Composite C1 Azure Publisher - User Guide

Composite 2012-10-04

Composite A/S
Nygårdsvej 16
DK-2100 Copenhagen
Phone +45 3915 7600

www.composite.net

Contents

1 INTRODUCTION 3

1.1 Who should read this guide 3
1.2 Getting started 4
1.3 Limitations 4

2 PREREQUISITES 5

2.1 Blob stores 5
2.2 Source website 6

3 COMPOSITE C1 AZURE DEPLOYMENT PACKAGE 7

3.1 Building custom Composite C1 Azure package 7

4 CONFIGURING COMPOSITE C1 AZURE 10

5 DEPLOYING COMPOSITE C1 AZURE ON WINDOWS AZURE 11

6 SETTING UP WINDOWS AZURE TRAFFIC MANAGER 13

7 INSTALLING C1 AZURE PUBLISHER 14

8 CONFIGURING C1 AZURE PUBLISHER 15

9 PUBLISHING WITH AZURE PUBLISHER 16

9.1 Advanced Publisher 17

10 DOWNLOADING WEBSITES WITH AZURE PUBLISHER 18

11 PERFORMANCE COUNTERS 19

12 BLOB CHANGE LOG 21

1 Introduction

Windows Azure with its data centers around the world and its Traffic Manager is perfect for a website that requires load balancing and/or geo-targeting.

Composite C1 with its integration with Windows Azure allows you to have a solution where you have a website running, for example, locally on WebMatrix and “push” changes on this “source” website to a single or multiple online “clone” deployments on Windows Azure.

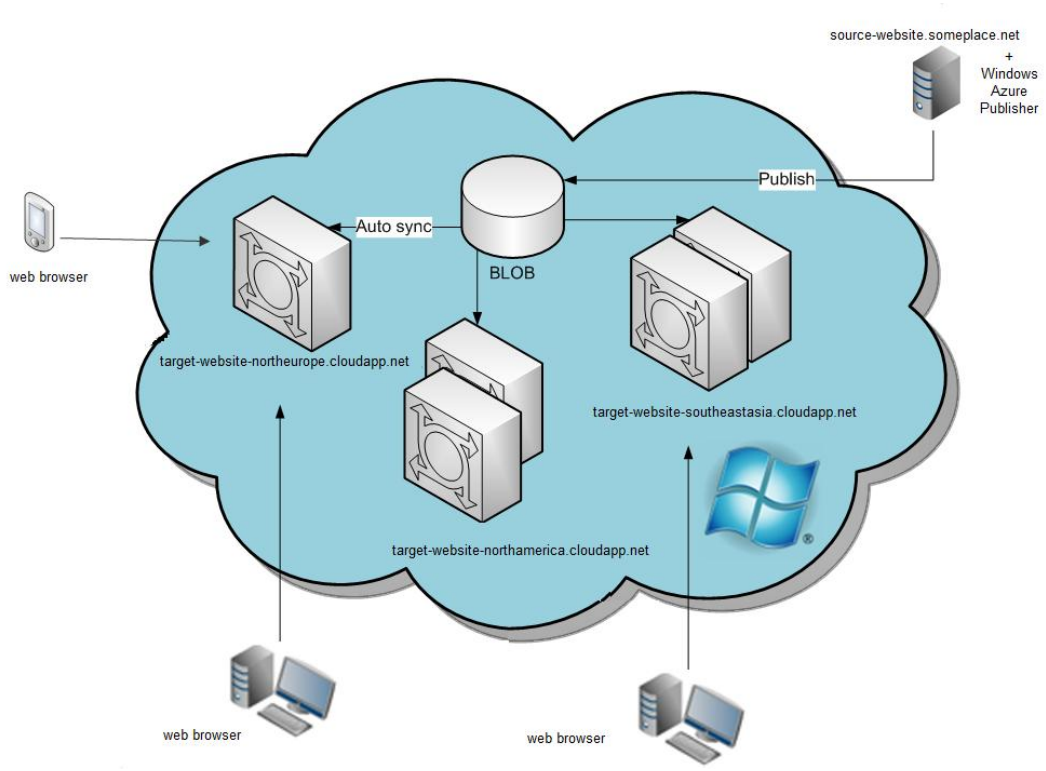

Figure 1: Synchronizing to multiple instances on Windows Azure

You can ensure this synchronization with the Azure Publisher package installed on your source website.

Azure Publisher enables you to publish the changes to your single or multiple website deployments on Windows Azure provided that these deployments in the cloud are "read only" or have Azure-aware code. The multiple-deployment scenario also requires that you use Traffic Manager on Windows Azure for your multiple Composite C1 Azure deployments.

Besides, with its website-downloading feature, you can easily set up a "staging" environment by downloading a "production" website running on Windows Azure to the host of your choice, make changes on this staging website and upload those changes to the production website with the publishing feature of the package.

1.1 Who should read this guide

The guide is intended for a technical person with access to the Windows Azure portal and capable of creating a new blob store and deploying packages on Windows Azure.

We expect that this person has a Composite C1 website running on XML.

Please note that the guide requires that Visual Studio 2010 and Azure SDK are used to compile a custom Composite C1 Azure deployment package. They are not needed, however, if you are using one of the pre-built Composite C1 Azure deployment packages provided by Composite.

1.2 Getting started

Azure Publisher can be used in two scenarios:

- Single deployment (free license)
- Multiple deployments (commercial license)

Most steps for both scenarios are identical and will be further presented as one procedure regardless these two scenarios. However, they differ in deploying and configuring Composite C1 Azure package, which will be treated separately in this guide.

Before you install, configure and start using Azure Publisher:

1. You should make sure that the prerequisites are in place.
2. Prepare if necessary and deploy one or more Composite C1 Azure deployment packages on Windows Azure.
3. In the multiple-deployment scenario, you should also configure performance policy with Windows Azure Traffic Manager for load balancing and geo-targeting.

When the above steps are completed, you can go on to:

1. Install Azure Publisher
2. Configure Azure Publisher

Now that everything is set up and running, you can start using Azure Publisher:

- Publish a source website to one or more deployments on Windows Azure
- Download the website on Windows Azure to a local host for your “staging” environment
- Monitor performance of the deployed instances on Windows Azure
- View the blob change log on Windows Azure

1.3 Limitations

1. C1 Azure Publisher works in the setup where there is one "source website" running wherever you like and one or more Windows Azure “Web roles”. C1 Azure Publisher is one-way in data handling (a source website to Web roles).
2. If Web roles collect any data from visitors etc, you should address this need on your own. C1 Azure Publisher does not handle these cases.

2 Prerequisites

Make sure you have these prerequisites in place:

- A blob store on Windows Azure
- A source C1 website

2.1 Blob stores

You should have a blob store where you will publish your “source website” (first time) or changes on it.

