Oracle™ Enterprise Manager Grid Control – Composite Application Monitor and Modeler (CAMM) 10.2.0.4 Configuration Guide

The specifications and information in this manual are subject to change without notice. All statements and recommendations in this manual are believed to be accurate but are presented without warranty of any kind, express or implied. Users must take full responsibility for their application of any products.

1 Overview

1.1 Introduction

The “Oracle Enterprise Manager Grid Control – Composite Application Monitor and Modeler (CAMM) 10.2.0.4 Configuration Guide” covers various CAMM 10.2.0.4 configuration and tune-up aspects.

1.2 Audience

The “CAMM 10.2.0.4 Configuration Guide” will be primarily used by the personnel who are responsible for the deployment of the CAMM 10.2.0.4 product in staging and production environments. These could be administrators, operation support specialists, architects or similar. The reader is not expected to have programming skills but is expected to know the basics CAMM installation procedures.

2 Configuration options

3 Configuration directory and files

After CAMM is installed all the components of the application package are located in $CAMM_HOME directory.

3.1 Directory structure

The following is directory structure of CAMM once it is installed:

```
$CAMM_HOME
        - bin
        - config
        - container
        - darchive
        - database
        - deploy
        - docs
        - j2sdk
        - lib
        - log
        - schema
        - tmp
        - Uninstall CAMM
```

bin directory has all the executable files to start and stop CAMM, run *deployer* for Agent and CAMM EJB, run export utility.

config directory contains all the CAMM runtime configuration parameters that control execution logic, CAMM schemas enablement, CAMM GUI functionality, Service Level Objectives definition, export logic and many more.

container has Tomcat server where CAMM Web Application (GUI) is located

database contains the data repository executables and performance and metadata storage files for CAMM.

deploy directory contains agent libraries and configuration files, as well as *CAMM EJB* and *CAMM Admin Web* Application. These components are deployed on the remote host (Web or Application servers) using *deployer* utility found in *bin* directory of CAMM package.

docs contains CAMM documentation

j2sdk contains minimum Java SDK sufficient to run CAMM Server.

lib has all the libraries required for CAMM's proper functionality

schema contains all the metadata definitions used for CAMM's needs

tmp is used by CAMM for temporary data storage during the runtime

Uninstall CAMM directory contains utility used to uninstall the CAMM.

The following directories are created once CAMM is up and running:

darchive stores temporary images for J2EE applications that need to be monitored as well as the results of analysis and modeling of the J2EE application.

log has all the diagnostics records of CAMM performance metrics collection activities. Also logs indicating successful deployment of CAMM Agents can be found here.

3.2 Config directory

Config directory has many files that potentially can be configured and make CAMM to run in a particular way. Any changes applied to files in this directory require restarting CAMM server.

Most of the files get never touched directly by user. The following are the main three files which can be configured manually to achieve desired effect:

Acsera.properties this file is main CAMM configuration file customization of which helps to tune up CAMM. Chapter 4.1 is completely dedicated to this file.

configuration.xml in this file you define location of Administration Server and credentials to access it. Usually you don't touch this file. The entire configuration is done through CAMM GUI. You can read more details in Chapter 4.2.

export.xml contains information that drives proper data export logic. It is used for manual and automatic export of performance metric and events data from the CAMM Data Repository. Chapter 4.3 digs into details on this.

It is worth mentioning that Service Level Objective definitions and Actions associated with the SLOs are described in *slo.xml* and *event.xml* respectively. The content of these files is completely controlled by definitions applied from CAMM GUI (configuration tab).

By default CAMM is not distributed with license key. Absence of *license.xml* indicates that CAMM will not be functioning properly.

```
AccessControl.xml       configuration.xml         metricmodel.xml
acseradomains.txt       database-mysql.xml        odelschema.xml
AcseraManagerKey.jks    description.xml           operation.xml
AcseraManagerTrust.jks  event-export.xml          OSMetrics.properties
Acsera.properties       events.txt                OSMip.xml
Acsera-ui.properties    event.xml                 policy.all
acserav1.mib            ibmjmxmip.xml             ServiceMetadata.dtd
acserav2.mib            infrastructuremodel.xml   ServiceMetadata.xml
```

Acsera-war.properties	*JAgent.properties*	*slo.config*
analysis.xml	*javamip.xml*	*sloschema.xml*
beajmxmip.xml	*license.pub*	*slo.xml*
castormapping.xml	*license.xml*	*SQLMip.xml*
configschema.xml	*metric-export.xml*	*ui.xml*

3.3 bin directory

bin directory has all the executable files to start and stop CAMM, run *deployer* for Agent and CAMM EJB, run export utility.

acseraenvlegacy.sh	*acshut.sh*	*readVersion.sh*	*runwas.sh*
acseraenv.sh	*checkregistry.sh*	*runExportEvent.sh*	*standalonelegacy.sh*
acseraenvwas.sh	*client.sh*	*runExportMetric.sh*	*standalone.sh*
acseralegacy.sh	*deployer.sh*	*runlegacy.sh*	*weblogicDeployer.sh*
acsera.sh	*encrypt.sh*	*run.sh*	*websphereDeployer.sh*

There are two main reasons why one would need to customize the content of the files in this directory:

- To change the pointers to the location of Java Runtime Environment and CAMM installation directory (CAMM_HOME)
- To configure CAMM Server JVM parameters, e.g. memory.

Only the files that are frequently customized will be described in this chapter.

