

# ZENworks. 2017 Update 3

## ZENworks Diagnostics and Probe Guide

August 2018

## Legal Notice

For information about legal notices, trademarks, disclaimers, warranties, export and other use restrictions, U.S. Government rights, patent policy, and FIPS compliance, see <https://www.novell.com/company/legal/>.

**Copyright © 2018 Micro Focus Software Inc. All rights reserved.**

---

# Contents

<b>About This Guide</b>	<b>5</b>
<b>1 ZENworks Diagnostics and Probe tool Overview</b>	<b>7</b>
1.1 ZENworks Diagnostics	7
1.1.1 ZENworks Databases	7
1.1.2 ZENworks Primary Servers	8
1.1.3 Deploy and undeploy probe	8
1.2 ZENworks Processes	8
1.3 ZENworks Probe	9
1.3.1 Applications	9
1.3.2 Threads	12
1.3.3 System	12
1.3.4 Connectors	15
<b>2 Troubleshooting ZENworks Primary Server using ZENworks Diagnostics and Probe</b>	<b>17</b>
2.1 High CPU Utilization	17
2.2 High Memory Utilization	17
2.3 Thread Utilization High	18
2.4 Database connectivity issue	18
<b>3 Common Procedures</b>	<b>19</b>
3.1 Launching Probe	19
3.2 Collecting Thread Dump	19
3.3 Collecting Heap Dump	19
3.4 Viewing the Request Pattern	20
3.5 Deploying and Removing Probe	20
3.6 Configuring the Health Metrics for Primary Servers	20
3.7 Enabling ZENworks Xplat Agent Service	21
3.8 Configuring Refresh Rate of Diagnostics Home Page	22
<b>4 Troubleshooting</b>	<b>23</b>



# About This Guide

The ZENworks Diagnostics and Probe Guide includes information to help you to use the Diagnostics and Probe tool. A new diagnostics dashboard and diagnostics tool is available for the ZENworks Primary Server. This allows the administrator to diagnose the state of Primary Servers. This uses the ZENworks Probe tool, an open source diagnostics toolkit, which is useful in monitoring and troubleshooting problems with Java processes.

- ♦ [Chapter 1, “ZENworks Diagnostics and Probe tool Overview,” on page 7](#)
- ♦ [Chapter 2, “Troubleshooting ZENworks Primary Server using ZENworks Diagnostics and Probe,” on page 17](#)
- ♦ [Chapter 3, “Common Procedures,” on page 19](#)

## Audience

This guide is intended for ZENworks administrators.

## Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the **comment on this topic** feature at the bottom of each page of the online documentation.

## Additional Documentation

ZENworks is supported by other documentation (in both PDF and HTML formats) that you can use to learn about and implement the product. For additional documentation, see the [ZENworks 2017 documentation website](#).



# 1 ZENworks Diagnostics and Probe tool Overview

The Primary Server is the central piece in the ZENworks Configuration Management architecture. The Primary Server is the source of getting information about the devices managed by the zone. Primary server is where an administrators performs all administrative tasks in the zone, through the ZENworks Control Center interface. The Primary Server is responsible for all the communication between the Database and the User Sources in the zone. Due to these responsibilities, the Primary Server can also become the single point of failure in the zone and therefore it is essential that we monitor the state of the ZENworks activity on the servers from time to time.

ZENworks Control Center has a diagnostics dashboard that allows the administrator to monitor and diagnose the state of Primary Servers. This uses the ZENworks Probe tool, an open source diagnostics toolkit, which is useful in monitoring and troubleshooting problems with Java process.

Using ZENworks Diagnostics, it is possible to use the ZENworks Control Center interface to inspect and debug the state of the ZENworks servers in the zone. ZENworks Diagnostics provides the ZENworks Administrator an intuitive portal to check the state of the LDAP sources (eDirectory or Active Directory) and also provides a probe feature wherein the different processes running on the ZENworks Server can further be analyzed or debug information could be collected very easily and provided to Novell Technical Services for analysis. Therefore, Diagnostics will help to narrow down on any specific issues within the ZENworks zone.