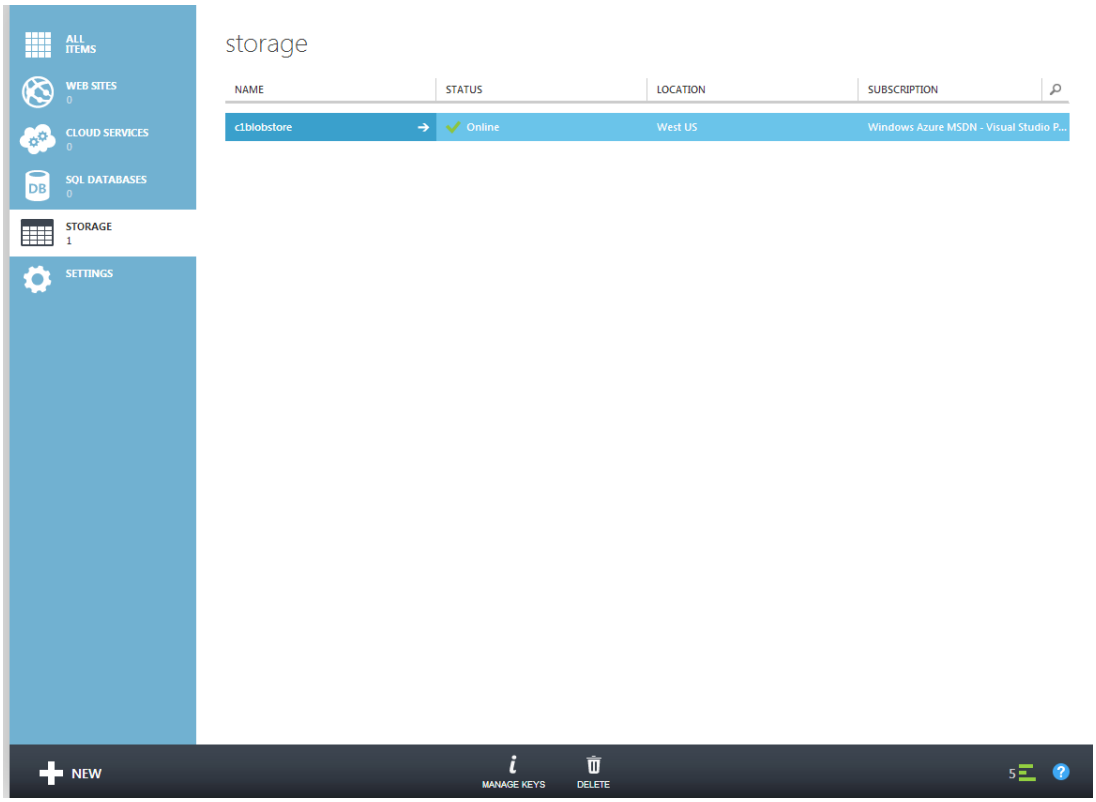

Figure 2: A blob storage on Windows Azure

You should make a note of its *account name* and *account primary key*. You will need this information when deploying a Composite C1 Azure package.

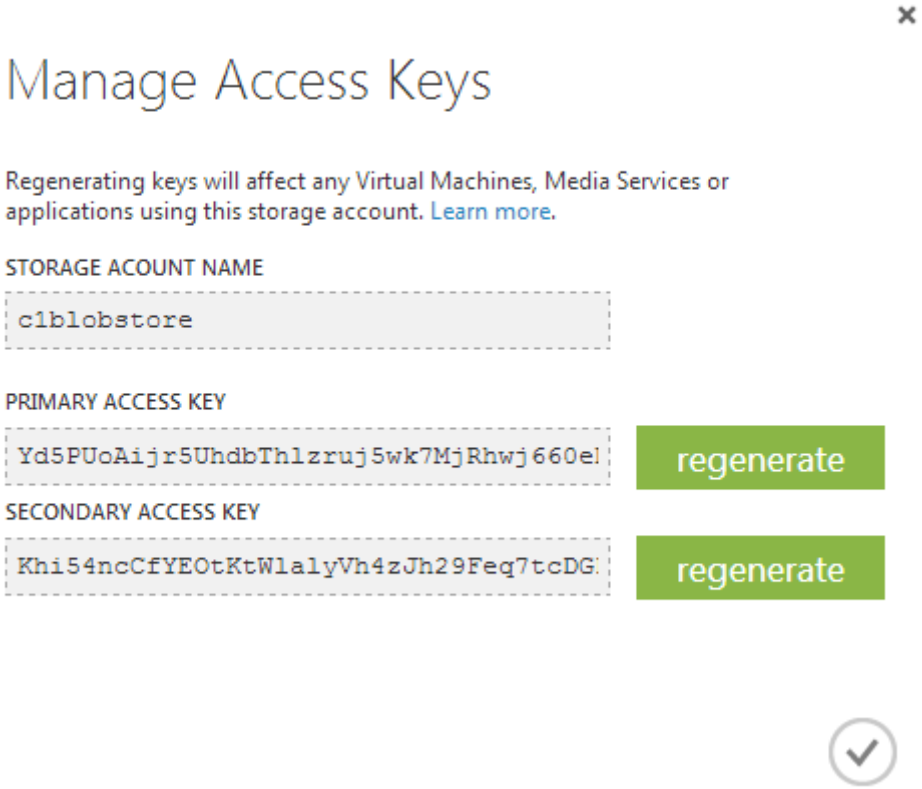

Figure 3: Getting the blob storage name and access key

When configuring C1 Azure Publisher you will need the entire blob connection string that includes the account name and account key as well as the default website container name (both taken from the Composite C1 Azure's configuration file).

It is a "recipient" endpoint (target). Here, you will deploy the Composite C1 Azure package to enable synchronization from the source website to here.

People will visit the site(s) deployed here.

2.2 Source website

This is a Composite C1 XML-based website. (The SQL-based website is not covered in this guide.)

We recommend using Composite C1 version 3.2 or later.

You can deploy it wherever it is convenient for you. It might be a local website running on WebMatrix, or an online instance on Windows Azure or your ISP.

It is a "sender" endpoint (source). Changes made here will be pushed to the online instances on Windows Azure.

The source website can be considered as a "staging" website while the website on Windows Azure - a "production" website (especially in a single-deployment scenario).

This website can also be occasionally called "local", which might be a little misleading, since you are not limited to where you host this website: on your local or some remote machine.

3 Composite C1 Azure Deployment Package

The Composite C1 Azure package (cspkg) must be deployed on all the online instances on Windows Azure where changes from the source website must be “pushed” to.

The package is distributed in two forms:

- As a pre-built solution for virtual machines of various sizes (ExtraSmall, Small, Medium, Large, ExtraLarge)
- As a VS2010 project that you should adapt to your needs and build a package from.