All the files in this directory should have these lines, pointing to the CAMM Installation location:

For UNIX: *CAMM_HOME=/home/CAMM; export CAMM_HOME*

For Windows: *set CAMM_HOME=C:\CAMM*

File Name (.sh, .bat)	Used for
acsera	*Main executable that start CAMM*
acseraadmin.exe	*Tomcat server executable started as Windows service*
acseraenv	*Sets up environment dependencies, called by run*
acseraenvlegacy	*Loads older version libs (used rarely, for some apps)*
acseraenvwas	*Sets up environment dependencies for WebSphere, called by webSphere deployer script*
acseralegacy	*Loads older version libs (is this defunct now?)*
acseramanager.exe	*CAMM Server executable started as Windows service*
acserarmiregistry.exe	*(defunct)*
acshut	*Main executable that stops CAMM*
admin	*Run Tomcat server to start CAMM admin Web Application*
checkregistry	*(defunct)*
client	*run CAMM GUI Client*
createadminservice	*Create Windows service for admin server (runs Tomcat)*
createmanagerservice	*Create Windows service for CAMM*
createsqlservice	*Create Windows service for CAMM Data Repository*
creatermiregistryservice	*(defunct)*
deployer	*run to deploy CAMM Agents and CAMM EJB*
encrypt	*Encrypts application server password*
loadAccess	*(defunct)*
readversion	*Used by other scripts*
run	*Starts CAMM JVM (called by CAMM)*
runExportEvent	*Execute manual export of events (SLO notifications)*
runExportMetric	*Execute manual export of performance metrics*
runlegacy	*Run CAMM in legacy mode (with older libs)*

runwas	*Run CAMM deployer JVM for WebSphere*
setVersion	*Used by other scripts*
Standalone	*Run CAMM in standalone mode*
Standalonelegacy	*Run CAMM in standalone app legacy mode*
weblogicDeployer	*(defunct)*
websphereDeployer	*Run to deploy CAMM Agent for WebSphere*

In addition to this *acseraenv.sh(.bat)* should have pointer to the JDK used by CAMM:
For UNIX: JAVA_*HOME=/home/CAMM/j2sdk;export JAVA_HOME*
For Windows: *set JAVA_HOME=C:\CAMM\j2sdk*

CAMM is Java application and runs in its own JVM. Default size of JVM memory is 600 MB. Should you desire to change this value you can do it by modifying *–Xms* and *–Xmx* parameters in *acsera.sh (.bat)* file.

Bear in mind that if you have installed CAMM as Windows service and need to change JVM Memory size you need to do it either by updating Windows Registry or rerunning the *createmanagerservice.bat* utility with the new JVM parameters.

3.4 Deploy Directory

Deploy directory contains CAMM Java Agent and OS Metric Agent distributables, including configuration files as well as corresponding libraries. These files are copied to the target systems hosting the Managed Servers when running the deployer utility. Rarely one needs to modify configuration files in this directory. Remember though if you modify the files they will be distributed to ALL targets within single server/cluster.

3.5 Container Directory

Container directory has same structure and content as the standard Tomcat distributable. One interesting point about it is that CAMM GUI application acseraadimn.war is residing in *$CAMM_HOME/container/webapps* directory.

3.6 Database Directory

Database directory is a CAMM Data Repository, where CAMM meta data, performance metrics, events and other data is stored. Structure of this directory is similar to MySQL distributable. CAMM provides modified *my.cnf* though to optimize MySQL performance for CAMM querying needs.

4 Configuration Details

4.1 Acsera.properties file

4.1.1 Log Files Management

This section of *Acsera.properties* file defines log files roll policies. *Log.MaxFiles* indicates max number of log files available at any given moment, whereas the *Log.MaxFileSizeMB* indicates maximum size of the log file. The log files are rolling, i.e. once maximum number of log files reached the first log file will be overwritten and so on:

```
Log.CopyOut = false
Log.MaxFiles = 10
Log.MaxFileSizeMB = 30
Log.MergeLogs = true

Debug.CopyOut = false
Debug.LogLevel = all
Debug.MaxFiles = 10
Debug.MaxFileSizeMB = 30
```

Log files are stored in:
$CAMM_HOME/log directory.
Log directory is not created unless *deployer* utility or CAMM runs at least once.

4.1.2 CAMM Configuration files load

This section sets the pointers to the various configuration files responsible for different aspects of CAMM functioning. All the files are located in *$CAMM_HOME/config* directory.

```
ConfigurationManager.AnalysisSchemaFile=analysis.xml
ConfigurationManager.ConfigFile=configuration.xml
ConfigurationManager.ConfigSchemaFile=configschema.xml
ConfigurationManager.DescriptionFile=description.xml
ConfigurationManager.LicenseSchemaFile=license.xml
ConfigurationManager.ModelSchemaFile=modelschema.xml
ConfigurationManager.OperationFile=operation.xml
ConfigurationManager.SLOSchemaFile=sloschema.xml
```

It is practically not required to do any changes in this section.

4.1.3 Security Features configuration

4.1.3.1 User password management

The following are the parameters to enforce password policies:

- length
- expiration
- complexity, and
- allowed special characters.

```
ConfigurationManager.PasswordMinLength=0
ConfigurationManager.PasswordMaxLength=0
ConfigurationManager.PasswordExpDays=
ConfigurationManager.PasswordMinLetters=0
ConfigurationManager.PasswordMinDigits=0
ConfigurationManager.PasswordMinSecialChars=0
ConfigurationManager.PasswordSecialCharSet=!@#$%^&*()_+[]
```

The default is no length or complexity or expiration day limitation.
Password length control equal to 0 means no restriction. Any other number, we will check for min and/or max length.

