>50</maxActive>              <maxWait>60000</maxWait>              <testOnBorrow>true</testOnBorrow>              <validationQuery>SELECT 1</validationQuery>              <validationInterval>30000</validationInterval>              <defaultAutoCommit>false</defaultAutoCommit>          </configuration>      </definition>  </datasource>    <datasource>      <name>WSO2REG\_DB</name>      <description>The datasource used for registry</description>      <jndiConfig>          <name>jdbc/WSO2REG\_DB</name>      </jndiConfig>      <definition type="RDBMS">          <configuration>  <url>jdbc:mysql://localhost:3306/registry?autoReconnect=true&amp;relaxAutoCommit=true&amp;</url>              <username>apiuser</username>              <password>apimanager</password>              <driverClassName>com.mysql.jdbc.Driver</driverClassName>              <maxActive>50</maxActive>              <maxWait>60000</maxWait>              <testOnBorrow>true</testOnBorrow>              <validationQuery>SELECT 1</validationQuery>              <validationInterval>30000</validationInterval>          </configuration>      </definition>  </datasource>    <datasource>      <name>WSO2UM\_DB</name>      <description>The datasource used for user management</description>      <jndiConfig>          <name>jdbc/WSO2UM\_DB</name>      </jndiConfig>      <definition type="RDBMS">          <configuration>     <url>jdbc:mysql://localhost:3306/userstore?autoReconnect=true&amp;relaxAutoCommit=true&amp;              </url>              <username>apiuser</username>              <password>apimanager</password>              <driverClassName>com.mysql.jdbc.Driver</driverClassName>              <maxActive>50</maxActive>              <maxWait>60000</maxWait>              <testOnBorrow>true</testOnBorrow>              <validationQuery>SELECT 1</validationQuery>              <validationInterval>30000</validationInterval>          </configuration>      </definition>  </datasource> |

The following diagram illustrates how databases are shared between IS and APIM as per the above configuration.  


* 1. **WSO2REG\_DB** - This is used to keep the registry information. The registry database is shared between WSO2 IS as the Key Manager and WSO2 APIM to share artifacts such as, meta data configurations, policies, and API details.
  2. **WSO2UM\_DB** - This is used to store the permissions (i.e. permission store) and the internal roles of the users.
  3. **WSO2AM\_DB** - This will be used to keep the identity data and API-related data. This includes OAuth tokens and keys. When serving key-validation requests, the key manager validates whether there are subscriptions made by the particular key. For this WSO2AM\_DB should be accessed.

1. Make the following change to the  <IS\_HOME>/repository/conf/registry.xml file. Create the registry mounts by inserting the following sections into the **registry.xml** file.

When doing this change, do not replace the existing <dbConfig> for "wso2registry". Simply add the following configuration to the existing configurations.

|  |
| --- |
| <dbConfig name="govregistry">          <dataSource>jdbc/WSO2REG\_DB</dataSource>  </dbConfig>    <remoteInstance url="[https://localhost"](https://localhost/)>          <id>gov</id>          <dbConfig>govregistry</dbConfig>          <cacheId>apiuser@jdbc:mysql://localhost:3306/registry</cacheId>          <readOnly>false</readOnly>          <enableCache>true</enableCache>          <registryRoot>/</registryRoot>  </remoteInstance>    <mount path="/\_system/governance" overwrite="true">          <instanceId>gov</instanceId>          <targetPath>/\_system/governance</targetPath>  </mount>    <mount path="/\_system/config" overwrite="true">         <instanceId>gov</instanceId>         <targetPath>/\_system/config</targetPath>  </mount> |

1. Change the datasource in the **user-mgt.xml** file found in the <IS\_HOME>/repository/conf/ directory to WSO2UM\_DB.

**user-mgt.xml configurations**

|  |
| --- |
| <Realm>          <Configuration>              ...              <Property name="dataSource">jdbc/WSO2UM\_DB</Property>          </Configuration>          ...  </Realm> |

1. Add the user store configuration correctly in the <IS\_HOME>/repository/conf/user-mgt.xml file so that both the Identity Server and API Manager point to the same user store. For more information on configuring user stores, see [here](https://docs.wso2.com/display/IS510/Configuring+the+Realm).

You must change the <UserStoreManager> element here since the internal LDAP user store is used by default. The <UserStoreManager class="org.wso2.carbon.user.core.ldap.ReadWriteLDAPUserStoreManager"> code block needs to be removed or modified and the right code block must be used.

1. Create the following databases in the MySQL database server.
   1. userstore
   2. registry
   3. apimgt

For creating the userstore and registry database, use the <IS\_HOME>/dbscripts/mysql.sql script.

Note that MySQL is used as an example here and you can use a different database if required.

When creating the apimgt db, run the following script; <APIM\_HOME>/dbscripts/apimgt/mysql.sql. The script found in the <APIM\_HOME>/dbscripts/apimgt/ directory has all the tables required to manage OAuth access tokens and also includes other identity-related features.

You can change the **CREATE TABLE** statements to **CREATE TABLE IF NOT EXISTS** when running the identity scripts after you have run the apimgt script.

