ORACLE

Anonymous Login Configuration for Oracle Analytics Server

Describes how to configure Oracle Analytics Server for Anonymous Login.

August 2023, version 1.0

Public

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Oracle Analytics Server Support covers the support on the configuration steps described in this document; however, support and maintenance for the third-party software (non-Oracle software) is outside the scope of Oracle Analytics Server Support.

Revision History

The following revisions have been made to this document since its initial publication.

DATE	REVISION
Aug 2023	Initial publication

Authors: Veera Raghavendra Rao Koka.

Table of Contents

Introduction

An anonymous login allows the embedding of Oracle Analytics Server (OAS) classic reports or dashboards and data visualization (DV) workbooks into an external portal without user authentication.

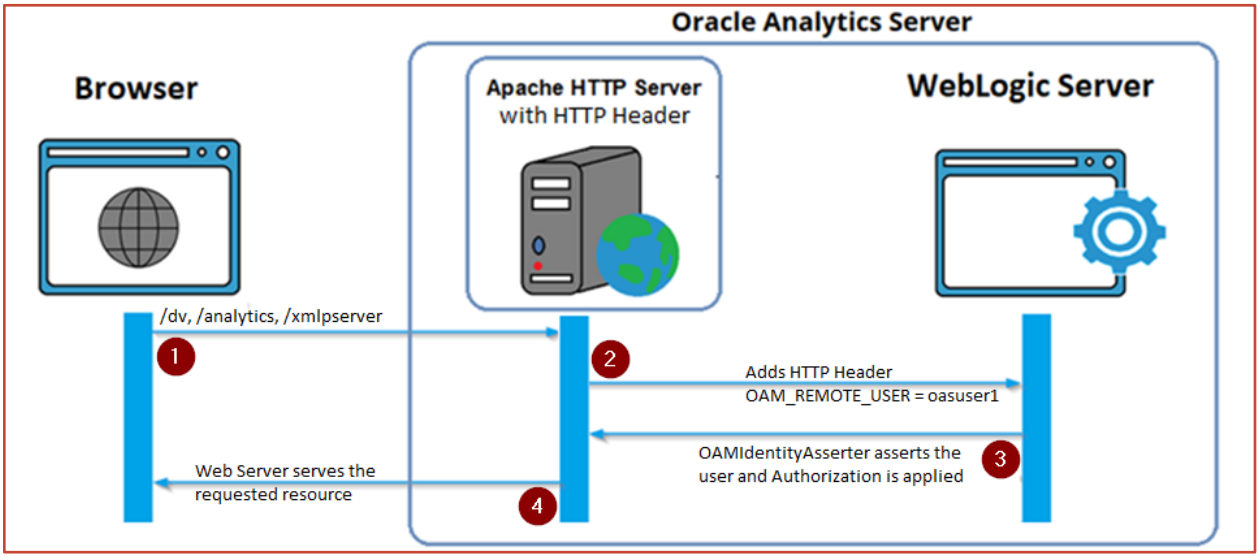
Configuring anonymous login for OAS is only appropriate for public-hosted environments and embedding scenarios and is not advised for general analytics.

Use case 1: If the external portal is a public website embedding OAS reports and workbooks, you can bypass OAS authentication using the anonymous login.

Use case 2: If an external portal utilizes the Go-URL method to access dashboards or reports from Oracle Business Intelligence Enterprise Edition (OBIEE), this is no longer supported from OBIEE version 12.2.1.3 onwards when lightweight SSO or SSO is enabled. In such cases, use the anonymous login for OAS.

Note: when the same SSO provider authenticates both the external portal and OAS, there is no requirement for anonymous login when embedding OAS reports and workbooks.

Approach



Notes:

1. When embedding OAS reports in a public website and OAS and Apache HTTP servers are running on-premises protected by a firewall, configure a load balancer for OAS in the DMZ.
2. When embedding OAS reports in a public website and OAS and Apache HTTP servers are running on Oracle Cloud, create the OAS server in a private subnet and the load balancer in a public subnet.

In a typical scenario with certified SSO providers for OAS like Oracle Access Manager and App Gateway, users accessing the OAS URL are redirected to the SSO Identity Provider for authentication. After a successful authentication, the SSO Identity Provider sends the username in the HTTP header to the OAS WebLogic server, where the WebLogic Identity Asserter asserts the username, applies the authorization, and grants access to the OAS application.