- ♦ [Section 1.1, “ZENworks Diagnostics,” on page 7](#)
- ♦ [Section 1.2, “ZENworks Processes,” on page 8](#)
- ♦ [Section 1.3, “ZENworks Probe,” on page 9](#)

## 1.1 ZENworks Diagnostics

The ZENworks Diagnostics helps the administrator to check and isolate the problems with the Primary servers.

- ♦ [Section 1.1.1, “ZENworks Databases,” on page 7](#)
- ♦ [Section 1.1.2, “ZENworks Primary Servers,” on page 8](#)
- ♦ [Section 1.1.3, “Deploy and undeploy probe,” on page 8](#)





### 1.1.1 ZENworks Databases

The ZENworks Databases panel displays the **Status**, **Database Size**, name of the **Host** database, **Type**, **Version**, and **Schema** of the databases present in the Zone.

The database schema can be ZENworks or Audit schema. Click the **Schema** hyperlink to open the **Table Record** panel with the **Table** and **Count** information.

## 1.1.2 ZENworks Primary Servers

The ZENworks Primary Servers panel displays the **Status**, **Name**, **Operating System**, **IP Address**, **Memory Used/Total(MB)**, **CPU Usage(%)**, **Time Sync**, and **User Source Connectivity** of the Primary Servers.

**Status:** The status of the Primary Server can be  **Normal**,  **Critical**,  **Warning** or  **Disconnected**. You can define the status of a Primary Server by configuring the health matrix. For steps to configure the health matrix refer to, [Section 3.6, "Configuring the Health Metrics for Primary Servers," on page 20.](#)

**Time Sync:** Indicates if the database and server time is synchronized.

The following tasks can be performed in this panel:

Link/Icon	Task
<b>Name</b> of the Primary Server	Click this link to open the ZENworks Process page.
<b>Pause Live Update</b> icon	Click this icon to pause the live updates on the ZENworks Primary Servers panel. To enable live update click <b>Enable Live Update</b> icon.

## 1.1.3 Deploy and undeploy probe

ZENworks Probe is available on all Primary Servers. The version of the ZENworks Probe that is running on the server is displayed on the Diagnostics page.

ZENworks Probe can be deployed or removed from the Diagnostics page. Click **Remove probe** to undeploy the probe. If Probe is not deployed, click **Deploy probe**. Select the appropriate `.war` file, then click **OK**.

### NOTE

- ◆ You can deploy the latest version of Probe by removing the existing ZENworks Probe deployment from the server. To obtain the latest version of the Probe `.war` file go to Novell download site. In case you want to redeploy probe, remove probe, extract the package.....
  - ◆ Extract the package `novell-zenworks-probe.msi` (Windows) / `novell-zenworks-probe.rpm` (Linux).
  - or
  - ◆ Copy the `.war` file from other Primary Servers.
- ◆ You will not need to restart the Server after deploying Probe.

## 1.2 ZENworks Processes

The ZENworks Processes page is displayed when you click any of the Primary Server names. This page provides information about the Java processes.

The Java Process List panel displays the following information:



**Process:** The list of processes. Click any of the following to open the ZENworks Probe page:

- ♦ ZENworks Server
- ♦ ZENworks Loader
- ♦ ZENworks Authentication Service
- ♦ ZENworks Join Proxy
- ♦ ZENworks Xplat Agent

---

**NOTE**

- ♦ ZENworks Xplat Agent is listed in this page if few pre-configurations are set. For more information, see the [Section 3.7, “Enabling ZENworks Xplat Agent Service,” on page 21](#).
- ♦ If ZENworks Control Center is launched using an IP address, it prompts for authentication before opening the ZENworks Probe page.

---

**Threads Used / Total:** The number of threads used by process compared to the maximum number of threads allowed by the thread pool in case of ZENworks Server and ZENworks Authentication Service. As ZENworks Loader and ZENworks Join Proxy do not use thread pools, this field shows the number of threads running. If multiple thread pools are configured, then this field shows the thread pool which is using maximum number of threads.

**Memory Used / Total (MB):** The amount of memory used by process compared to the maximum memory available.

**Database Connections Used / Total:** The database connections used by process compared to the maximum connections available.

---

**NOTE:** The total database connections is shown as not applicable for ZENworks Authentication Service and ZENworks Join Proxy services as they do not make connections to Database.

---

## 1.3 ZENworks Probe




ZENworks Probe is an open source diagnostics toolkit useful in monitoring Java applications. The ZENworks Probe page has the following tabs:

- ♦ [Section 1.3.1, “Applications,” on page 9](#)
- ♦ [Section 1.3.2, “Threads,” on page 12](#)
- ♦ [Section 1.3.3, “System,” on page 12](#)
- ♦ [Section 1.3.4, “Connectors,” on page 15](#)

### 1.3.1 Applications

The Application tab, provides information about each Web application deployed on the Tomcat server: list of Web application with total number of requests processed, whether it is running, total number of sessions, session timeout, and whether it is distributable.

The following tasks can be performed from this tab:

Icons / Link	Task
	Click this icon to undeploy the application.
Status	Click this link to stop or start an application.
	Click this icon to refresh the application status.
Request count	Click this link to open the Servlet page. This page provides information about the number of request per servlet.
	Click this icon to get statistics for all of the Web applications.

This tab lists the installed web applications for the Tomcat server. The **NAME**, **STATUS**, **DESCRIPTION**, **REQ**, **SESS TIMEOUT**, **CLSTRED.?**, and **SESS.** fields are displayed in the panel.

**NAME:** Specifies the application name. Click the application name to open the Application Summary page.

**STATUS:** Specifies the status of the application, running or down. Click the status to stop or start an application.

**REQ:** Specifies the number of requests for the application. Click **REQ** to sort the application by request count. This helps you see how loaded an application is.

**CLSTRED.?:** Specifies whether the application is distributable.

**SESS.:** Specifies the session count.

This page provides the following information:




- ◆ [“Application Summary” on page 10](#)
- ◆ [“Application Deployment Descriptor” on page 11](#)
- ◆ [“Application Servlet” on page 11](#)

## Application Summary

This page gives you the information that is specific to the selected application: Doc.Base, Servlet version, Number of servlets present for that application, session timeout, and whether it is serial or not.

The graphical representation of **NUMBER OF REQUESTS** and **AVERAGE RESPONSE TIME** helps users to determine the load for that application at any given time.

The following tasks can be performed by clicking on the icons:

Icon	Description
	Click this icon to stop the application.
	Click this icon to reload the application.
	Click this icon to undeploy the application.  <b>NOTE:</b> If an application is undeployed, it cannot be deployed through probe again. However, it can be deployed by copying the webapp from another server or by locating the package that provides the webapp, extracting the <code>.war</code> file, and then deploying it.

## Application Information

The Application Information panel displays the **Application name**, **Doc.base**, **Description**, **Servlet version**, **Servlet count**, **Session timeout**, and **Clustered** application information.

**Doc.base:** The document base directory (known as Context Root) for this web application, or the path name to the web application archive file (if executed directly from the WAR file).

**Servlet count:** The number of servlets available for this application. Click the servlet count value to open the Servlets page.

## Statistics Charts


This panel displays the following statistics charts for the application:

**NUMBER OF REQUESTS:** A chart with coordinates corresponding to the REQUEST COUNT (Total number of requests for that application) and ERROR COUNT (Number of errors).

**AVERAGE RESPONSE TIME (MS):** A chart with coordinates corresponding to PROCESSING TIME (Processing time for the request), MIN TIME (Minimum time), MAX TIME (Maximum time), and AVG RESPONSE TIME (Average response time).

## Application Deployment Descriptor

This page displays the `web.xml` of the application. This contains important information such as url-mapping, which helps you understand which servlet will be called for a specific type of URL.

Click  **download** to download the XML file.

## Application Servlet

The Application Servlet page displays the **NAME**, **AVAIL**, **STARTUP**, **LOAD TIME**, **REQ.**, **PROC TIME**, **ERR**, **MIN TIME**, **MAX TIME**, **MULTI THRD**, **Servlet mappings** and **Show All** servlet information.

**AVAIL:** Whether the servlet is available.

**STARTUP:** The sequence in which the servlet is loaded at servlet container startup.

**REQ.:** The number of requests processed by each servlet.

**PROC TIME:** The time taken by the servlet to process all the request received.

**ERR:** The error count.

**MIN TIME:** The minimum time taken to process the request.

**MAX TIME:** The maximum time taken to process the request.

**MULTI THRD:** Whether the servlet is multithreaded.

**Servlet mappings:** Opens the Servlet mapping page.

**Show All:** Opens the Servlet page, which lists the information for all servlets on the Tomcat server.

## Servlet Mappings

The Servlet Mapping page displays **URL**, **SERVLET NAME**, **SERVLET CLASS**, and **AVAIL** mapping between the servlets and the URLs.

**AVAIL:** Whether the servlet is available.

## 1.3.2 Threads

The Threads tab displays the **ID**, **NAME**, **EXEX. POINT**, **STATE**, **IN.NATIVE**, **SUSP.**, **WC**, **BC**, **Thread Pool**, and **Dump All Threads** information about the threads running on the current Java Virtual Machine (JVM).

**EXEX. POINT:** The current execution point for the thread.

**IN.NATIVE:** Whether the thread is executing native code.

**SUSP.:** Whether the thread is suspended.

**WC:** The total number of times the thread was in wait notification state. That is, the number of times a thread has been in `java.lang.Thread.State.WAITING` or `java.lang.Thread.State.TIMED_WAITING` state.

**BC:** The total number of times the thread was blocked. That is, the number of times a thread has been in the `java.lang.Thread.State.BLOCKED` state.

**Threads Pool:** Click this link to open the Threads Pools page.

**Dump All Threads:** Click this link to access the thread dump of the system in its present state. This can be saved on the local machine and opened in any thread dump analyzer.

## Threads Pools

The Threads Pools page displays the **NAME**, **CURRENT THREAD COUNT**, **CURRENT THREADS BUSY**, **MAX THREADS**, **MAX SPARE THREADS**, and **MIN SPARE THREADS** thread pool information for the Tomcat process.

## 1.3.3 System

This page provides links to the following system information:

**Overview:** Open the System Information page.

**Memory utilization:** Open the Memory Utilization page.

**System properties:** Open the System Properties page.

**OS information:** Open the OS Information page.

- ◆ [“System Overview” on page 13](#)
- ◆ [“Memory Utilization” on page 13](#)
- ◆ [“System Properties” on page 14](#)
- ◆ [“OS Information” on page 14](#)

## System Overview

This page provides information about memory utilization, operating system, and the Tomcat container.

### Memory Utilization Panel

This panel displays the **CURRENT MEMORY USAGE IS, FREE, TOTAL, MAX** and **Advice GC** and **Dump Heap** memory utilization information.

**TOTAL:** The total memory space.

**MAX:** The maximum memory space available.

**Advice GC:** Click this link to advise the JVM to perform garbage collection.

**Dump Heap:** Click this link to generate the memory dump of the JVM. This is stored in the machine where the target JVM is running.

### OS Information Panel

This panel displays the **JVM, OS, PROCESSORS, CURRENT TIME,** and **WORKING DIR** OS information.

**JVM:** The Java platform and version.

**OS:** The operating system name and version.

**PROCESSORS:** The number of processors.

**WORKING DIR:** The working directory for the Tomcat server.

### Container information Panel

This panel displays the **CONTAINER, CATALINA.BASE, CATALINA.HOME, APPLICATION BASE,** and **CONFIGURATION BASE** Tomcat container information.

## Memory Utilization

This page displays the memory utilization for Java objects in different generations and based on the usage, can even advise for garbage collection. This page has the following panels:

### CURRENT MEMORY USAGE:

- ◆ **NAME:** The name of the pool.
- ◆ **USAGE SCORE:** The usage scope bar chart.
- ◆ **PLOT:** Hides or unhides the memory usage graph.
- ◆ **USED:** The amount of memory currently used, including the memory occupied by all objects, both reachable and unreachable.

- ♦ **COMMITTED:** The amount of memory guaranteed to be available for use by the JVM. The amount of committed memory may change over time. For example, the Java virtual machine might release memory to the system, so the amount of committed memory could be less than the amount of memory initially allocated at start up. The amount of committed memory will always be greater than or equal to the amount of used memory.
- ♦ **MAXIMUM:** The maximum amount of memory that can be used for memory management. Its value could change or be undefined. A memory allocation could fail if the Java VM attempts to increase the used memory to an amount greater than committed memory, even if the amount used is less than or equal to maximum value (for example, when the system is low on virtual memory).
- ♦ **INITIAL:** The initial memory allocated.
- ♦ **TOTAL:** The total memory allocated.

#### **MEMORY USAGE HISTORY:**

- ♦ **Permanent Generation (non-heap):** The pool containing all of the reflective data on the virtual machine itself, such as class and method objects. With Java VMs that use class data sharing, this generation is divided into read-only and read-write areas.
- ♦ **Tenured Generation (heap):** The pool containing objects that have existed for some time in the survivor space.
- ♦ **Survivor Space (heap):** The pool containing objects that have survived the garbage collection of the Eden space.
- ♦ **Code Cache (non-heap):** The HotSpot JVM also includes a code cache, containing memory that is used for the compilation and storage of native code.
- ♦ **Eden Space (heap):** The pool from which memory is initially allocated for most objects.

## **System Properties**

This page displays the following Java system properties information:

**PROPERTY NAME:** The system property name.

**PROPERTY VALUE:** The property value.

## **OS Information**

This page displays the following operating system information:

**OS Information:** This panel displays the OS name, Version, Total RAM, Free RAM, Committed JVM memory (the amount of virtual memory guaranteed to be available to the running process), Total swap, and Free swap.

**HISTORICAL CHARTS:** This panel displays the JVM CPU UTILIZATION (%) chart, OS & JVM MEMORY USAGE(KB) chart, and SWAP USAGE(KB) chart.

## 1.3.4 Connectors

This page provides the list of connectors and its information for Tomcat servers. The charts in this tab helps users to find out the number of incoming requests for each connector and how much time it took to process at each interval and the amount of data that got transferred at each interval. The user can find the remote IP which is requesting for a particular URL and the time it took for processing that request.

The following chart is displayed for each connector type:

**NUMBER OF REQUESTS IN EACH INTERVAL:** The number of requests in each interval.

**PROCESSING TIME (MS) IN EACH INTERVAL:** The processing time spent for the requests in each interval.

**TRAFFIC VOLUME (BYTES) IN EACH INTERVAL:** The traffic volume in each interval.

**Other information:** The remote IP of the request, the stage of the request, processing time for the request, and the originating URL of the request has come.





# 2 Troubleshooting ZENworks Primary Server using ZENworks Diagnostics and Probe

The few scenarios in which administrator can use ZENworks Diagnostics and Probe are:

- ♦ [Section 2.1, “High CPU Utilization,” on page 17](#)
- ♦ [Section 2.2, “High Memory Utilization,” on page 17](#)
- ♦ [Section 2.3, “Thread Utilization High,” on page 18](#)
- ♦ [Section 2.4, “Database connectivity issue,” on page 18](#)

## 2.1 High CPU Utilization

If Administrator notices that there is High CPU Utilization on one of the Primary Server.

Then he has to follow the below procedure to rectify this issue:

- 1 Identify the process which is causing high CPU utilization on that Primary Server.
- 2 If it is any of the ZENworks processes listed in the Java process list then, [Launch Probe](#) for that process.
- 3 Check for the thread status.
- 4 Take [Thread Dumps](#) at regular intervals. This thread dump will help analyze which thread is the cause for high utilization of CPU.
- 5 Send the collected thread dumps to Novell Technical Support for further analysis.

## 2.2 High Memory Utilization

If Administrator notices that there is High Memory Utilization on one of the Primary Servers.

Then he has to follow the below procedure to rectify this issue:

- 1 Click the Primary Server link to open the ZENworks Process page.
- 2 Check if any of the ZENworks process is utilizing more memory.
- 3 If yes, then [Launch Probe](#) for the specific process.
- 4 In System tab, Memory Utilization tab provided information on which pool of the heap memory is utilized more. This information can be used to configure the heap memory.
- 5 Take [Heap Dump](#) at regular intervals to analyze performance or memory leak issues.
- 6 Send the collected heap dumps to Novell Technical Support for further analysis.