1. Create a user ‘apiuser’ with password ‘apimanager’. Grant all permissions for this user in the above three databases. For example:

|  |
| --- |
| grant all on apimgt.\* TO apiuser@localhost identified by "apimanager";  grant all on userstore.\* TO apiuser@localhost identified by "apimanager";  grant all on registry.\* TO apiuser@localhost identified by "apimanager"; |

1. JWT configuration must be done in the <IS\_HOME>/repository/conf/api-manager.xml file in the Identity Server. See [here](https://docs.wso2.com/display/AM170/Passing+Enduser+Attributes+to+the+Backend+Using+JWT) for more information on JWT Token generation. Enable the ClaimsRetrieverImplClass, ConsumerDialectURI and SignatureAlgorithm. Set SignatureAlgorithm to NONE.
2. [Start the server](https://docs.wso2.com/display/IS510/Running+the+Product) for the changes to take effect.

### Step 4 - Configure the API Manager

1. Open the <APIM\_HOME>/repository/conf/datasources/master-datasources.xml file and add the following datasources.

|  |
| --- |
| <datasource>      <name>WSO2AM\_DB</name>      <description>The datasource used for API Manager database</description>      <jndiConfig>          <name>jdbc/WSO2AM\_DB</name>      </jndiConfig>      <definition type="RDBMS">          <configuration>  <url>jdbc:mysql://localhost:3306/apimgt?autoReconnect=true&amp;relaxAutoCommit=true&amp;</url>              <username>apiuser</username>              <password>apimanager</password>              <driverClassName>com.mysql.jdbc.Driver</driverClassName>              <maxActive>50</maxActive>              <maxWait>60000</maxWait>              <testOnBorrow>true</testOnBorrow>              <validationQuery>SELECT 1</validationQuery>              <validationInterval>30000</validationInterval>              <defaultAutoCommit>false</defaultAutoCommit>          </configuration>      </definition>  </datasource>    <datasource>      <name>WSO2REG\_DB</name>      <description>The datasource used for registry and user manager</description>      <jndiConfig>          <name>jdbc/WSO2REG\_DB</name>      </jndiConfig>      <definition type="RDBMS">          <configuration>  <url>jdbc:mysql://localhost:3306/registry?autoReconnect=true&amp;relaxAutoCommit=true&amp;</url>              <username>apiuser</username>              <password>apimanager</password>              <driverClassName>com.mysql.jdbc.Driver</driverClassName>              <maxActive>50</maxActive>              <maxWait>60000</maxWait>              <testOnBorrow>true</testOnBorrow>              <validationQuery>SELECT 1</validationQuery>              <validationInterval>30000</validationInterval>          </configuration>      </definition>  </datasource>    <datasource>      <name>WSO2UM\_DB</name>      <description>The datasource used for registry and user manager</description>      <jndiConfig>          <name>jdbc/WSO2UM\_DB</name>      </jndiConfig>      <definition type="RDBMS">          <configuration>     <url>jdbc:mysql://localhost:3306/userstore?autoReconnect=true&amp;relaxAutoCommit=true&amp;              </url>              <username>apiuser</username>              <password>apimanager</password>              <driverClassName>com.mysql.jdbc.Driver</driverClassName>              <maxActive>50</maxActive>              <maxWait>60000</maxWait>              <testOnBorrow>true</testOnBorrow>              <validationQuery>SELECT 1</validationQuery>              <validationInterval>30000</validationInterval>          </configuration>      </definition>  </datasource> |

1. Open the **user-mgt.xml** file found in the <APIM\_HOME>/repository/conf directory and change the permission datasource.
   1. Add the datasource configuration as below

**user-mgt.xml configurations**

|  |
| --- |
| <Realm>          <Configuration>              ...              <Property name="dataSource">jdbc/WSO2UM\_DB</Property>          </Configuration>          ...  </Realm> |

* 1. Configure the <UserStoreManager> section of the <AM\_HOME>/repository/conf/user-mgt.xml file of the API Manager.

Make sure you add the user store configuration correctly. This is the same configuration that you did in the Identity Server. For more information on how to do this, see [here](https://docs.wso2.com/display/IS510/Configuring+User+Stores).

1. Create the registry mounts. Open the <APIM\_HOME>/repository/conf/registry.xml file and insert the following sections.

|  |
| --- |
| <dbConfig name="govregistry">         <dataSource>jdbc/WSO2REG\_DB</dataSource>  </dbConfig>    <remoteInstance url="[https://localhost"](https://localhost/)>         <id>gov</id>         <dbConfig>govregistry</dbConfig>         <cacheId>apiuser@jdbc:mysql://localhost:3306/registry</cacheId>         <readOnly>false</readOnly>         <enableCache>true</enableCache>         <registryRoot>/</registryRoot>  </remoteInstance>    <mount path="/\_system/governance" overwrite="true">         <instanceId>gov</instanceId>         <targetPath>/\_system/governance</targetPath>  </mount>    <mount path="/\_system/config" overwrite="true">         <instanceId>gov</instanceId>         <targetPath>/\_system/config</targetPath>  </mount> |