Password expiration – if any number set then the password will expired after the specified days.

Password complexity – forces to check letter, digits, special characters. Default is no complexity check. If you set any number on any of these properties the CAMM will check complexity. For example, *PasswordMinLetters=2* means you must have at least 2 letters in your password.

For special characters checking, there is a list of default special characters. This is configurable.

4.1.3.2 CAMM User Session time out

CAMM User Session can be set to time out after specified time.

```
# ConfigurationManager.SessionTimeout's unit is minutes.
ConfigurationManager.SessionTimeout=10
```

The default is 10 minutes, which means that after 10 minutes of no activity on the applet the applet will close automatically and redirect the user back to the login screen.

4.1.4 Multi-domain monitoring configuration

One can limit number of domains to be monitored by setting resource limit parameter: ConfigurationManager.ResourceLimit=2

```
ConfigurationManager.ResourceLimit=2
```

The default is 2, this means that by default CAMM is set to monitor no more than two Application Server domains.

4.1.5 CAMM RMI Port assignment

CAMM uses RMI ports for communication with the agents and collects incoming performance metrics from particular RMI port. By default RMI port is set on the same machine that hosts CAMM. *RMI.Registry.Host* needs to be un-commented and have value other than localhost if the host is multi-homed (i.e. many network interfaces) and you need to make sure that CAMM listens to the incoming traffic on particular interface.

You may need to change *RMI.Registry.Port* value in case the default 51099 port number has been allocated to other application. Also if CAMM is running in multi-instance mode the port number shall be different from instance to instance.

```
#RMI.Registry.Host = localhost
RMI.Registry.Port = 51099
RMI.Registry.UseExternal = false  [NK- what exactly this means?]
```

4.1.6 CAMM Internal Data Repository configuration

CAMM is tightly integrated with MySQL, which is internal repository for metadata and real-time performance metric. Default CAMM MySQL configuration is optimized to ensure minimal GUI queries and performance metrics insertion latencies.

```
#RMI.Registry.Host = localhost
DataManager.MySQL.DbDriverName = com.mysql.jdbc.Driver
# {0} = DbConnectionType
# {1} = DbConnectorDbms
# {2} = DbHost
# {3} = DbPort
# {4} = DbLocalName
DataManager.MySQL.DbUrlFormat = {0}:{1}://{2}:{3}/{4}
DataManager.MySQL.DbConnectorType = jdbc
DataManager.MySQL.DbConnectorDbms = mysql
DataManager.MySQL.DbHost = localhost
DataManager.MySQL.DbPort = 53306
DataManager.MySQL.DbName = mysql
DataManager.MySQL.DbUser = acsera
DataManager.MySQL.DbPassword = acserapass
DataManager.MySQL.DbLocalName = acsera
DataManager.MySQL.InListLimit = 0
```

If your database runs on machine other than the one hosting CAMM then you need explicitly point to MySQL database host by setting *DataManager.MySQL.DbHost* parameter.

CAMM uses its internal instance of MySQL database. If you are running multiple instances of MySQL database on the same machine you need to specify proper port number in the *DataManager.MySQL.DbPort* parameter which is by default 53306.

Pair *DataManager.MySQL.DbUser* and *DataManager.MySQL.DbPassword* define user credentials that allow to access the database. By default they are empty.

The name of the database to be used by CAMM is specified in *DataManager.MySQL.DbLocalName* parameter. If you use simultaneously running multiple instances of CAMM then the name of the database shall be different from instance to instance.

4.1.7 CAMM Export Configuration

CAMM stores real-time performance metrics in its internal data repository (MySQL database). If you want to store this data in your historical data repository, CAMM provides automatic means for performance data export. You can control frequency of export runs, time when the export should run and time range of export data within a day.

```
# Setting for integrated export
AggregationManager.IntegratedExport = false
AggregationManager.ExportDataStartHour = 0
AggregationManager.ExportDataEndHour = 0
AggregationManager.ExportDataSetRangeInHour = 4
AggregationManager.ExportDataSetIntervalInHour = 1
AggregationManager.ExportDataSetDelay = 10000
AggregationManager.ExportStartTime = 0
AggregationManager.ExportEndTime = 0
AggregationManager.ExportFilePurgeTime = 10d
```

By default automatic data export feature is disabled. To enable it set *AggregationManager.IntegratedExport* parameter to “true”.

A pair of parameters *AggregationManager.ExportDataStartHour* and *AggregationManager.ExportDataEndHour* indicate time range within a day, that you are interested in performance data export in. By default it is 24 hrs.
Parameter *AggregationManager.ExportDataSetRangeInHour* shows how much worth of data is stored in each export file. By default it is 4 hours worth of data. This means that CAMM will create multiple export data files with 4 hours worth of data.

In order to minimize export query impact on normal CAMM performance data collection functionality you need to spread out lengthy data exporting queries. This is achieved by setting *AggregationManager.ExportDataSetIntervalInHour* parameter. By default it is 1 hr. This means that export query thread will be running every hour. *AggregationManager.ExportDataSetDelay* defines cool down time for consecutive export queries. By default it is 10 sec. This means that next export query will happen not early than 10 seconds after the previous one.

AggregationManager.ExportFilePurgeTime indicates how many days the export data file will be available before CAMM deletes it. By default it is 10 days.

Current default values are optimal and this section should not be changed except from enabling the automatic export feature.