Anonymous login is a hybrid solution that involves a webserver, like Apache HTTP Server or Oracle HTTP Server, simulating the SSO Identity Provider authentication and sending a hard-coded username as the authenticated user in the HTTP header to the backend OAS WebLogic Server.

The OAS WebLogic Server is configured with a WebLogic Identity Asserter to assert the username sent in the HTTP Header. This example uses the OAMIdentityAsserter.

Once the user exists in the default embedded LDAP or an external LDAP, the user is allowed access to OAS per the user's authorization.

Any external portal or user accessing OAS, either via embedding or a direct URL, is automatically logged in and can view the reports, dashboards, and workbooks.

Considerations

OAMIdentityAsserter supports any of these headers: OAM_REMOTE_USER, iv-user, and SM_USER.

With this solution, you need to restrict the privileges and permissions of the hard-coded user only to consume or view reports, and to deny permission to edit or create new OAS reports.

After the configuration, anonymous login is available for /analytics, /dv, and /xmlpserver.

Because this solution is based on sending a username in the HTTP header, any application, browser, tool, or load balancer can also simulate the hard-coded username in an HTTP header and access the OAS application. To manage the security implications, you should restrict access for all end users to the OAS WebLogic managed server (bi_serverN) on port 9502 or 9503.

For example, restrict access to the direct OAS URL like http://oas_server:9502/dv or https://oas_server:9503/dv. You can do it by blocking the 9502 and 9503 ports of the OAS servers in the network. You must also ensure only the webserver can access the OAS servers on ports 9502 and 9503.

The best practice is running a webserver on the OAS Server.

You will also need to suppress the use of the HTTP header in the browser, load balancer, and any external applications, to make sure only the webserver can add values in the HTTP header and send those to the backend OAS WebLogic server.

To more fully understand how to mitigate security issues, read the white paper, Anonymous Login for Oracle Analytics Server, or post questions in the Oracle Analytics community.

Protected, Public, and Excluded List of Resources for OAS

Specific resources are intentionally left unprotected. These resources are accessed through the OAS's basic authentication mechanism (user ID and password) by tools that can't use the SSO.

Refer to OAS documentation, Updating the Protected, Public, and Excluded Resources for an Enterprise Deployment.

Configure Apache HTTP Server as a Web Server for Oracle Analytics Server

References:

See Configure Apache HTTP Server as a Web Server for Oracle Analytics Server.

See SSL Offloading at Web Server for Oracle Analytics Server on Oracle Cloud Marketplace.

Install Apache HTTP Server

Oracle Linux 7: `yum install httpd mod_ssl -y`

Oracle Linux 8: `dnf install httpd mod_ssl -y`

Enable the Service: `systemctl enable httpd.service`

Start the Service: `systemctl start httpd.service`

Stop the Service: `systemctl stop httpd.service`

Configure Apache HTTP Server

Download the files via the link below and unzip those files to the Apache HTTP Server config path, for example, /etc/httpd/conf.d

Download the Zip File: anon_config.zip

Change Apache Configuration

Connect to the Apache HTTP Server as root user.

As a best practice, we suggest installing Apache HTTP Server on the same server hosting OAS.

```
cd /etc/httpd/conf.d
```

```
chmod +x change_httpd_config.sh
chmod +x generate_ssl_certs.sh
```

Run the change_httpd_config.sh script

```
/etc/httpd/conf.d/change_httpd_config.sh
```

The above script will change the Apache HTTP Server configurations to use a more robust cipher suite, TLSv1.2, as the SSL protocol, etc.

Also, configure the below settings.

```
SSLEngine on
SSLProxyEngine on
RequestHeader set WL-Proxy-SSL "true"
RequestHeader set IS_SSL "ssl"
RewriteEngine On
RewriteOptions Inherit
# Uncomment below 5 lines if the backend OAS WebLogic Managed Server runs in SSL port
#SSLVerifyClient none
#SSLProxyVerify none
#SSLProxyCheckPeerName off
#SSLProxyCheckPeerCN off
#SSLProxyCheckPeerExpire off
SSLProtocol TLSv1.2
```

Generate SSL Certificates

```
cd /etc/httpd/conf.d
```

Run the generate_ssl_certs.sh script

```
/etc/httpd/conf.d/generate_ssl_certs.sh
```

This script will generate a private key and CSR file for the Apache HTTP Server with FQDN hostname; you must get the CSR signed by your Certificate Authority (CA).