1. Open the **api-manager.xml** file found in the <APIM\_HOME>/repository/conf directory and change the following.
   1. Change the ServerURL of the AuthManager to point to IS.  
      <ServerURL> https://${IS\_SERVER\_HOST}:{port}/services/ </ServerURL>
   2. Change the ServerURL of the APIKeyValidator to point to IS.  
      <ServerURL> https://${IS\_SERVER\_HOST}:{port}/services/ </ServerURL>
   3. Change the KeyValidatorClientType from ThriftClient to WSClient.
   4. Change EnableThriftServer to false.
2. Navigate to the <AM\_HOME>/repository/deployment/server/synapse-configs/default/api directory and change the URLs specified in the following files of the API Manager so that they point to the IS node. Note that when you run the API Manager in distributed mode, these configurations should be done in the API Gateway node.
   1. \_AuthorizeAPI\_.xml
   2. \_RevokeAPI\_.xml
   3. \_TokenAPI\_.xml

**For example**

|  |
| --- |
| <endpoint>         <address uri="https://{ip\_address\_of\_IS}:{IS\_management\_port}/oauth2/token"/>  </endpoint> |

Make sure you add the MySQL JDBC driver to both servers. I.e. put the .jar file into the <PRODUCT\_HOME>/repository/components/lib directory.

# NO CERTIFICATE ERROR IN CHROME

<https://bugs.chromium.org/p/chromium/issues/detail?id=134418>

# Permission denied error when binding a port

## **Cause**

Ports below 1024 are called **Privileged Ports** and in Linux (and most UNIX flavors and UNIX-like systems), they are not allowed to be opened by any non-root user. This is a security feature originally implemented as a way to prevent a malicious user from setting up a malicious service on a well-known service port.

|  |
| --- |
| There are a few different solutions to work around this:   1. Install and configure [Apache](https://confluence.atlassian.com/doc/using-apache-with-mod_proxy-173669.html) or [nginx](https://confluence.atlassian.com/confkb/how-to-use-nginx-to-proxy-requests-for-confluence-313459790.html) as a reverse proxy server, which can be started as root to open the port, and then downgrade its privileges back to a normal user. 2. Set up a firewall on the server using iptables or an alternative, so that the lower port number is forwarded internally to a higher port number listened by Confluence. 3. Use [jsvc](http://commons.apache.org/proper/commons-daemon/jsvc.html), which is able to open ports as root, and then downgrade privileges. 4. Use [authbind](http://en.wikipedia.org/wiki/Authbind) to grant privileges for a non-root user to open a privileged port. 5. If using Linux 2.6.24 or later, you can set up a file capability on the java executable, to give elevated privileges to allow opening privileged ports only, and no other superuser privileges:   # setcap cap\_net\_bind\_service+ep /path/to/bin/java  After setting this you may notice errors when starting Java like this, for example:  $ java -version  /path/to/bin/java: error while loading shared libraries: libjli.so: cannot open shared object file: No such file or directory  This means that the library is being imported from a dynamic path, and not in the trusted ld.so path. See <http://bugs.sun.com/view_bug.do?bug_id=7157699> for details. To fix this, you need to locate the library, and add its path to the ld.so configuration. Note that the below is an example, and this may differ depending on Linux distribution. Replace JAVA\_HOME with the correct location:  $ find JAVA\_HOME -name 'libjli.so'  JAVA\_HOME/lib/amd64/jli/libjli.so    # echo "JAVA\_HOME/lib/amd64/jli" > /etc/ld.so.conf.d/java-libjli.conf  # ldconfig -v  After setting this all up, you need to make sure that Confluence only starts java with the direct binary path, and not via a symbolic link, otherwise the capability will not be picked up. |

Link: <https://confluence.atlassian.com/confkb/permission-denied-error-when-binding-a-port-290750651.html>

|  |  |
| --- | --- |
| $ | setcap cap\_net\_bind\_service+ep /home/vmadmin/jdk1.7.0\_60/bin/java |
| $ | echo "/home/vmadmin/jdk1.7.0\_60/lib/amd64/jli" > /etc/ld.so.conf.d/java-libjli.conf |
| $ | ldconfig -v |

# SAML 2.0 SSO with Identity server

Source: <https://docs.wso2.com/display/IS530/SAML+2.0+Web+SSO>

|  |
| --- |
| **If the user accesses a service from a service provider:**   1. The service provider determines which identity provider to use (this is the case when there are multiple identity providers. SAML identity provider [discovery profile](http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-idp-discovery.pdf) may be used). 2. The service provider generates a SAML message and then redirects the web browser to the identity provider along with the message. 3. Identity provider authenticates the user. 4. The identity provider generates a SAML message and then redirects the web browser back to the service provider. 5. The service provider processes the SAML message and decides to grant or deny access to the user.   If the user accesses the identity provider directly, then only the steps 3, 4 and 5 are in the flow. |

The following diagram illustrates the scenario:

