Orasi AppDynamics LoadRunner Performance Extension Installation

Please note that these instructions are under development. Please inform [***Jon.fowler@orasi.com***](mailto:Jon.fowler@orasi.com) and [***David.Guimbellot@orasi.com***](mailto:David.Guimbellot@orasi.com) if you have any suggestions for improvement.

# Installation of Lr2appd widget on LR

In this step you will add binaries onto the systems which will run the scripts for load generation. These binaries will enable the scripts to communicate with the Windows OS native performance monitoring on the local computer.

1. Gain administrator access to install binaries on each of the LoadRunner machines which will run the scripts. Note that is VuGen for testing and a Load Generator for execution during run time.
2. Download the Orasi AppDynamics LoadRunner Performance Extension from xxxx.

## Orasi Performance Utility Installation Instructions

There are two DLL’s required as part of the installation process. The both need to be in the %LG\_PATH% directory in these specific locations:

### OrasiAppDynamicsLoadToolsExtension.dll

Copy OrasiAppDynamicsLoadToolsExtension.dll to the %LG\_PATH%\bin directory

Example: VuGen or Load Generator

* xcopy OrasiAppDynamicsLoadToolsExtension.dll"%LG\_PATH%\bin\"

### OrasiPerformanceCounterUtility.dll

Copy OrasiPerformanceCounterUtility.dll to the %LG\_PATH%\bin\OrasiPerformanceCounterUtility directory

Example: VuGen or Load Generator

* xcopy OrasiAppDynamicsLoadToolsExtension.dll"%LG\_PATH%\bin\OrasiPerformanceCounterUtility\"

### Dependencies

The DLL requires the Visual C++ Redistributable Packages for Visual Studio 2013

* <http://download.microsoft.com/download/2/E/6/2E61CFA4-993B-4DD4-91DA-3737CD5CD6E3/vcredist_x64.exe>
* <http://download.microsoft.com/download/2/E/6/2E61CFA4-993B-4DD4-91DA-3737CD5CD6E3/vcredist_x32.exe>

# LR Script Update and Validation

In these steps we will modify the VuGen scripts and validate them locally. Note that updating a Windows performance counter is very fast and will have no measurable impact on script execution.

## Script Updates

In this step we will modify the LR scripts to increment the counter associated with each transaction.

* Add the code to your LoadRunner VuGen scripts in the format of LoadRunner(MyTransactionName)

### Example LoadRunner VuGen code

vuser\_init()

{

       int retval = 0;

       retval = lr\_load\_dll("OrasiAppDynamicsLoadToolsExtension.dll");

       IncrementCounter("LoadRunner(VUsers)", 1);

       return 0;

}

Action()

{

       lr\_start\_transaction("MyTransactionName");

       web\_service\_call( "StepName=MyTransactionName",

           ...

       );

       lr\_end\_transaction("MyTransactionName", LR\_AUTO);

       //Pass in the Category and instance.

//Count and Rate/Sec is then recorded in Perfmon

       IncrementCounter("LoadRunner(MyTransactionName)", 1);

       return 0;

}

vuser\_end()

{

       IncrementCounter("LoadRunner(VUsers)", -1);

       return 0;

}

## LR script Validation process

The goal of this step is to validate that Windows Performance Counters and VuGen agree on the transaction counts.

1. Run your script as you would normally for a short duration.
2. Within LR validate the count of transactions for each of the scripts that you modified
3. Using the Windows Performance Counter tool, which you can invoke via perfmon.exe or using the perfmon.msc
4. Add the performance counters for each of the transactions you defined.
   1. Add the performance counters LoadRunner(MyTransactionName)\Rate/Sec
   2. Add the LoadRunner(MyTransactionName)\Count
5. Validate that the counters rate and total align.

### Troubleshooting Tips

* Make sure you get it to work in a single execution of Vugen before deploying to Load Generators
* After creating a controller scenario, run it against the local load generator before running against load generators on other machines as it is easier to debug issues on a single machine.
* Double check the directories in the installation instructions
* Verify everything is in the %LG\_PATH% directory as specified in the installation instructions above.
* You can use the Windows Dependency Walker (<http://www.dependencywalker.com/>) to check if any dependent DLL’s are not available.
* It is common to see GPSVC.DLL, DBGHELP.DLL, and SHDOCVW.DLL listed as missing dependencies in Windows Dependency Walker
* If you are getting an error from the Controller only, make sure that you copy the DLL to the same directory of all the Load Generator machines.
* If you are loading the DLL from the script’s directory, you could put the DLL in the Controller’s Design View -> Details -> More -> Files. However, I have not needed to do this.
* If you are still having problems, try to put your DLL into a path that is specified in the PATH environment variable of the machine. However, I have not needed to do this.

## AppDynamics Setup

In this step we will update the AppDynamics server to recognize our new counters and we will also create a basic dashboard to show the LR results within the AppDynamics UI.

### AppDynamics Machine Agent Setup

This step will install the machine agent with the controller.

1. Install the AppDynamics .Net Machine Agent by using the dotNetAgentSetup64.msi
   1. <https://download.appdynamics.com/onpremise/public/latest/dotNetAgentSetup64.msi>
2. Run the Agent Wizard to set the AppDynamics Controller name

### AppDynamics Counter Configuration Update

This step will update the central configuration file for AppDynamics controller to recognize our new counters. It can be done with the help of a tool, or by manually updating the config.xml file.

1. Use the tool named Windows Performance Counter - Configuration Extension
   1. <http://community.appdynamics.com/t5/eXchange-Community-AppDynamics/Windows-Performance-Counter-Configuration-Extension/idi-p/9713>
2. Modify the Config.xml manually
   1. Find %ProgramData%\AppDynamics\DotNetAgent\Config\config.xml
   2. Edit the file to upload the appropriate counters to the AppDynamics Controller
   3. Notepad %ProgramData%\AppDynamics\DotNetAgent\Config\config.xml

  <machine-agent>

    <perf-counters>

      <perf-counter cat="LoadRunner" name="Rate/Sec" instance="MyTransactionName" />

     <perf-counter cat="LoadRunner" name="Count"    instance="MyTransactionName" />

    </perf-counters>

  </machine-agent>

1. In services.msc, restart the AppDynamics.Agent.Coordinator service

### Validate the Counters in Machine Agent UI

Follow these steps to insure proper correlation of AppDynamics and the data gathered from each of the machines running VuGen.

1. Start the scripts for all of the systems running VuGen that have been updated.
2. View the results in AppDynamics Metric Browser
   * Analyze
   * Metric Browser
   * Select a Metric:
     + Application Infrastructure Performance
     + Machine Agent
     + Individual Nodes
     + <My Node Name>
     + Custom Metrics
     + Performance Monitor
     + LoadRunner
     + Rate/Sec
     + <MyTransactionName>
   * Double-click to add to graph

### Create an AppDynamics Custom Dashboard

* Custom Dashboards
* Create Dashboard
* Add Metric Data Graph
* Add Data Series
* Select Application:
* <Application Name>
* Select a Metric Category:
* Custom (use any metrics)
* Select a Metric:
  + Application Infrastructure Performance
  + Machine Agent
  + Individual Nodes
  + <My Node Name>
  + Custom Metrics
  + Performance Monitor
  + <My Category Name>
  + Rate/Sec
  + <MyTransactionName>

# Appendix: Validated Platforms of HP LR

Orasi AppDynamics LoadRunner Performance Extension has been verified to work on the following versions of HP LoadRunner

  - 11.50