The pre-built packages are provided by Composite and available at the Windows Azure Download page. A pre-built package is ready for deployment on Windows Azure. If you choose to use this package, you should skip the step of building a custom Composite C1 Azure package and take the steps of configuring and then deploying it.

You can use the package distributed as a VS2010 project, too. This is an additional step you need to take before configuring and then deploying it and normally needed if you want a custom Composite C1 Azure package adapted to your specific needs.

Regardless the form of the package you choose to use, the package should include:

- CompositeC1Azure.cspkg, that is, the package file itself
- ServiceConfiguration.cscfg, the package’s configuration

Please read the following sub-section if you choose to build your own, custom package.

3.1 Building custom Composite C1 Azure package

NOTE: Take this step only if you need a custom Composite C1 Azure package.

When you use a VS2010 project rather than pre-built solution, you can compile it as it is into a deployable package unless you want to first make some specific changes or tweaks for your own purposes. These possible changes are not covered in this guide.

IMPORTANT: Make sure you have installed Azure SDK 1.7 or later on your machine.

To create your custom Composite C1 Azure deployment package:

1. Download and unzip the CompositeC1Azure source code.
2. Open the solution in Visual Studio: CompositeC1Azure.sln.

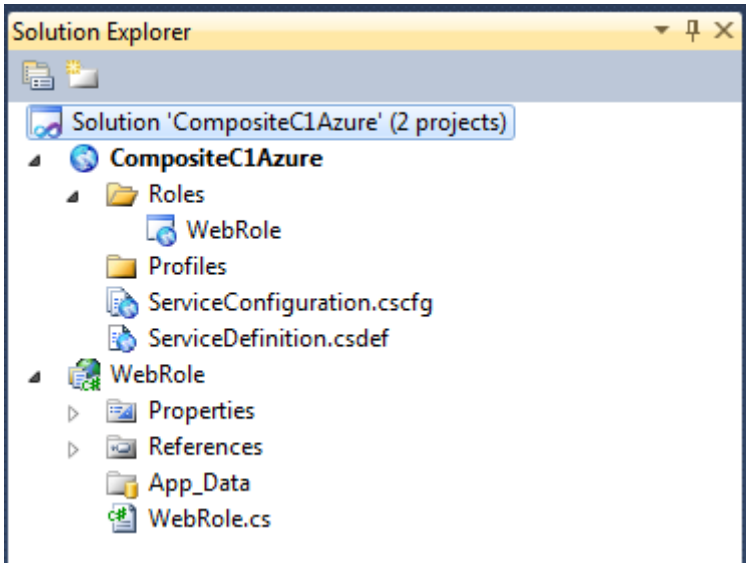

Figure 4: Composite C1 Azure project in Visual Studio

3. If necessary, edit \CompositeC1Azure\ServiceDefinition.csdef and specify the required size of the virtual machine in vmsize: "ExtraSmall" (default), "Small", "Medium", "Large", "ExtraLarge".
4. Make other changes if needed.
5. Right click the **CompositeC1Azure** project in Visual Studio and select **Package**.

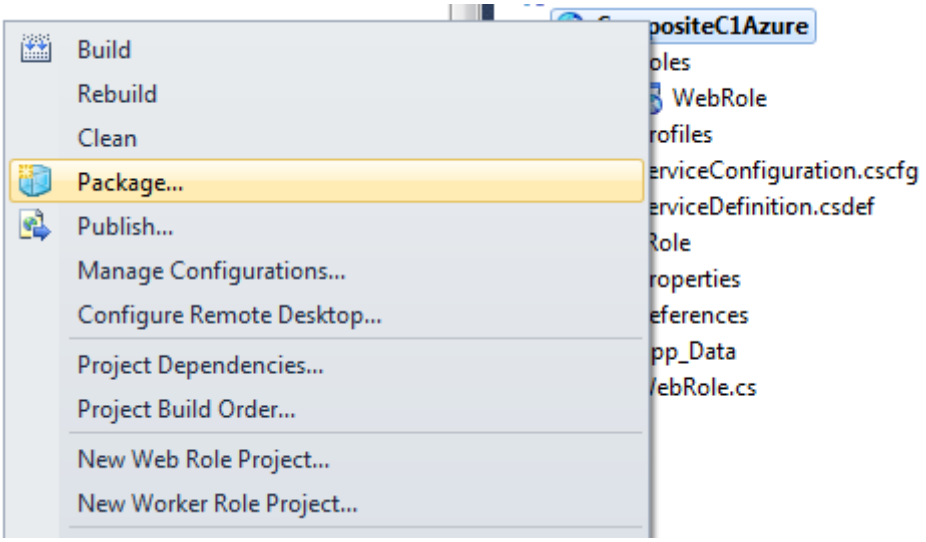

Figure 5: Creating a Composite C1 Azure package

6. In the **Package Windows Azure Application** window, specify the necessary values: "Service configuration" and "Build configuration".

