Orleans Remote Deployment

Table of Contents

[Overview: 1](#_Toc310499760)

[Prerequisites 2](#_Toc310499761)

[Deployment Steps 2](#_Toc310499762)

[Orleans Deployment Manifest 3](#_Toc310499763)

[Orleans Silo Configuration 4](#_Toc310499764)

[Gateway Load Shedding 5](#_Toc310499765)

[Orleans Powershell Scripts 6](#_Toc310499766)

[Deploying Orleans using Powershell Script 6](#_Toc310499767)

[Confirming Orleans Status 7](#_Toc310499768)

[Monitoring Orleans 7](#_Toc310499769)

[Gathering Orleans Log Files 8](#_Toc310499770)

[Removing Orleans 8](#_Toc310499771)

# Overview:

Orleans based services are xcopy deployable. All you need to do to deploy an Orleans based service to a set of machines is copy the following set of file to the target machines and start the OrleansHost.exe host process:

* Content of [SDK-Root]\Binaries\OrleansServer folder
* OrleansConfiguration.xml file with configuration for the deployment to replace the default placeholder from [SDK-Root]\Binaries\OrleansServer
* Binaries with grain interfaces and grain implementation classes of the service along with any external dependencies to Application\<service name> subdirectory of the folder on the target with binaries from [SDK-Root]\Binaries\OrleansServer

This simple task may be accomplished in many different ways and with different tools, such as Autopilot. The SDK includes a set of PowerShell scripts that provide a way to deploy an Orleans based service to a cluster of machines and remote start the hosting processes on them. There are also scripts for un-deploying a service, monitoring its performance, start and stop it, and collect logs. These are the scripts we found useful for ourselves while building and testing Orleans.

# Prerequisites

The following table lists the prerequisites for deploying and running Orleans on a remote machine:

|  |  |
| --- | --- |
| Prerequisite | Details |
| .Net Framework 4.0 | Orleans runs under the .Net Framework 4.0, which can be installed from this link: <http://www.microsoft.com/downloads/details.aspx?FamilyID=9cfb2d51-5ff4-4491-b0e5-b386f32c0992&displaylang=en> |
| Powershell 2.0 with WinRM | Windows 7 and Windows Server 2008 R2 should have these installed by default.  For other versions of windows, select the appropriate download from this article: <http://support.microsoft.com/kb/968929>  To confirm you are running the required version of PowerShell, start a PowerShell window and type Get-Host – the resulting output needs to say “Version: 2.0” |
| WinRM Configuration | Both the source and target machines must be configured for remote operations: Open a PowerShell window as an Administrator and run the command below on the target machine (enter ‘y’ at the prompts):  winrm quickconfig |
| Increase PowerShell job memory | Set the memory limit on remotely invoked jobs to 4Gb:  Set-Item wsman:localhost\Shell\MaxMemoryPerShellMB 4096  To change it on a remote machine, use the following steps:   * Connect-WSMan -ComputerName <string> * Set-Item wsman:*<computerName>*\Shell\MaxMemoryPerShellMB * Disconnect-WSMan –ComputerName <string> |
| PowerShell Execution Policy set to run remote scripts | Open a PowerShell window as an Administrator and run the command below on the target machine(enter ‘y’ at the prompt):  Set-ExecutionPolicy RemoteSigned  This will set the machine to require signing for remote scripts only. |

# Deployment Steps

Note that the user running the scripts must be a member of the Administrators Group on the remote machines.

The basic steps are as follows:

1. Setup a deployment manifest (Deployment.xml).
2. Adjust the Orleans Configuration (OrleansConfiguration.xml) to suit the environment.
3. Run the PowerShell deployment scripts to deploy Orleans into your remote environment.

## Orleans Deployment Manifest

The Orleans Deployment scripts use a manifest (XML) file to specify details of the deployment, including source and destination locations and local or remote machines to be deployed to.

By making small changes to an existing deployment manifest xml file (typically by listing the different host machines), the same PowerShell scripts can deploy and run that Orleans system on a set of remote machines with equal ease as deploying and running that system on the local machine.