Automatic export function relies on the definition of the data to be exported by looking into *$CAMM_HOME/config/export.xml* file and exports performance data based on the rules defined in this file.

4.1.8 CAMM Aggregation and Data Life time Configuration

CAMM has sophisticated multi-tiered logic for aggregation (or compression) of performance data. This helps to optimize performance of interaction with the internal data repository (MySQL) both when querying data for presentation or inserting new performance metrics.

By default aggregation is On (*AggregationManager.AggregationOn* parameter).

```
# Aggregation related setting
MetricQueryAdaptorImpl.QueryRawTierOnly = false
AggregationManager.Trace = false
AggregationManager.AggregationOn = true
AggregationManager.AggregateStartDelay = 0
AggregationManager.AggregateTaskInterleaveSecs = -1
AggregationManager.InsertUnits = 1000
AggregationManager.InsertDelayInSecs = 5
AggregationManager.AggregateIntervalDivider = 4
```

CAMM has multi-tiered compression logic. Tier 0 – very granular uncompressed data. Tier 1 – compression of Tier 0 data. Tier 2 is further compression of Tier 1 data.

For every tier there are 3 parameters that define logic of data compression: *Metric.Grain.[0-2]* – sets granularity level of compressed data. Typically in Tier 0 this value is 0, which means no compression, data is as granular as it can be (driven by global sampling rate value). Tier 1 is set to 3 minutes by default, this means that Tier 1

performance data will have one record per 3 min of particular metric (average). Tier 2 has this value set to 30 min, i.e. each record in this tier is average of 30 min of performance data.
Note: in production environment you need to set *Model.GlobalSamplingRateSeconds* for monitoring sampling rate. By default it is 60 seconds. This switch equals to setting *Metric.Grain.0*.

Metric.TableInterval.[0-2] – indicates how much of data is stored in each database table for that tier. Tier 0 stores 4 hours of data in each table before creating new table. Tier 1 stores one day worth of compressed information. And Tier 2 stores 7 days worth of compressed metrics in one table before creating a new table.

Metric.DataLife.[0-2]- defines how long data will stay in the table of corresponding tier before it is deleted.

```
############################
# Production setting
# NOTE: use Model.GlobalSamplingRateSecs to configure Metric.Grain.0
############################
Metric.Grain.0 = 0s
Metric.TableInterval.0 = 4h
Metric.DataLife.0 = 1d

Metric.Grain.1 = 3m
Metric.TableInterval.1 = 1d
Metric.DataLife.1 = 8d

Metric.Grain.2 = 30m
Metric.TableInterval.2 = 7d
Metric.DataLife.2 = 7d

MetricPurging.DataLifetimeHours = 24
EventPurging.ExpirationHours = 24
```

MetricPurging.DataLifetimeHours – overrides the Metric.DataLife.0 parameter [NK-clarify this]

EventPurging.ExpirationHours – this parameter sets life time for Events data (SLO violations) before it is deleted.

4.1.9 Aggregating incoming metrics on fly

CAMM by default is aggregating data coming from multiple entities of the cluster thus minimizing rate of insertion in to the data repository. This greatly improves performance of CAMM in heavily loaded environment.

As a side effect of this approach though, the user is unable to see metrics from instrumentation (processes and portals) on per resource (server) level. If you need to enable this then set the *JavaMIP.AggregateInserts* to "*false*".

4.1.10 Process Tracking configuration – obsolete

CAMM has two ways of acquiring process execution metrics:

- via byte code instrumentation of process methods or
- reliance on WebLogic process tracking information

the latter is obsolete.
The following parameters are used in case if Process Tracking option is used for process performance metrics gathering.

```
BEAWLIEventConsumer.Trace=false
BEAWLIEventConsumer.DebugQuery = false
BEAWLIEventConsumer.DeleteAllWliProcessEventTableRows = false
BEAWLIEventConsumer.DeleteSelectedWliProcessEventTableRows = false
BEAWLIEventConsumer.UseEventTimeForElapsedTime = false
```

4.1.11 Configuring Architectural Analysis

By default Architectural Analysis is enabled.

To exclude unnecessary and expensive architectural analysis of specific application(s), specify the application(s) after *Model.ArchitectureAnalysis.Application.Exclude=*. Use comma to delimit multiple items.

To include specific application(s) for architecture analysis, specify the application(s) after *Model.ArchitectureAnalysis.Application.Include=*. Use comma to delimit multiple items.

To completely disable Architectural Analysis set *Model.ArchitectureAnalysis* to *"false"*.

```
# Control which applications to analyze
#
Model.ArchitectureAnalysis=true
Model.ArchitectureAnalysis.CompileJSP=true
Model.ArchitectureAnalysis.CompileJSPSingle=false
Model.ArchitectureAnalysis.TypeMask=J2EE.Container,J2EE.Container.Component,J2EE.D
eploymentDescription.Component,J2EE.DeploymentDescription.EAR
Model.ArchitectureAnalysis.Application.Include=
Model.ArchitectureAnalysis.Application.Exclude=
```

4.1.12 Configuring list of applications to be monitored or excluded from monitoring

To avoid overhead of unnecessary monitoring of certain applications you can explicitly state which applications to monitor, or which applications to exclude from monitoring.

```
# Control which applications to analyze
#
ComponentProvider.Application.Include=
ComponentProvider.Application.Exclude=WLI System EJBs,WLI-AI Design-
time,B2BDefaultWebAppApplication,WLI
Worklist,JWSQueueTransport,Deployer,BEA_WLS_DBMS_ADK,Acsera
```