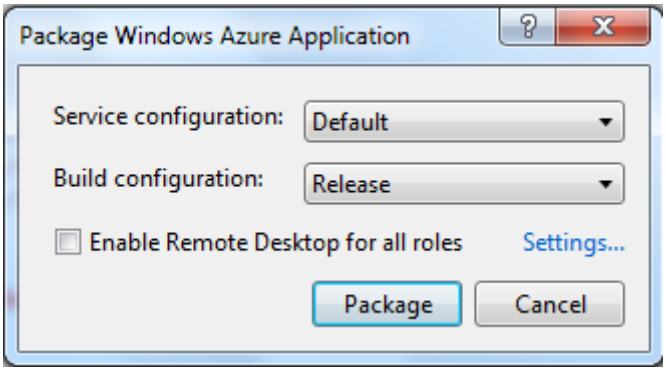

Figure 6: Selecting the package's options

7. Enable **Remote Desktop** if necessary, and configure its **Settings** as you need.

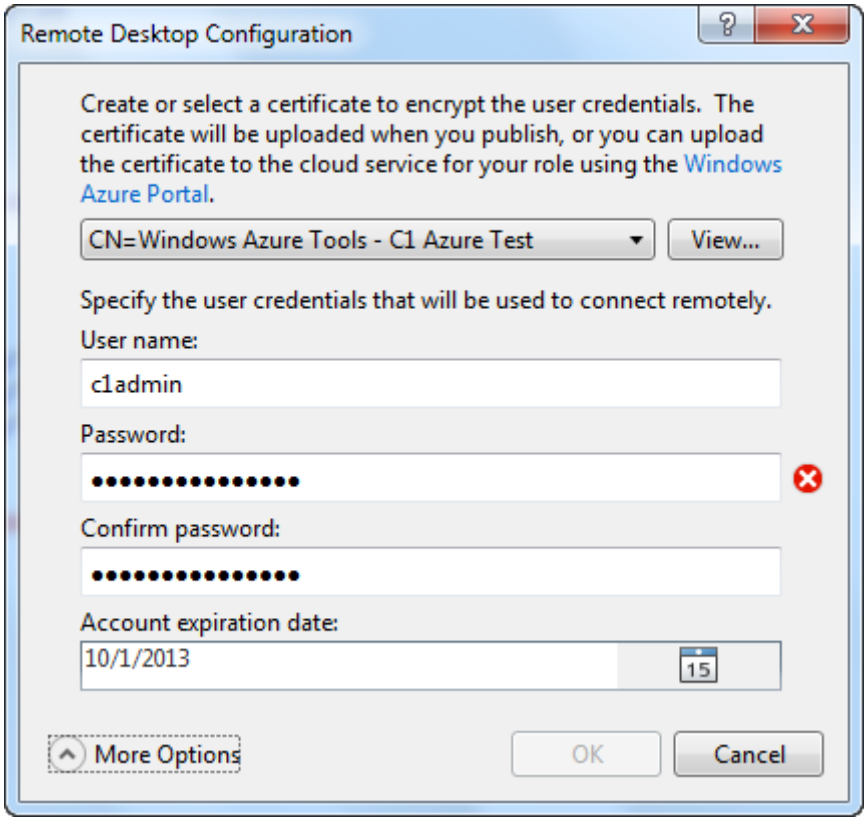

Figure 7: Setting up a Remote Desktop access (optional)

8. Click **Package**. The building process will start.

Once the package is built, you'll find these files in \CompositeC1Azure\bin\Release\app.publish:

- ServiceConfiguration.cscfg
- CompositeC1Azure.cspkg

These are the files you should use for your deployment instead of default Composite C1 Azure bits by Composite.

Now you need to configure configuration file for your deployment(s) on Windows Azure.

4 Configuring Composite C1 Azure

Before deploying the Composite C1 Azure on Windows Azure, in the ServiceConfiguration.cscfg file, you should provide the following information specific to your Storage Account (**BlobConnectionString**) and your deployment (**DisplayName**) on Windows Azure.

BlobConnectionString specifies the blob where your website is published to. Here, you should provide the information on your specific Storage Account such as:

- Account Name
- Account Primary Key

The storage account (blob store) will have its own account name and its primary key you should have made a note of in advance.

(Please make note of the BlobConnectionString and DefaultWebsiteName settings when set up. You will use their values when configuring Azure Publisher.)

DisplayName is the descriptive title of your deployment, the name used in the C1 Console as a "publishing destination node", allowing the user of Azure Publisher (normally, the administrator) to see update progress for that node.

If you deploy to multiple Azure hosting centers, consider using the display name accordingly so each deployment have individual names.

```
<ServiceConfiguration serviceName="CompositeC1Azure"
xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceConfigura
tion" osFamily="2" osVersion="*" schemaVersion="2012-05.1.7">
  <Role name="WebRole">
    <Instances count="1" />
    <ConfigurationSettings>
      <Setting name="DeploymentName" value="compositec1" />
      <Setting name="DefaultWebsiteName" value="defaultwebsite" />
      <Setting name="DisplayName" value="North Europe" />
      <Setting name="BlobConnectionString"
value="DefaultEndpointsProtocol=http;AccountName=c1blobstore;AccountKey=
Yd5PUoAijr5UhdbThlzruj5wk7MjRhwj560eEdQqyA2Hh8bim8w957R6gnKXESLCsppGZsZ8uXr
w9pD06Iw+iA==" />
      <Setting name="ZippedWebsiteUrl"
value="http://package.composite.net/AzureInstallFiles/DownloadWebsite.aspx"
/>
      <Setting name="CompositeC1AzureRuntimeFilesUrl"
value="http://package.composite.net/AzureInstallFiles/Azure17" />
      <Setting name="CheckConfigUpdateTime" value="15000" />
      <Setting
name="Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString" value=""
/>
    </ConfigurationSettings>
  </Role>
</ServiceConfiguration>
```

Listing 1: Example of ServiceConfiguration.cscfg

For other configuration settings in ServiceConfiguration.cscfg, please see Composite C1 Azure Standard Configuration.

5 Deploying Composite C1 Azure on Windows Azure

On Windows Azure, create a hosted service (also known as a “cloud service’) using the blob store you have specified in ServiceConfiguration.cscfg.

For the multiple-deployment scenario, create as many hosted services as you need.

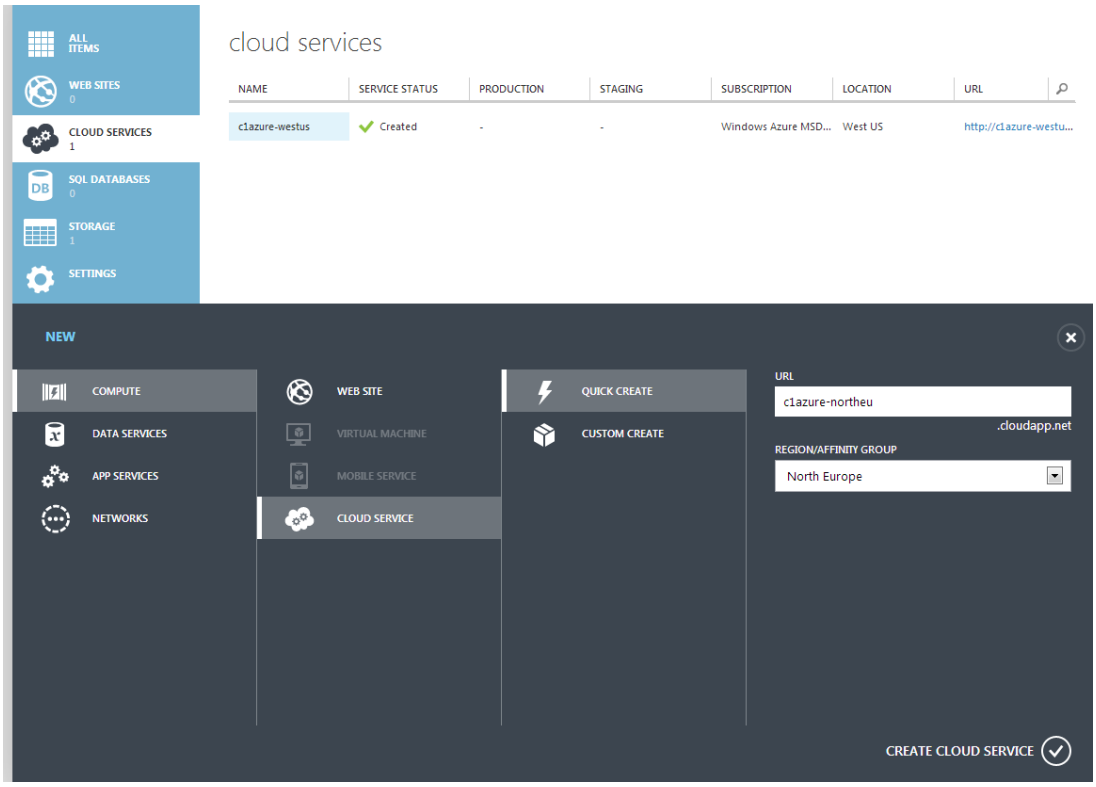

Figure 8: Creating a cloud service on Windows Azure

For the single-deployment scenario, you need only one hosted service.

