

To utilize web farm synchronization, the system requires that all servers that make up a web farm are registered and configured in the system. To start using the web farm synchronization:

1. Choose a web farm mode:
 - [Automatic web farm](#)
 - [Manually configured web farm](#)
2. [Perform further configurations.](#)

You can also perform additional low-level settings for web farm synchronization by adding the keys listed in [Web.config file settings](#) into the **/configuration/appSettings** section of your web.config file.

Configuring web farms automatically

For most web farm environments, we recommend the automatic web farm mode. This mode dynamically adds and removes web farm servers from the system.



When you are running Kentico on **Azure**, the automatic web farm mode is enabled by default.

To set the automatic mode, we recommend that you first enable the automatic web farm on **one server** and after that you start adding other servers to the web farm (connect them to the database).

1. Open the **Settings** application.
2. Select the **Versioning & Synchronization -> Web farm** category.
3. Select **Automatic** as the **Web farm mode**.
4. Click **Save**.

Now you can start adding other servers to the web farm (connect the servers to the database). The system will register them automatically.



Automatic server name format

The system generates the names of automatic web farm servers by combining the machine name and the virtual directory in which the application is running.

If this combination of variables is not unique in your environment (for example if you run your web farm instances on the same machine without virtual directories, e.g. distinguished by port numbers), you need to manually assign unique server names to the web farm servers. To set a web farm server name, see *Step 5* in [Configuring web farms manually](#).

Configuring web farms manually

If you need to have a complete control over your web farm or you are running a web farm with a static number of web farm servers, which you do not plan to change, configure the web farm servers manually in your system.

First, enable the manual web farm mode.

1. Open the **Settings** application.
2. Select the **Versioning & Synchronization -> Web farm** category.
3. Select **Manual** as the **Web farm mode**.
4. Click **Save**.

Now you must configure the web farm servers in the system.

1. Open the **Web farm** application.
2. Click **New server**.
3. Fill in the following fields:
 - **Server display name** - a descriptive name for the server displayed in the administration interface.
 - **Server enabled** - allows you to manually enable or disable web farm synchronization for the particular server.

4. Click **Save** to register the server.
5. Open the web.config file on the particular server and add the **CMSWebFarmServerName** key into the *appSettings* section:

```
<add key="CMSWebFarmServerName" value="ServerCodeName" />
```

Replace **ServerCodeName** with the server code name that the system created for the server (or the code name that you manually entered). Every server must contain only one such key with its own name.



Setting the server name in the web.config file overrides the automatic naming done by the system in the Automatic web farm mode.

Repeat the process for every server in your web farm.

Keep in mind that if you remove a machine from a web farm, you also need to disable or remove the corresponding server from the system.

Further configurations of web farm servers

When you have defined the web farm servers and enabled the web farm, go through the following configurations to ensure that your web farm works correctly.

Storing session state on a web farm

The default session state mode for ASP.NET applications is the **InProc** mode, which stores session state in the memory of the server. InProc session state mode is not supported on most types of web farms because requests for the same session can be served by different servers. This can cause unexpected behavior and data loss. For example, items stored in a shopping cart can disappear after making another request on the website.

If you are using a load balancer in sticky-session mode, you can use the default InProc mode. However, visitors may still experience session loss when a server in the web farm becomes unavailable.

To configure your web farm to store session state data correctly, use one of the following modes:

- **StateServer** mode - stores session state in a separate process.
- **SQLServer** mode - stores session state in an SQL server database.

To use one of these modes, adjust the `<sessionState>` element and its attributes in the **web.config** file. For more information, see the [Session-State Modes](#) article.



To learn how to configure Azure projects to store session data, see [Storing cache and session state data in an Azure environment](#).

Configuring the machineKey element in web.config

By default, the Internet Information Services (IIS) server uses an automatically generated machineKey to encrypt and validate view state and forms authentication information. If a user tries to access the same session from two servers with different machine keys, the validation fails, and a Message Authentication Code (MAC) view state validation error occurs. To prevent this, the machineKey element needs to be the same across all Kentico instances in the web farm configuration.

Generating a machineKey element



Never use any websites that will generate a machineKey for you with the click of a button. It is impossible to know whether these keys were created securely or if they are being recorded to a secret database. You should only ever use machineKey configuration elements that you created yourself.