The default file name for the manifest is Deployment.xml, and if you just modify this file, it will not be necessary to specify a different name. There are times, such as during testing, it may be advantageous to maintain multiple deployment files that specify a different set of silos. These other files may be specified explicitly to the deployment tools as specified in the respective sections below.

A Deployment manifest contains many different items, which collectively allow deployment of the Orleans runtime and applications onto a variety of local and remote configurations:

* Source location for the **Orleans system runtime**
  + Located in the “Path” attribute of the <Deployment><Packages><Package> element where the “Type” attribute is set to “System”.
  + Typically: [ORLEANS-SDK]\Binaries\OrleansServer.
  + Example: <Package Name="Orleans Runtime" Type="System" Path="." />
* Source location for any additional **Orleans applications / grains** to be included in this Orleans system
  + Also located in the <Deployment><Packages><Package> nodes.
  + The “Type” attribute *must be set to “Application”.*
  + Use the “Filter” attribute to constrain the files deployed
  + <Package Name="Chirper" Type="Application" Path="..\Binaries\Applications\Chirper" />
* Source location for the **server configuration file** to be used by the Orleans host process
  + Located in the “Path” attribute <Deployment><RuntimeConfiguration> element.
  + The file name must be *OrleansConfiguration.xml* – if necessary, just change the path.
  + Example: <RuntimeConfiguration Path=".\OrleansConfiguration.xml" />
* **Target location** to install the Orleans server-side binaries on each machine.  
  Typically: C:\Orleans (for test or production nodes) or [ORLEANS-SDK]\LocalSilo (for the local development silo).
  + Located in the <Deployment><TargetLocation> element.
  + *Must be an absolute path* (i.e. no “..” locations).
  + Example: <TargetLocation Path="C:\Orleans" />
* The set of **silos** (host machines and optional silo names) this Orleans system should to be deployed to.
  + Located in the <Deployment><Nodes><Node> elements.
  + Typically: **Primary** on localhost, or multiple machines with one silo each.
  + The “HostName” attribute specifies the machine name.
  + The “NodeName” attribute specifies the name of the silo. Generally, this is arbitrary, with the exception that *if multiple silos will run on any one machine, then silo names must be unique*.
  + Example:   
    <Nodes>  
     <Node HostName="MACHINE1" NodeName="Primary" />  
     <Node HostName="MACHINE2" NodeName="Node2" />  
     <Node HostName="MACHINE3" NodeName="Node3" />  
    <Nodes />
* **Deployment Group ID** – this is a GUID which distinguishes one Orleans runtime system from another, even if both Orleans systems are running on the same machines.
  + Located in the <Deployment> element.
  + Example:  
    <Deployment   
     Name="Deployment1"   
     DeploymentGroup="F219832A-1EE1-45DA-B35D-0BB3060C9FDA"   
     xmlns="urn:xcg-deployment">

## Orleans Silo Configuration

Refer to the configuration guide - Orleans-Configuration.docx - for information on how to configure silos.

# Orleans Powershell Scripts

The following sections detail the PowerShell scripts provided with Orleans to aid with deployment and monitoring. (Use the /? option to get the latest usage info directly from the scripts.)

|  |  |  |
| --- | --- | --- |
| Script Name | Parameters | Description |
| DeployOrleansSilos.ps1 | [$deploymentConfigFile] | Copies the Orleans files to machines specified in the deploymentConfigFile (default is Deployment.xml). |
| UndeployOrleansSilos.ps1 | [$deploymentConfigFile] | Stops and removes Orleans from the deployment servers deploymentConfigFile (default is Deployment.xml). |
| MonitorOrleansSilos.ps1 | [$deploymentConfigFile]  [$networkInstance] [$samplesToLog] [$headerInterval]  [$repeatHeaderInFile | Monitors CPU, Memory, Network Send, and Network Receive performance counters, and logs the data to files both as an aggregate of all data, and in separate files for each server. See usage text for details about the parameters. |
| ShowOrleansSilos.ps1 | [$deploymentConfigFile] | Does a quick survey of the deployment silos and reports if Orleans is running on them. |
| GatherOrleansSiloLogs.ps1 | [$deploymentConfigFile]  [$outputPath] | Retrieve all log files from deployment silos and stores them in the specified output folder. |
| UtilityFunctions.ps1 | none | Provides ancillary functionality to the other scripts. |

## Deploying Orleans using Powershell Script

Start a separate PowerShell command window as an administrator.