On the hosted service, deploy your Composite C1 Azure package by specifying:

- the Composite C1 Azure package file (CompoisteC1Azure.cspkg)
- the service configuration file (ServiceConfiguration.cscfg)

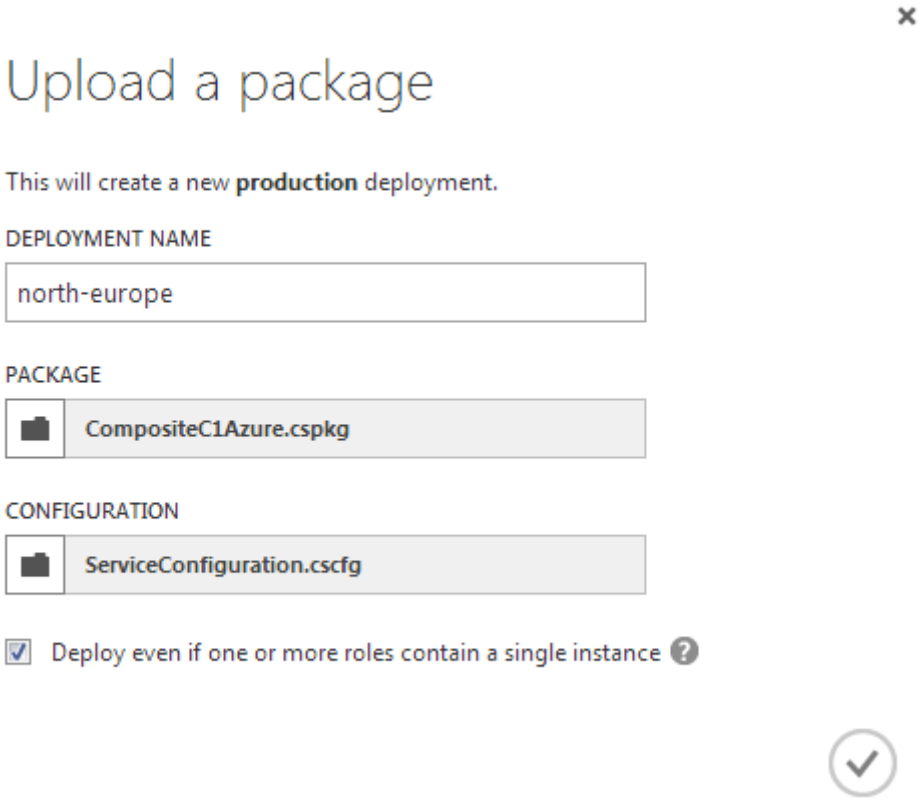

Figure 9: Uploading Composite C1 Azure package

For the multiple-deployment scenario, repeat the deployment of the package but use ServiceConfiguration.cscfg with a different value for the DisplayName setting.

When deploying, you can choose the deployment type: staging or production.

(In the old Windows Azure portal, normally, creating a hosted service and deploying Composite C1 Azure package can be done in one step. However, you can always delete the deployment itself from the hosted service and deploy it again.)

6 Setting Up Windows Azure Traffic Manager

NOTE: Take this step only if you use the multiple-deployment scenario.

You should set up Traffic Manager on Windows Azure to load balance incoming traffic to your website geographically.

1. In the Windows Azure management portal, navigate to **Virtual Network** | **Traffic Manager**.
2. Click **Create**.

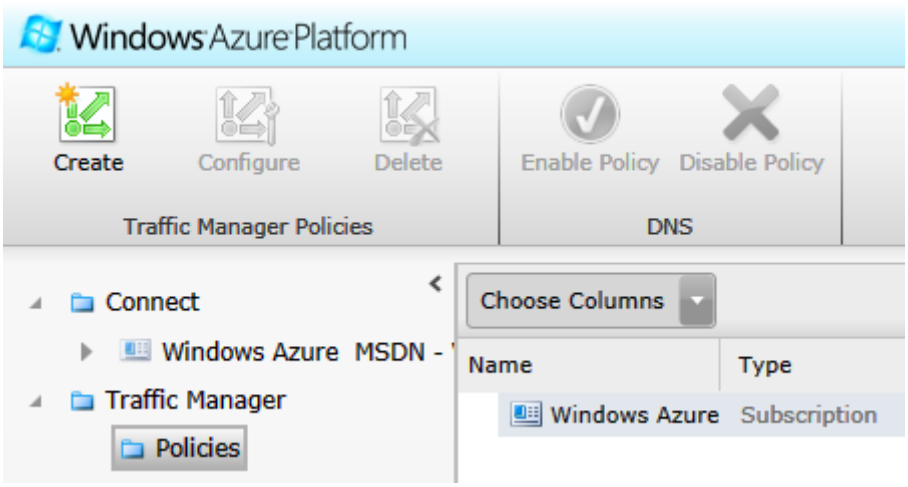

Figure 10: Creating a Traffic Manager policy

3. Select the balancing method: "Performance".
4. Select DNS names (your deployments).
5. Specify the **Traffic Manager DNS prefix**.
6. Change the port number from 80 to 8888.
7. Click **Create**.