The following block of parameters specifies what types of applications is desired to be analyzed.

```
# Control which applications to analyze by type
#
ComponentProvider.Application.WLI=true
ComponentProvider.Application.WLW=true
ComponentProvider.Application.Portal=true
ComponentProvider.Application.Struts=true
ComponentProvider.Application.J2EE=true
```

4.1.13 OS Metric configuration

This section describes the attributes of OS Metric subsystem.
RMI.Registry.OSMetric.Port points to the port on remote system from which the OS metrics is accessed. Direct RMI connection is the default mode
OSMIP.UseOSMetricEJBToCollect instructs to use CAMM OS Metric EJB as alternative to direct connection for OS metrics collection. One would use this option to overcome Firewall issues.
OSMIP.ExcludeIPAddress indicates which computers (IP addresses) are excluded from OS metrics monitoring.

```
# Control System Performance Metrics
#
RMI.Registry.OSMetrics.Port = 51099

OSMIP.Trace = false
OSMIP.UseOSMetricEJBToCollect = false
OSMIP.ExcludeIPAddress =
```

4.1.14 Firewall mitigation (for couple of internal RMI ports)

There are two more internal ports used. They are explicitly defined to let proper Firewall configuration that would not interfere with CAMM's normal operation. If Firewall is not

an issue then *RMI.Anonymous.Server.Port.enabled* can be set to "*true*" and CAMM automatically will pick up the available ports.

```
# RMI Ports
#
RMI.Anonymous.Server.Port.enabled=false
RMI.RemoteServiceController.ServerPort=55000
RMI.AgentLiason.ServerPort=55001
```

4.1.15 SLO Dampening

There are times when user deliberately wants to cut down on number of repeated notifications should SLO violation persist for give period of time. In order to suppress "annoying" notifications of the same violation in short period of time, CAMM provides SLO Dampening feature. Once enabled, should SLO violation occur and be repeated several times in short period CAMM will not fire SLO violation notification for the time period defined in *SLO.RearmDelay*. To disable this feature just set the value of this parameter to 0.
SLO.SuppressDelayedAsserts indicates if violation still persists upon time period expiry should CAMM fire the SLO notification. By default it is "*false*", i.e. fire the notification.

```
# The following property is specified in units of
# minutes (m), hours (h) or days (d)
SLO.RearmDelay = 15m
SLO.SuppressDelayedAsserts = false
```

4.2 Configuration.xml file

The configuration file is a place where all pointers and credentials to direct CAMM to Administration Server. You can customize this file either manually or by running CAMM and configuring new domains to be monitored from confi-guration GUI. Bear in mind you need to run CAMM at least once to configure the Domain Administration Server credential and address as well as add other domains before restarting CAMM to actually monitor newly defined domains.

Modifying this file from CAMM administration GUI is preferred way. It intuitive to enable and disable various features on per domain level, like instrumentation and OS Metric collection.