Execute the **DeployOrleansSilos.ps1** script, providing the location of the deployment configuration file (deployment.xml is the default and will be used if you don’t supply a value).  
Examples: .\DeployOrleansSilos.ps1   
 .\DeployOrleansSilos.ps1 C:\Orleans\MyDeploy.xml

The deployment will execute the following steps:

* 1. Stop any running instances of Orleans that are running on the deployment machines.
  2. Copy the Orleans files and any application files that are listed in the deployment manifest.
  3. When the copy is completed, start the silos. This will pause after starting the first silo so that it is available for the other silos to register with.
  4. Pause to allow the start-up to complete.
  5. Report the progress of the deployment.

When the deployment is complete, Orleans is ready for clients to connect to it.

## Confirming Orleans Status

To determine if Orleans is running on the servers in the deployment manifest, run the **ShowOrleansSilos.ps1** script.

If you have used a deployment manifest file named something other than the default, specify it on the command line.

Examples: .\ShowOrleansSilos.ps1   
 .\ShowOrleansSilos.ps1 C:\Orleans\MyDeploy.xml

## Monitoring Orleans

Once Orleans is deployed, you can start an optional script that will monitor the Orleans deployment using standard performance counters. Run a dedicated PowerShell command prompt as an administrator, and execute the .\**MonitorOrleans.ps1** script to start monitor performance counters for an Orleans Deployment.

The following parameters configure the monitoring to suit individual circumstances:

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Default |
| DeploymentConfigFile | The deployment manifest used to install Orleans. | Deployment.xml |
| NetworkInstance | The name of the network for the network performance counters. | corp |
| SamplesToLog | The number of samples to record in the current run. Use Ctrl-C to stop the script sooner. | 480 which taken in one minute intervals should continue for eight hours |
| HeaderInterval | The number of samples to write before repeating the header. | 10 |
| RepeatHeaderInFile | If this switch is present, the header will be repeated in the log file at the interval specified by the previous parameter. | Only include the header at the top of the file. |

The script will store the data in the following types listed below. The files will be written to a folder called PerformanceData under the directory where the monitoring script is run from.

|  |  |  |  |
| --- | --- | --- | --- |
| File Type | Description | | FileNameBase |
| Machine Specific | | Contains only the data for a single machine. If there are four machines in the deployment, then there will be four of these files. | “PerfData-“ + the machine name and the Date/Time stamp. |
| Combined | | Contains all of the data for all machines consolidated into a single file. | “ConsolidatedPerfData-“ + the Date/Time stamp. |

## Gathering Orleans Log Files

To retrieve all log files from deployment silos and store them in the specified output folder, run the **GatherOrleansSiloLogs.ps1** script.

If you have used a deployment manifest file named something other than the default, specify it on the command line. You may also specify an output folder where the collected log files will be stored otherwise a .\logs subdirectory will be used by default.

Examples: .\GatherOrleansSiloLogs.ps1   
 .\GatherOrleansSiloLogs.ps1 C:\Orleans\MyDeploy.xml  
 .\GatherOrleansSiloLogs.ps1 C:\Orleans\MyDeploy.xml C:\MyLogs

## Removing Orleans

When it is time to remove an Orleans deployment, use the **UndeployOrleansSilos.ps1** script.

If you have used a deployment manifest file named something other than the default, specify it on the command line.

Examples: .\UnDeployOrleansSilos.ps1   
 .\UnDeployOrleansSilos.ps1 C:\Orleans\MyDeploy.xml