## 2.3 Thread Utilization High

If **Thread Used/Total** for a java process is critical then, [Launch Probe](#). Open Application tab. This provides information on which application is [servicing most number of the requests](#) and which application is taking more time to respond. If the issue is consistent then contact Novell Technical Support.

## 2.4 Database connectivity issue

If **Database Connection Used/Total** for a java process is critical then, investigate why there are so many connections to database. Clean up orphaned connections or long pending transactions will help resolve this issue or contact Database Administrator (DBA) or Novell Technical Support.

# 3 Common Procedures

- ◆ Section 3.1, “Launching Probe,” on page 19
- ◆ Section 3.2, “Collecting Thread Dump,” on page 19
- ◆ Section 3.3, “Collecting Heap Dump,” on page 19
- ◆ Section 3.4, “Viewing the Request Pattern,” on page 20
- ◆ Section 3.5, “Deploying and Removing Probe,” on page 20
- ◆ Section 3.6, “Configuring the Health Metrics for Primary Servers,” on page 20
- ◆ Section 3.7, “Enabling ZENworks Xplat Agent Service,” on page 21
- ◆ Section 3.8, “Configuring Refresh Rate of Diagnostics Home Page,” on page 22

## 3.1 Launching Probe

- 1 In ZENworks Control Center, click the **Diagnostics** tab.
- 2 In the Diagnostics page, click the name of the primary server to open the ZENworks Process page.
- 3 In the ZENworks Processes page, click on the process name link to launch probe for that process.

---

**NOTE:** If ZENworks Control Center is launched using an IP address, it prompts for authentication before opening the Probe page.

---

## 3.2 Collecting Thread Dump

- 1 In the Probe page, click **Threads** tab.
- 2 Click **Dump All Threads**.

The thread dump file can be saved on the local machine and opened in any thread dump analyzer.

## 3.3 Collecting Heap Dump

- 1 In the Probe page, click **System** tab.
- 2 Click **Memory Utilization** tab.
- 3 Click **Dump Heap**.

Click this link to generate the memory dump of the JVM. This is stored in the machine where the target JVM is running.

## 3.4 Viewing the Request Pattern

To view the request pattern in the **Applications** tab:

- 1 Click **Application Statistics** tab to view the graphical representation of the requests. The Avg Response time and number of request should help user analyze the request pattern. If Avg response time is high it has to be investigated further.
- 2 Sort the applications by Req number to check which application has the maximum request.

**To view the request pattern in the Connectors tab:** In **Connectors** tab, we can find the url which is servicing the request at present.

**To view the request pattern in the Threads tab:** Take **Thread Dump**, at regular interval and check which thread is running for a long time or is it waiting on Database or LDAP for long duration. If Database response is slow then, check the number of connections to Database in Diagnostics page. If the status is normal, we may have to find long running queries in Database and analyze Database performance.

## 3.5 Deploying and Removing Probe

Probe can be deployed or removed from the Diagnostics page in ZENworks Control Center.

To Deploy probe:

- 1 If Probe is not deployed, click **Deploy probe**.
- 2 Select the appropriate `.war` file, then click **OK**.

---

### NOTE

- ♦ To obtain the latest version of the Probe `.war` file:
    - ♦ Extract the package (Windows) / (Linux).
    - ♦ Copy the `.war` file from other Primary Servers.
  - ♦ You will not need to restart the Server after deploying Probe.
- 

To Remove probe:

- ♦ Click **Remove probe** to undeploy the probe.

## 3.6 Configuring the Health Metrics for Primary Servers

This is a Server specific configuration.

- 1 After installing the Primary Server, open the following file:

- ♦ On Windows: `ZENWORKS_HOME\share\tomcat\webapps\zenworks\WEB-INF\conf\diagnostics.conf`

- ♦ On Linux: `/opt/novell/zenworks/share/tomcat/webapps/zenworks/WEB-INF/conf/diagnostics.conf`

Ensure that the `zenworks.diag.processor.metrics` attribute is not commented.