For IIS 7 and below:

Machine keys can be generated in the **Machine Key** section under ASP.NET in the IIS Manager:

1. Select your preferred encryption and decryption methods.
2. Clear the **Automatically generate at runtime** and **Generate a unique key for each application** check boxes.
3. Click **Generate keys**.
4. Insert the machineKey element into the <system.web> section in the **web.config** file of all Kentico instances in your web farm configuration.

```
<system.web>
...
    <machineKey decryption="..." decryptionKey="..." validation="..."
validationKey="..." />
...
</system.web>
```

5. Copy the generated decryption and validation keys into the **decryptionKey** and **validationKey** attributes.



The **decryption** and **validation** attributes need to contain the names of your specified encryption and decryption algorithms. If you left the **Decryption method** set to *Auto* when generating the machine keys in the IIS Manager, enter *AES* as the decryption.

For IIS 8 and above:

The ability to generate machine keys is not available in the IIS Manager 8 or newer. You can use a PowerShell script to generate the machineKey element according to the instructions in [this article from Microsoft](#). Insert the generated machineKey element into the <system.web> section in the **web.config** file of all Kentico instances in your web farm configuration.

Configuring servers for synchronizing macros

The system uses signatures to ensure the security of [macro expressions](#). Macro signatures contain the user name of the macro's author (or an assigned macro identity) and a hash of the given expression.

The hash function used to create the signatures appends a [salt](#) to the input. To ensure that macro expressions work correctly in a web farm environment, you need to configure all servers to use the same hash salt:

- Set the **CMSHashStringSalt** key in the *appSettings* section of the web.config file to the same value on all web farm servers. For example:

```
<add key="CMSHashStringSalt" value="e68b9ad6-a461-4707-8e3e-ece73f03dd02" />
```

The best option is to set the hash salt value before you start creating content for your website. Changing the salt causes all current hash values to become invalid. To fix existing macro expressions in the system after changing the hash salt, you need to re-sign the macros. See [Working with macro signatures](#) for more information.

Setting up a shared file storage for a web farm

The web farm synchronization mechanism synchronizes files reliably up to the size of 250 MB. Larger files take a long time to synchronize. Even though the synchronization for larger files may work without problems in your environment, we recommend that you configure a **shared storage** for your web farm in such case.

You can configure the following shared file storages for your web farm:

- [Azure Blob Storage](#)
- [Amazon S3 storage](#)

- Custom network storage - you need to [configure a file system provider](#).

When you configure a shared storage for you web farm, the system automatically recognizes this setup and stops creating synchronization tasks for these files.

Configuring scheduled tasks to be executed on every server of a web farm

If you are using a web farm, [scheduled tasks](#) will be executed randomly by one of the servers. However, scheduled tasks that work with the application file system should be executed by each server in the web farm.

The following scheduled tasks must be **executed by each server** in a web farm, if you are NOT using a shared file storage (Azure Blob Storage or Amazon S3 storage):

- Execute local [search tasks](#)
- Optimize local search indexes
- Process [analytics](#) log
- Delete image editor history
- Delete old file system cache files
- Delete old temporary upload files
- Delete old temporary attachments
- Clean Azure storage cache

The following scheduled task must be executed by each server in a web farm regardless of the used file storage:

- Clean unused memory

To configure these tasks to be executed by each server in a web farm, clone the original tasks and configure them to be executed on specific servers:

1. Open the **Scheduled tasks** application.
2. Edit the scheduled task.
3. Type the **Server name** according to the name of one of the servers in the web farm and click **Save**.
4. Return to the main page of the **Scheduled tasks** application.
5. Click **Other actions (...)** next to the scheduled task and select **Clone**.
6. Choose a name for the cloned task and click **Clone**.
7. Edit the cloned scheduled task.
8. Change the **Server name** to the name of a different server in the web farm and click **Save**.
9. Repeat for each server in your web farm.

Perform this procedure for all scheduled tasks listed above.

When creating new scheduled tasks, you can ensure that the task is created for all web farm servers by selecting the **Create tasks for all web farm servers** check-box.

SSL in a web farm environment

If you use an SSL offload device or accelerator as part of your web farm and your website is configured to require SSL for the administration interface or on specific pages, you may encounter problems with redirection loops.

For this type of scenario, you need to add some custom code to your website according to the instructions in [SSL accelerator support](#).

Using port numbers to identify web farm servers

If your environment uses unique port numbers to identify web farm servers (for example with a load balancer that internally maps URLs to ports), the system generates certain types of URLs with the internal port number. Such URLs may not be processed correctly, leading to problems with related features (for example *GetResource* requests for files).

To resolve these problems, add the following key into the *appSettings* section of the web.config file on all servers:



```
<add key="CMSURLPort" value="80" />
```

The key ensures that the system generates all URLs without the port number (the default HTTP port is used).