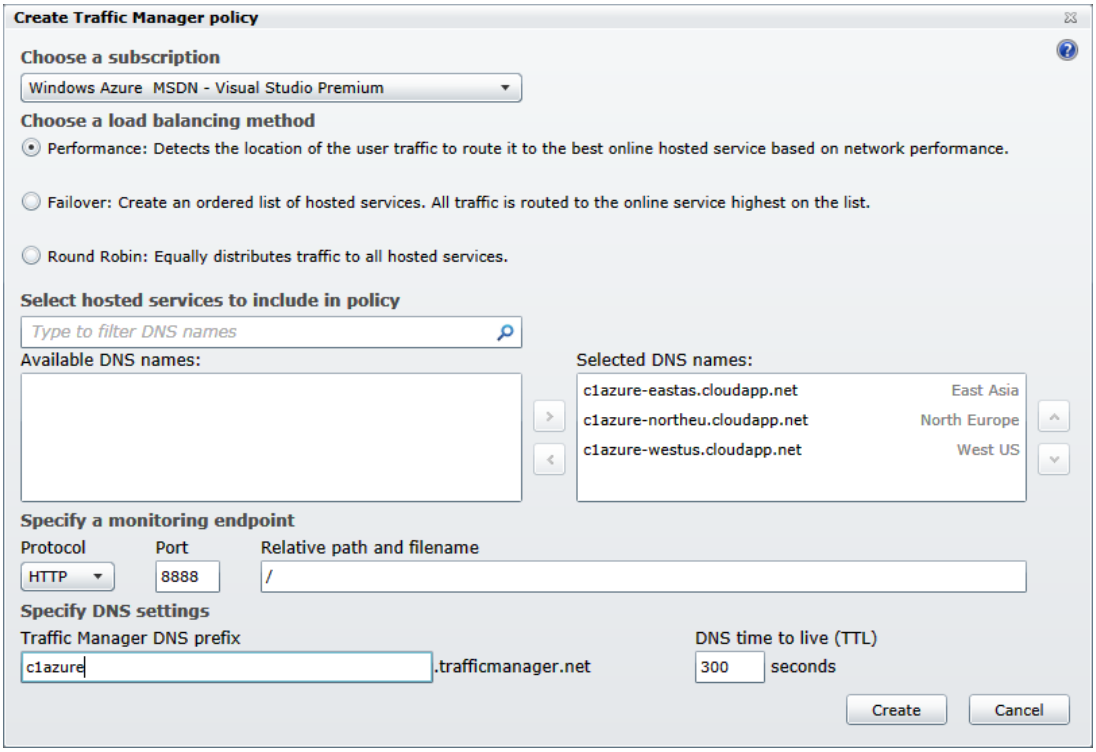

Figure 11: Policy for performance

7 Installing C1 Azure Publisher

Now that you deployed Composite C1 Azure and there is one or more website running on Windows Azure, you need to install C1 Azure Publisher on your source website.

It turns a Composite C1 website into a "Windows Azure staging server with a publish-to-azure button".

To install the package:

1. Log in to the C1 Administrative Console
2. In the **System** perspective, expand **Packages**, **Available Packages, Composite.Azure**.
3. Select **Composite.Azure.Publisher** and click **Package Info** on the toolbar.
4. In the **Package Info** view, click **Install**.
5. Complete the wizard.

Once the package has been installed, the C1 Console will reload and the "Windows Azure Publisher" element will appear in the Content perspective.

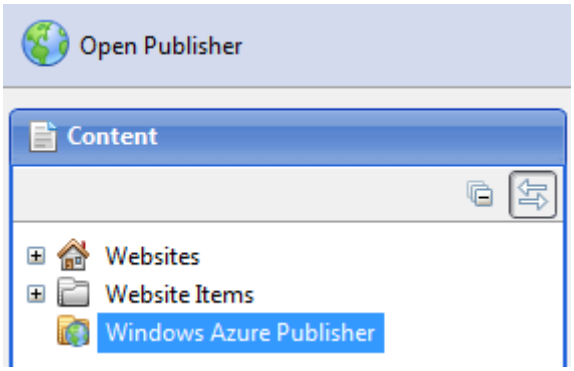

Figure 12: Windows Azure Publisher in the Content perspective

8 Configuring C1 Azure Publisher

To configure C1 Azure Publisher:

1. In the **Content** perspective, right-click **Windows Azure Publisher**.
2. Click **Configuration** in the shortcut menu to open the **Configuration** view.

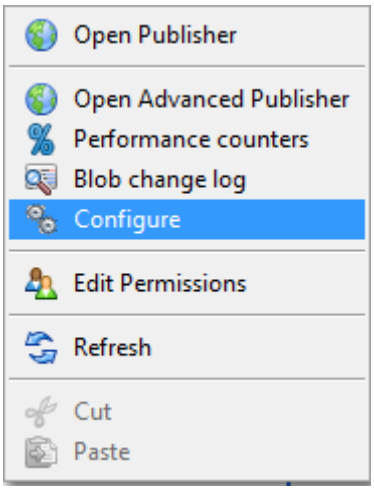

Figure 13: Selecting Configure from the menu

3. Fill out the fields:
   - **Blob connection string**: The blob connection string used in your Composite C1 Azure deployment. The same as the value of the BlobConnectionString parameter in ServiceConfiguration.cscfg.
   - **Website container name**: The blob container where the website files are placed. The same as the value of the DefaultWebsiteName setting in ServiceConfiguration.cscfg.
4. Click **Save**.

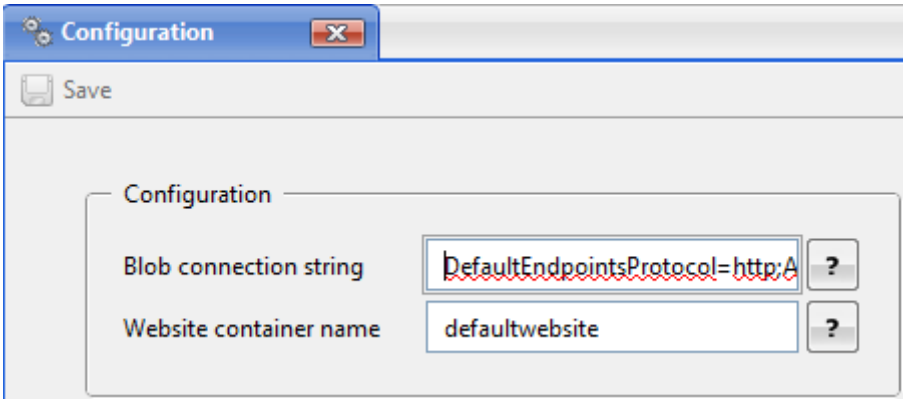

Figure 14: Configuring Azure Publisher

9 Publishing with Azure Publisher

At this point, you must have C1 Azure Publisher correctly configured.

When you publish your source website for the first time, it may take a few minutes because the entire website will be pushed to online instances.

When publishing, you synchronize the website to blobs. The Azure packages deployed on the online instances will pick up on this and update themselves.

Next time you publish, only changes will be published, and it will thus take significantly shorter time.

The changes can be "anything" in the website folder, including new assemblies, CSS, data schemas, content etc.

To publish:

1. Log in to the C1 Administrative console.
2. In the **Content** perspective, select **Windows Azure Publisher**.
3. Click **Open Publisher** on the toolbar to open the **Windows Azure Publisher** view.