In this example, we use a self-signed certificate; you can use your CA-signed certificate.

If the Load Balancer (LB) is front ending the Apache Server, we suggest using the LB Hostname as the ServerName in the **httpd.conf** and **ssl.conf** files. Also, use the Load Balancer's SSL Certificates.

Set the Apache ServerName

If there is a Load Balancer in front of the Apache Server, set the Apache ServerName to be Load Balancer FQDN

httpd.conf:

ServerName Apache-or-Load-Balancer-FQDN.com:80

`e.g. ServerName oas.ceal.com:80`

ssl.conf:

ServerName Apache-or-Load-Balancer-FQDN.com:443

`e.g. ServerName oas.ceal.com:443`

Load the Apache Server Configuration Files

The Apache HTTP Server will default load all files in the conf files in the conf.d folder, which loads the below files.

```
vi /etc/httpd/conf/httpd.conf
rewrite_engine.conf
redirect_http_to_https.conf
psr.conf
analytics.conf
```

If the OAS Server environment is a multi-node (clustered) env, rename the below files.

Rename `analytics.conf` to `analytics.conf.txt`.

Rename `analyticsclustered.conf.txt` to `analyticsclustered.conf`

Rename `workers.conf.txt` to `workers.conf`

Enter the OAS server names and port numbers in the workers.conf file.

Start Apache Server

As Root User

```
httpd -k start | systemctl start httpd.service
httpd -k stop | systemctl stop httpd.service
```

Configure OAS WebLogic Server for SSO

We need the hard-coded username for the simulated SSO in the WebLogic server. If the user exists, the OAS server applies the authorization.

WebLogic Administration Console

Log in to WebLogic Administration Console > Security > myrealm > Providers.

Change the Control Flag of "Default Authenticator" to "SUFFICIENT."

Change the Control Flag of External Authenticator (If any exists) like Active Directory or any other LDAP Directory or SQLAuthenticator to "SUFFICIENT."

Add "**OAMIdentityAsserter**" as a Provider and set the Control Flag as "REQUIRED."

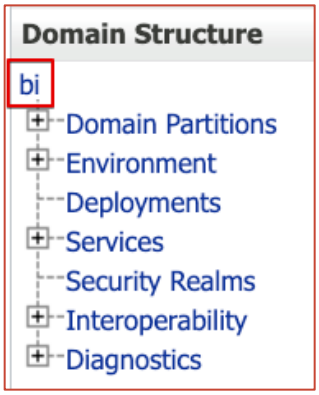
Re-order the list of Authenticators to have the first OAMIdentityAsserter and then the second SQLAuthenticator, and the rest follows.

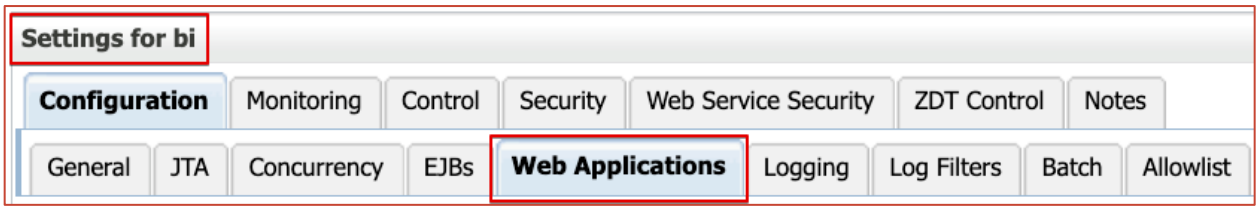
If you have **BISQLGroupProvider**, BISQLGroupProvider should always be the first in the order list.

Enable WebLogic Plugin

Log in to WebLogic Admin Console and Enable WebLogic Plugin

On the top left section, click "bi"> WebApplications > Enable WebLogic Plugin.