Any modification in this file requires restart CAMM for new configuration to take effect.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<configuration verbose="true" xmlns="http://www.acsera.com/ns/configuration">
 <infrastructures name="default">
  <infrastructure enabled="true" name="Acsera">
   <resource name="WebLogic" enabled="true">
    <resourceDefinition>WebLogic_8.1.3</resourceDefinition>
    <mipProfiles>
     <mipProfile enabled="true" name="JavaProf1">
      <mip>Java</mip>
     </mipProfile>
     <mipProfile enabled="true" name="JMXProf1">
      <mip>BEAJMXMIP</mip>
      <configParameter>
       <key>username</key>
       <value>weblogic_user_name</value>
      </configParameter>
      <configParameter>
       <key>password</key>
       <value>password_name</value>
      </configParameter>
      <configParameter>
       <key>protocol</key>
       <value>t3</value>
      </configParameter>
      <configParameter>
       <key>host</key>
       <value>Admin Server Host Name or IP Address</value>
      </configParameter>
      <configParameter>
       <key>port</key>
       <value>Admin Server Port</value>
      </configParameter>
     </mipProfile>
     <mipProfile enabled="true" name="OSProf1">
      <mip>OS</mip>
     </mipProfile>
    </mipProfiles>
   </resource>
  </infrastructure>
 </infrastructures>
</configuration>
```

4.3 Export.xml

This file contains all the directives and filters required to succeed in performance metrics and SLO violation events export. This file is used by CAMMs Integrated Automatic export feature as well as manual export scripts.

exportMetric and *exportEvent* attributes define if performance metric and events (SLO violation notifications) are required to be exported. *metricDataGrain* indicates granularity of data to be exported, by default it is 3 minutes.

Next type *output* directs which way export data is desired to be stored: either as records in database or as records in files. In case of database storage a proper database driver must be provided. For files the directory name is sufficient.

entitiyTypes section defines what to export. If the *exportAllTypes* is set to true then all the metrics will be exported. Otherwise only the *entityType* indicated in *entityTypes* block will be exported.

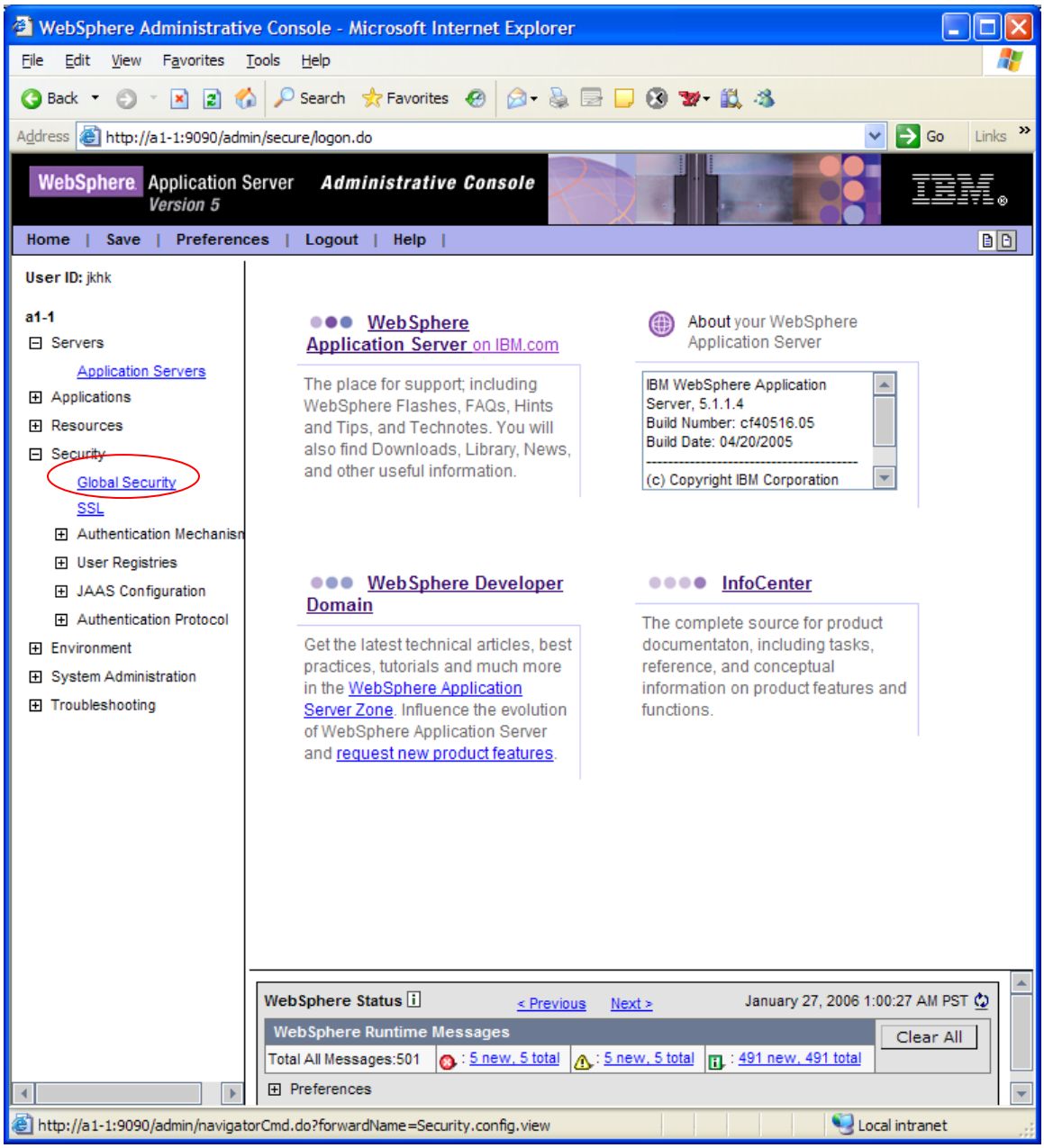
```
<?xml version="1.0" encoding="UTF-8"?>
<export xmlns="http://www.acsera.com/ns/export" verbose="true" exportMetric="true"
exportEvent="true" metricDataGrain="180s" exportFullMetric="true">
<!--
 <output type="jdbc" convertTimeFormat="true"
arguments="access,metric,sun.jdbc.odbc.JdbcOdbcDriver,jdbc:odbc:acsera"/>
-->
 <output type="file" convertTimeFormat="false"
arguments="/home/acsera/acsera/export,metric"/>