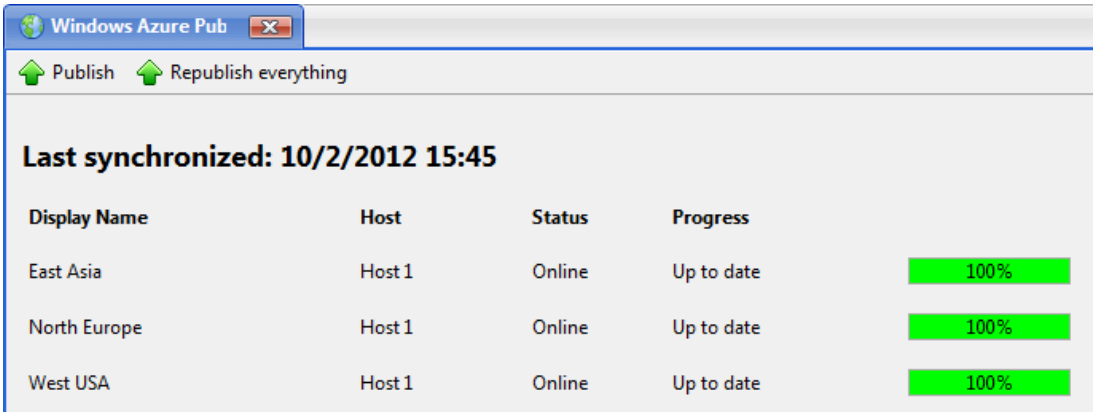

Figure 15: Windows Azure Publisher

4. In the view, click **Publish**.

The synchronization will start? and you will be able to see the file upload progress to the blob store.

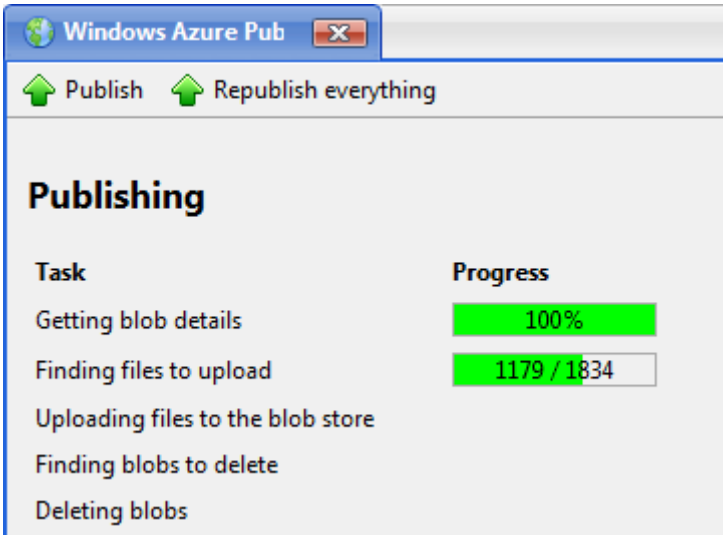

Figure 16: Uploading changed files to the blob store

Once the files are uploaded, you will be able to track the synchronization progress and status for each web role instance in each deployment.

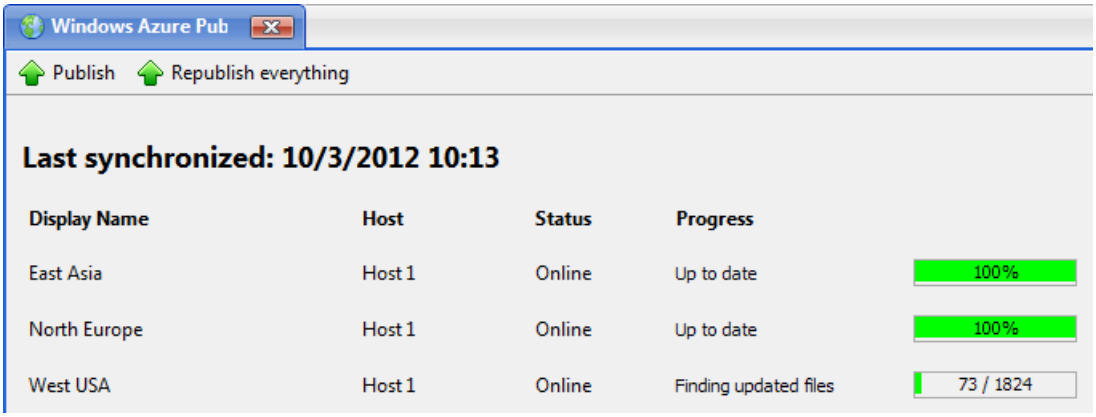

Figure 17: Synchronizing to multiple deployments

If you want to republish your entire source website rather than only synchronize changes, click **Republish Everything**.

9.1 Advanced Publisher

You can also use an advanced version of Azure Publisher to publish or republish your changes. (For information about downloading websites, please see “Downloading Websites with Azure Publisher”).

Advanced Publisher gives you more detailed information about the status and progress of synchronization.

To publish with Advanced Publisher:

1. Log in to the C1 Administrative console.
2. In the **Content** perspective, right-click **Windows Azure Publisher**.
3. Select **Open Advanced Publisher** in the context menu.

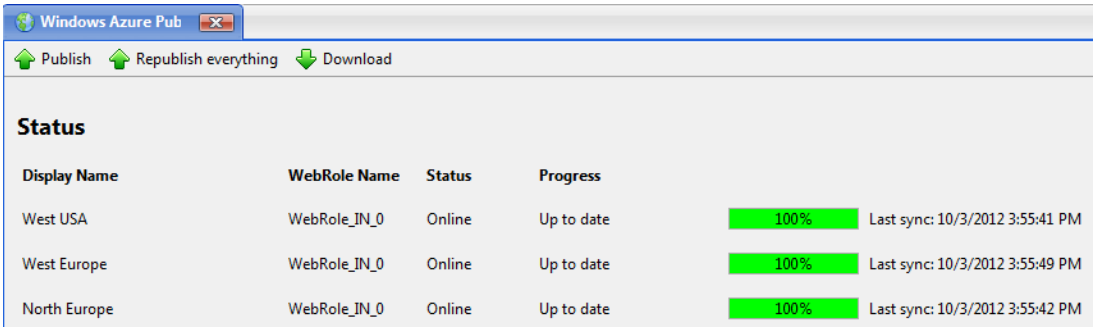

Figure 18: Advanced Publisher

4. In the view, click **Publish**.

The advanced view shows the last synchronization date and time for each Composite C1 Azure deployment.

10 Downloading Websites with Azure Publisher

Apart from uploading changes to a website on Windows Azure, Azure Publisher allows you to download the website from Windows Azure and run it on a host of your choice. In this way you can set up a "staging" copy of a website from Windows Azure, for example, locally on WebMatrix or IIS (or even remotely on your ISP hosting or Windows Azure.)

And with its publishing features, you can synchronize your local changes back to the "production" website (as described above).

To download a website with Azure Publisher, first prepare a local website:

1. Locally install a new "blank" website selecting "Bare Bones" in the setup wizard.
2. Install **Composite.Azure.AzurePublisher**.
3. Configure the "Blob connection string" and "Website container name" fields in Azure Publisher as described above, pointing it to the blob with your website.