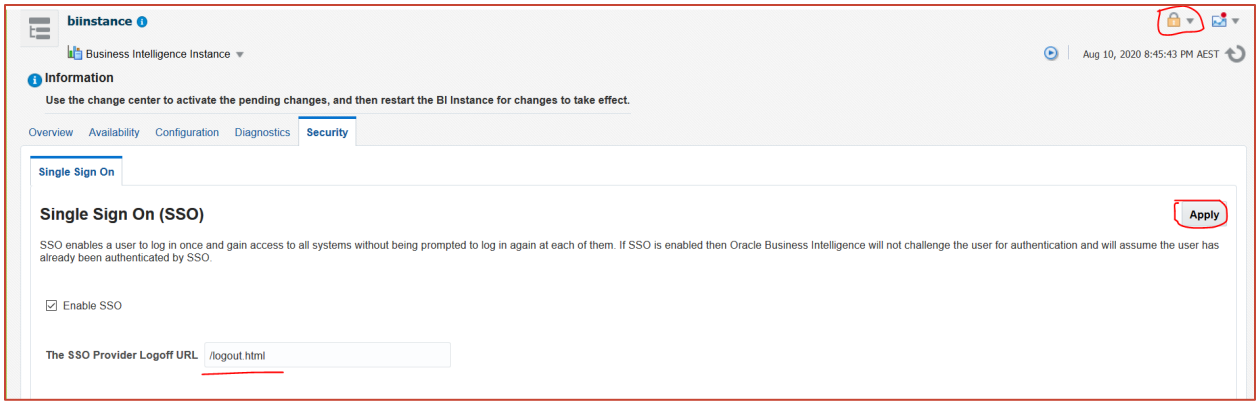
WebLogic FMW EM

If you have multiple Authentication providers, log in to WebLogic FMW EM and add virtualize=true in the **Service Provider Configuration** section.

Config the SSO Logout URL

Log in to WebLogic FMW EM and set the SSO Logout URL as /logout.html

Lock and Edit > Set the logout URL to "`/logout.html`"> Apply > Activate Changes.



Copy the below logout.html file to Apache /var/www/html folder.

```
cp /etc/httpd/conf.d/logout.html /var/www/html/logout.html
```

Restart the OAS Services.

Restart Apache HTTP Server.

OAS Application Roles for the hard-coded user

Assigning the consumer roles (DVConsumer or BIConsumer) to the anonymous login user is recommended.

Using Oracle HTTP Server (OHS) as a Web Server

Configure OHS to Route Requests to the Application Server. See here.

Use the syntax provided in analytics.conf and update the mod_wl_ohs.conf accordingly.

Edit `$DOMAIN_HOME/config/fmwconfig/components/OHS/ohs1/mod_wl_ohs.conf`

Add the following configuration.

```
Define AnonUser oasuser1
RequestHeader unset OAM_REMOTE_USER
<Location /dv>
   SetHandler weblogic-handler
   WebLogicCluster oasserver.company.com:9502
   WLProxySSL ON
   WLProxySSLPassThrough ON
   RequestHeader set OAM_REMOTE_USER ${AnonUser}
</Location>
```

Test the OAS URL's

https://Apache-or-LB-hostname.domain.com/analytics

https://Apache-or-LB-hostname.domain.com/dv

https://Apache-or-LB-hostname.domain.com/xmlpserver

Mitigating the Security Issues

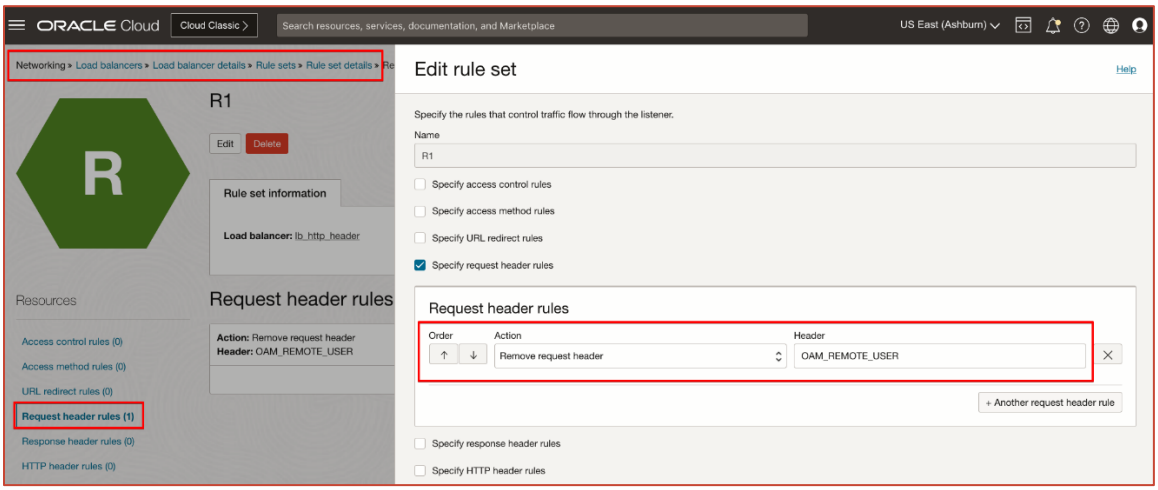
NOTE: Implement the suggested steps to mitigate the HTTP header-based security issues, and you should take complete ownership of the security testing of the HTTP header mitigation.