 <entityTypes exportAllTypes="false">
        <entityType name="BEA.ProcessNode"/>
        <entityType name="J2EE.Dispatcher"/>
        <entityType name="J2EE.JDBC.ConnectionPool"/>
        <entityType name="J2EE.JVM"/>
 </entityTypes>
 <!--extra filter -->
 <!--
 <filters>
  <filter key="containerID" values="cgServer"/>
 </filters>
 -->
 <!-- don't modify this -->
 <columns>
  <column header="Timestamp" type="Timestamp"/>
  <column header="EntityID" type="EntityID"/>
  <column header="Application" type="Entity" key="applicationID" default=""/>
 </columns>
</export>
```

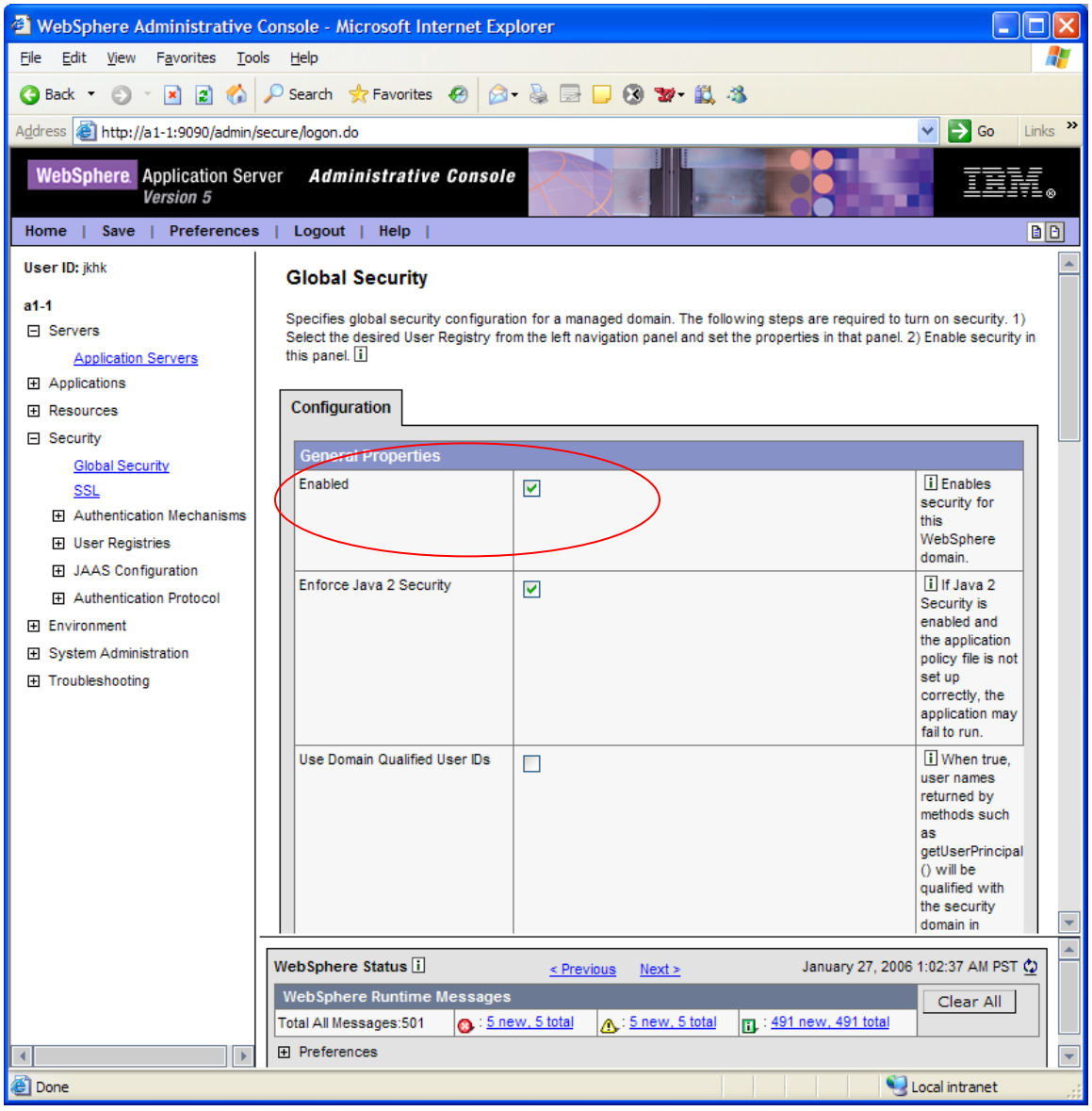
5 CAMM Configuration for WebSphere 5.1 Global Security

1) **Enable Global Security** in WebSphere 5.1 using the WebSphere admin console. This will enable all ports to WebSphere and the admin console to run with authentication and encryption.

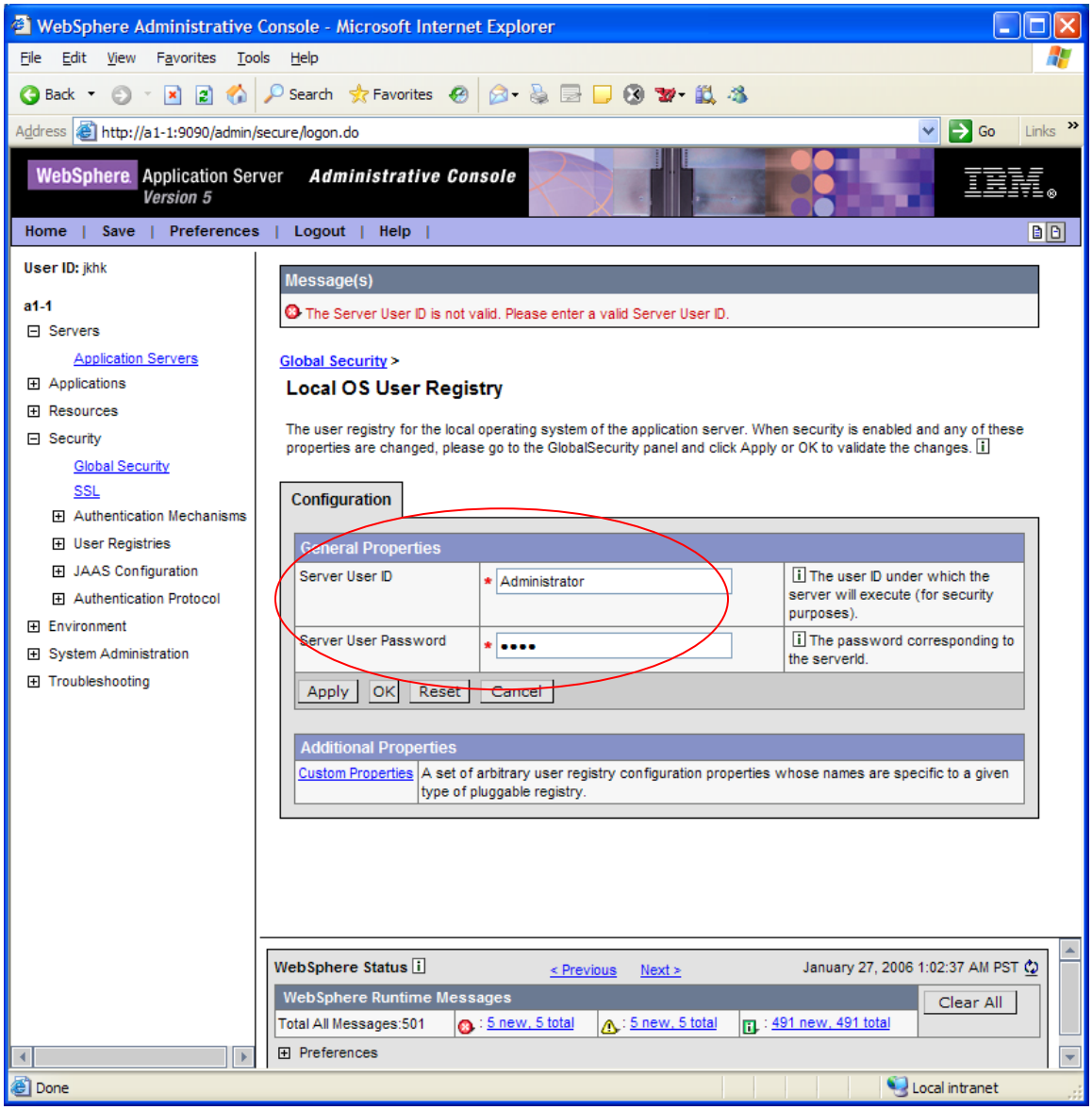
a) Navigate with left side menu tree: *Security -> Global Security*



b) Click on the Enabled checkbox to enable WebSphere global security for node.



c) It may prompt you for the OS root or administrator username/password, if these have not been updated before.



2) **Update server side WebSphere property** files in [WAS_HOME]/properties:

soap.client.props