Now go to download the website:

1. In the **Content** perspective, right click **Windows Azure Publisher** and select **Open Advanced Publisher**.

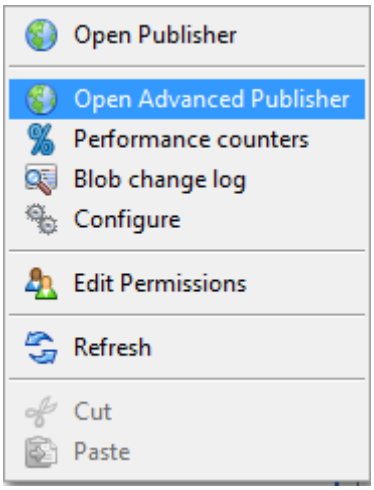

Figure 19: Selecting Open Advanced Publisher from the menu

2. In the **Windows Azure Publisher** view, click **Download**.

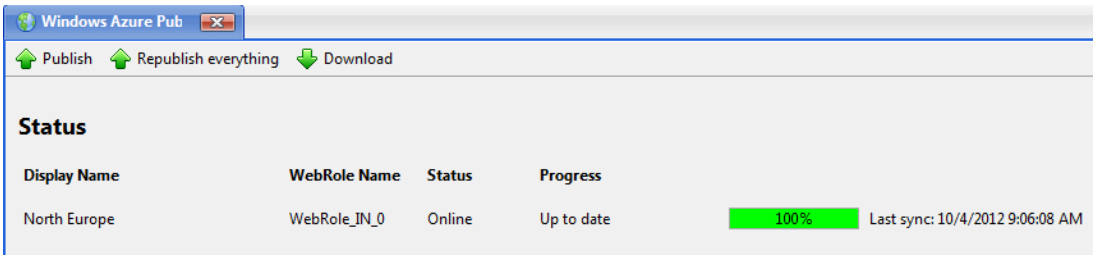

Figure 20: Advanced Publisher with the Download feature

It might take some time to download and locally set up the website. Once the website has been download and ready, the C1 Console will reload and you will have the copy of the website from Windows Azure.

In some cases, you might need to reload the C1 Console manually.

11 Performance Counters

You can view the website performance using performance counters. The default configuration provides you with these counters:

- Processor time use (%)
- Available memory (Mbytes)
- ASP.NET Applications: The number of requests / second
- ASP.NET Applications: The total number of requests
- The number of bytes in all heaps of .NET CLR memory

To view the performance counters:

1. In the **Content** perspective, right click **Windows Azure Publisher** and select **Performance counters**.

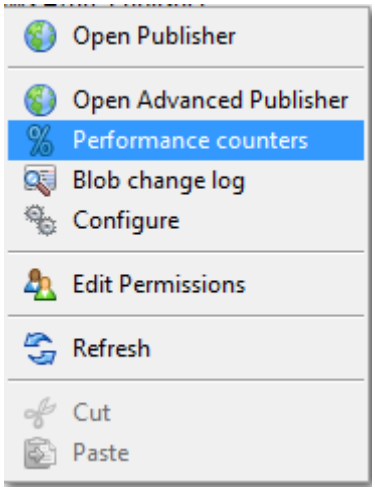

Figure 21: Selecting Performance counters from the menu

2. Check the charts in the **Performance Counters** view.

Each Composite C1 Deployment has its own chart.

Figure 22: Performance counters

You can change the time period to view the counters of by selecting the value form the drop box - the last 1 to 36 hours.

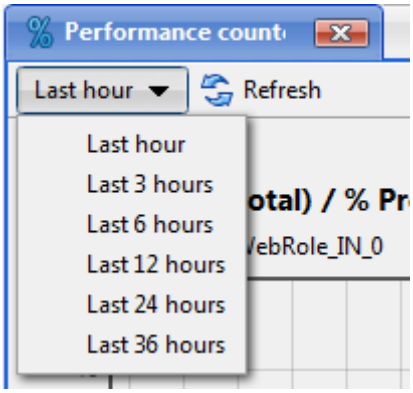

Figure 23: Time periods for the counters

You should configure the performance counters in your Composite C1 Azure deployment on Windows Azure. For information about configuring the counters, please refer to "Performance Counters".

12 Blob Change Log

In Windows Azure Publisher, you can also keep track of blob changes - in the blob change log.

To view the blob change log:

1. In the **Content** perspective, right click **Windows Azure Publisher** and select **Blob change log**.

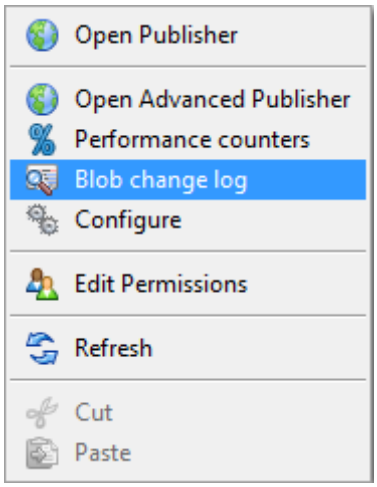

Figure 24: Selecting Blob change log from the menu

2. Check the entries in the **Blob change log** view.

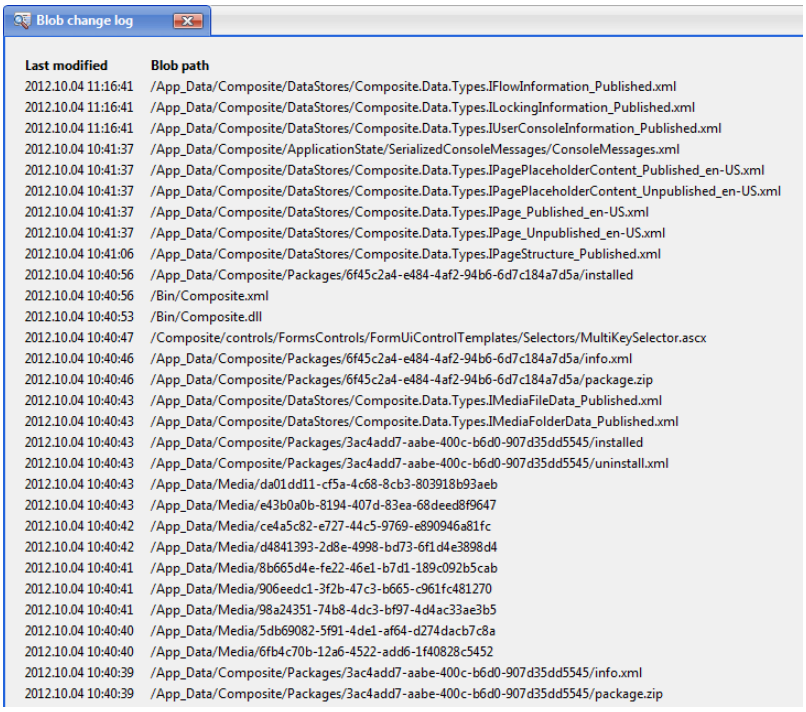

Figure 25: Blob change log

In the log, you can see the file (with the path) and its last change date.