Because this solution is based on sending a username in the HTTP header, any application, browser, tool, or load balancer can also simulate the hard-coded username in an HTTP header to access the OAS application and misuse the privileges.

For example, a Consumer user can use an Admin username in the header and log in as an Administrator. Here are the mitigations to avoid such issues: Always allow only the Apache HTTP Server to set the username in the HTTP header.

1. If there is a Load balancer front ending the Apache HTTP Server, which further front ends the OAS WebLogic Server, the HTTP header like OAM_REMOTE_USER, SM_USER, and iv-user should be unset from the Load balancer.

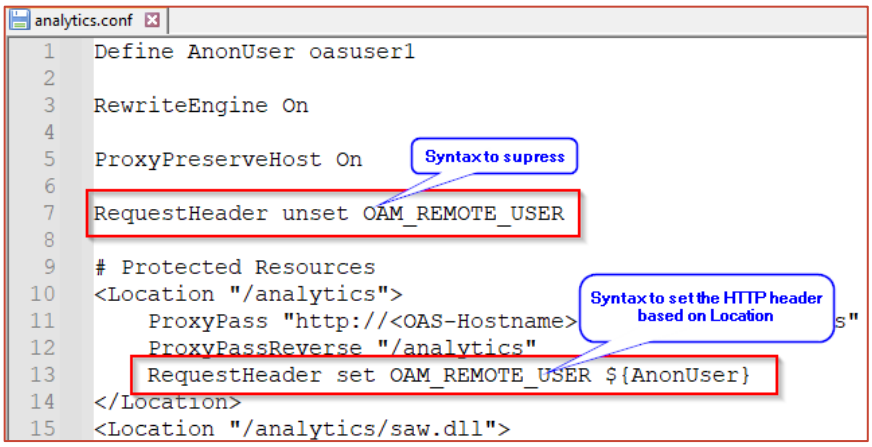
   Solution: At the Load balancer, unset the HTTP header as shown in the below example for an OCI Load balancer.



   Add the rule to the LB listener.

2. Suppose the Apache HTTP Server front ends the OAS WebLogic Server without a load balancer. In that case, the Apache HTTP Server should only set the HTTP header and suppress any HTTP header set by external tools or browsers.

   Solution: The analytics.conf file, which loads the OAS configuration in the Apache HTTP Server, has already been handled to manage such configuration. For example, below is the code snippet.



3. The browsers or other tools should not be able to send the username in the HTTP header directly to the backend OAS WebLogic managed server like bi_server(N) at its non-SSL and SSL ports.
   Solution: Block the direct access to OAS WebLogic managed server bi_server(N) at 9502 or 9503 outside the OAS Server.

Configuring WebLogic to prevent direct access to BI

Follow the WebLogic documentation to configure a Connection Filter so that only the Apache HTTP Server and machines running BI components are allowed to access the WebLogic server:

1. Instructions for configuring the default connection filter are in the "Using Connection Filters" section in the WebLogic documentation at Using Network Connection Filters.
2. Instructions on writing an appropriate filter rule are in the "Guidelines for Writing Connection Filter Rules" section in WebLogic documentation at Guidelines for Writing Connection Filter Rules.
3. Your filter rule should look like this:

```
[Apache http server IP Address] * [WebLogic Admin Server Port] allow
[Apache http IP Address] * [WebLogic Managed Server Port] allow
[BI component server IP Address] * [WebLogic Admin Server Port] allow
[Another BI component server IP Address (if it exists)] * [WebLogic Managed Server Port] allow
0.0.0.0/0 * * deny
```

Test that you can access the WebLogic Administration Console and Analytics URLs via the web server but not directly from any other machine.