```
com.ibm.SOAP.securityEnabled=true
com.ibm.SOAP.loginUserid=admin
com.ibm.SOAP.loginPassword=test

com.ibm.ssl.keyStore=[WAS_HOME]/AcseraAgent/config/AcseraJavaAgentKey.jks
com.ibm.ssl.keyStorePassword=acserajava

com.ibm.ssl.trustStore=[WAS_HOME]/AcseraAgent/config/AcseraJavaAgentTrust.jks
com.ibm.ssl.trustStorePassword=acserajava
```

sas.client.props

```
com.ibm.CORBA.securityEnabled=true

com.ibm.ssl.keyStoreType=JKS
com.ibm.ssl.keyStore=[WAS_HOME]/AcseraAgent/config/AcseraJavaAgentKey.jks
com.ibm.ssl.keyStorePassword=acserajava

com.ibm.ssl.trustStoreType=JKS
com.ibm.ssl.trustStore=[WAS_HOME]/AcseraAgent/config/AcseraJavaAgentTrust.jks
com.ibm.ssl.trustStorePassword=acserajava
```

server.policy

Add the following entries in server.policy:

```
// Allow the Acsera Agent all permissions
grant codeBase "file:${was.install.root}/AcseraAgent/lib/-" {
        permission java.security.AllPermission;
};

// Allow the Acsera Deployer EJBs all permissions
grant codeBase "file:${was.install.root}/installedApps/[node]/[Acsera app name].ear/-" {
        permission java.security.AllPermission;
};
```

Normally, using the websphereDeployer command, the CAMM deployer EJBs would be deployed in the WebSphere server environment with the application name of the form:

CAMM_<node name>_<server name>

For example, this is an application name of a *deployer* deployed on node *a6-7* and server *WebSphere_Portal*.

CAMM_a6-7_WebSphere_Portal

Configure CAMM for security:

3) Update configuration on CAMM side.

a) Update *$CAMM_HOME/config/Acsera.properties*:

The property values shown in the following are set by default. If a WebSphere installation is configured with a custom set of certificates, these properties will need to be updated. The path of the JKS files are specified as relative to the *$CAMM_HOME/config* folder.

```
// Allow the Acsera Agent all permissions
javax.net.ssl.keyStore=AcseraManagerKey.jks
javax.net.ssl.keyStorePassword=acseramanager
javax.net.ssl.keyStoreType=JKS
javax.net.ssl.trustStore=AcseraManagerTrust.jks
javax.net.ssl.trustStorePassword=acseramanager
javax.net.ssl.trustStoreType=JKS
```

b) Update *ACSERA_HOME/config/configuration.xml*:

The username and password parameters for the WebSphere resource need to be updated since these values will be used to authenticate the manager with WebSphere.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<resource name="a1-1" enabled="true">
        <resourceDefinition>WebSphere_5.1.0</resourceDefinition>
        <mipProfiles>
                <mipProfile enabled="true">
                        <mip>IBMJava</mip>
                </mipProfile>
                <mipProfile enabled="true">
                        <mip>IBMJMXMIP</mip>
                        <configParameters>
                                <configParameter>
                                        <key>username</key>
                                        <value>websphere</value>
                                </configParameter>
                                <configParameter>
                                        <key>password</key>
                                        <value>websphere</value>
                                </configParameter>
                                <configParameter>
                                        <key>protocol</key>
                                        <value>soap</value>
                                </configParameter>
                                <configParameter>
                                        <key>host</key>
                                        <value>a6-7</value>
                                </configParameter>
                                <configParameter>
                                        <key>port</key>
                                        <value>8881</value>
                                </configParameter>
                        </configParameters>
                </mipProfile>
        </mipProfiles>
</resource>
```