---

**NOTE:** On a Windows Server, ensure that `zenworks.diag.processor.metrics` is assigned the path `C:/Program Files(x86)/Novell/ZENworks/share/tomcat/webapps/zenworks/WEB-INF/conf/HealthMetrics.xml`

---

2 Open the following file:

- ◆ On Windows: `ZENWORKS_HOME\share\tomcat\webapps\zenworks\WEB-INF\conf\HealthMetrics.xml`

- ◆ On Linux: `/opt/novell/zenworks/share/tomcat/webapps/zenworks/WEB-INF/conf/HealthMetrics.xml`

---

**NOTE:** The health of a Primary Server is computed based on health attributes such as: `RAM_USAGE`, `CPU_USAGE`, `TIME_SYNC`, `LDAP_CONNECTIVITY`, `MEMORY_USAGE`, `THREAD_COUNT_PERCENTAGE`, `DB_CONNECTIONS`, and `THREAD_COUNT`.

---

3 Modify the `Min` and `Max` parameter for each `HealthAttribute` to the desired value using a text editor then save the file.

---

**NOTE:** If the health attribute value is within the specified `Min` and `Max` range, the status of the Primary Server is Good. Primary Server status can be Good, Critical, or Warning.

---

4 Restart the ZENworks server service on the Primary Server.

5 Launch ZENworks Control Center then click **Diagnostics** on the left panel.

6 Check the status of each attribute of the Primary Server in the **ZENworks Primary Server** panel.

---

**NOTE:** The icon indicating the status of the Primary Server based on the values provided in the `HealthMetrics.xml` file. The health of the Primary Server is computed based on these attributes and the attributes of the Java processes.

---

7 Click the Primary Server hyperlink. The Service page is displayed listing the ZENworks Java processes running on that server and their attributes.

8 Click the Process hyperlink to launch the Probe page.

---

**NOTE:** You can right-click the Process hyperlink and then select Open Link in a new tab to open Probe for different processes simultaneously.

---

## 3.7 Enabling ZENworks Xplat Agent Service

By default the Xplat agent service for Linux will not be available in the list of services. To enable the service:

- 1 Open the Host file. Change the hostname associated with 127.0.0.1 to the hostname that is present on the certificate for the Linux server.

- 2 Open the file `/etc/init.d/novell-zenworks-xplatcmd`. Change the value associated with `Djava.rmi.server.hostname` to hostname that is present on the certificate for the Linux server.

- 3 Restart the `xplat-zmd` service with `/etc/init.d/novell-zenworks-xplatcmd restart`.

Incase there are multiple Linux servers in the Zone, ensure to change the hostname in all the servers, repeat [Step 1](#) through [Step 3](#).

- 4 Set `checkXplatAgent=true` in the `diagnostics.conf` file located in `ZENWORKS_HOME/share/tomcat/webapps/zenworks/WEB-INF/conf` on the server from where ZENworks Control Center is launched.

## 3.8 Configuring Refresh Rate of Diagnostics Home Page

This is a Server specific configuration.

- 1 Open the `diagnostics.conf` file from the path:

**Window:** `/zenworks_home/share/tomcat/webapps/zenworks/web-inf/conf`

**Linux:** `/opt/novell/zenworks/share/tomcat/webapps/zenworks/WEB-INF/conf`

- 2 Set the property, `refreshRate` to the desired interval.
- 3 Restart the server.

# 4 Troubleshooting

The following sections explain the scenarios that you might encounter while using the ZENworks Diagnostic tool of ZENworks:

- ♦ “An authentication error is displayed when you run the Diagnostics tool to check the status of the ZENserver process” on page 23

## An authentication error is displayed when you run the Diagnostics tool to check the status of the ZENserver process

Source: ZENworks 2017; ZENworks Configuration Management; Diagnostics.

Explanation: When you run the diagnostics tool to check the status of the ZENserver process, an authentication error occurs. This issue occurs because instead of the hostname, the IP address is included in the `openid-provider.properties` file.

Action: Navigate to the `openid-provider.properties` file and replace the IP address with the hostname and then restart the tomcat server. The `openid-provider.properties` file can be accessed from the following location:

- ♦ **On Windows:** `%ZENWORKS_HOME%\conf`
- ♦ **On Linux:** `/etc/opt/novell/zenworks/`