Protecting direct HTTP access to OBIPS

Follow the guidance in the 'Managing Security for Oracle Analytics Server' documentation guide, SSO Implementation Considerations.

For example:

```
<Listener port="XXXX" ssl="false">
   <Firewall>
      <Allow address="127.0.0.1"/>
      <Allow address="XXX.XXX.X.XXX"/>
      <Allow address="XXX.XXX.X.XXY"/>
   </Firewall>
</Listener>
```

Web Server Configuration for General Analytics Usage

When using the OAS environment for public hosting and embedding scenarios, we recommend using the anonymous login configuration only for this purpose and refrain from using this configuration for general analytics usage.

If you use the same OAS environment for public hosting & embedding scenarios and general analytics, follow the suggested approaches below.

Approach 1

On the same OAS server, install and configure two Apache HTTP Servers or two Oracle HTTP Servers or one Apache HTTP Server and one Oracle HTTP Server. Use one of the web servers with anonymous login configuration for public hosting and embedding scenarios. Use the other web server without the anonymous login configuration for general analytics usage.

Approach 2

Install Apache HTTP or Oracle HTTP server on the OAS server and configure the web server with two virtual hosts. Use one virtual host for anonymous login and the other for general analytics.

Follow the below steps to do it:

1. In the **httpd.conf** file of the Apache HTTP Server, comment on the line that loads all the conf files existing in the **conf.d** folder and explicitly loads the required conf files.

   For example, as the root user login to the Apache web server

   ```
   sudo su root
   cd /etc/httpd/conf
   vi httpd.conf
   ```

   Comment on the below line.

   ```
   IncludeOptional conf.d/*.conf
   ```

   After the comment, the line should look as below:

   ```
   # IncludeOptional conf.d/*.conf
   ```

   Load the required files explicitly.

   ```
   Include conf.d/psr.conf
   Include conf.d/redirect_http_to_https.conf
   Include conf.d/rewrite_engine.conf
   ```

```
Include conf.d/ssl.conf
```

2. Default **ssl.conf** file listens at 443 port and the virtualhost as apache-or-lb-hostname.com:443

   Copy the ssl.conf to ssl2.conf and modify the Listen port to 4443 and virtualhost as apache-or-lb-hostname.com:4443.

   For example, as the root user login to the Apache web server

   ```
   sudo su root
   cd /etc/httpd/conf.d
   cp ssl.conf ssl2.conf
   vi ssl2.conf
   ```

   Change the Listen port no and the port no in the virtual host, keeping the rest of the configuration the same.

   After the changes, ssl2.conf should look as below:

   ```
   Listen 4443 https
   ...
   <VirtualHost _default_:4443>
   # General setup for the virtual host, inherited from global configuration
   #DocumentRoot "/var/www/html"
   ServerName apache-or-lb-hostname.com:4443
   ...
   ```

3. Load the ssl2.conf in the httpd.conf file.

   For example, as the root user login to the Apache web server

   ```
   sudo su root
   cd /etc/httpd/conf
   vi httpd.conf
   ```

   It should look as below:

   ```
   # IncludeOptional conf.d/*.conf
   Include conf.d/psr.conf
   Include conf.d/redirect_http_to_https.conf
   Include conf.d/rewrite_engine.conf
   Include conf.d/ssl.conf
   Include conf.d/ssl2.conf
   ```

4. Instead of loading the analytics.conf file in the httpd.conf file, create two different analytics.conf files, one with anonymous login configuration and another for general analytics usage load them in the respective ssl.conf files just before the closure of the virtualhost tag.

   For example, as the root user login to the Apache web server

   ```
   sudo su root
   cd /etc/httpd/conf.d
   cp analytics.conf general_analytics.conf
   ```

Remove the definition of AnonUser and the RequestHeader statement from the general_analytics.conf file.

```
Define AnonUser oasuser1
RequestHeader set OAM_REMOTE_USER ${AnonUser}
```

Edit the ssl.conf and ssl2.conf files and add the load statement to load the analytics.conf and general_analytics.conf respectively.

vi ssl.conf

```
<VirtualHost _default_:443>
...
Include conf.d/analytics.conf
</VirtualHost>
```

vi ssl2.conf

```
<VirtualHost _default_:4443>
...
Include conf.d/general_analytics.conf
</VirtualHost>
```

5. Restart the Apache HTTP Server.

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle