Desktop Client Deployment and Setup

- Choosing the right approach
- Advanced Windows deployment options
  - Features
  - Installation
  - Changing installed features
  - Installation folder
  - Disabling automatic updates
  - Launch after installation
  - No reboot after installation
- Non-interactive account provisioning
  - Required parameters
  - Optional parameters
  - Behavior
  - Common failure cases
  - Examples
  - Recommendations
- Interactive wizard preconfiguration
  - Behavior
  - Examples
  - Important distinction from non-interactive provisioning
- Summary

This chapter describes administrator-facing deployment and setup options for the Nextcloud Desktop client.

These options are intended for managed rollouts and other administrative scenarios, such as deployment scripts, software-management platforms, login scripts, MDM or RMM workflows, and

other automated installation or configuration processes.

Depending on your deployment goals, you can use the desktop client in one of several ways:

- customize Windows installation behavior,
- create an account automatically during deployment without user interaction, or
- prefill or constrain the interactive setup wizard.

Choosing the right approach

Use the option that best matches your deployment goal:

I want to customize how the client is installed on managed Windows systems.

See Advanced Windows deployment options.

I want to silently create an account during deployment.

See Non-interactive account provisioning.

I want users to complete setup interactively, but with predefined values or restrictions.

See Interactive wizard preconfiguration.

Advanced Windows deployment options

If you just want to install the desktop client on your local system, simply launch the `.msi` file and follow the installation wizard.

The following options are intended for advanced Windows installations, for example when automating deployment or customizing installed features.

> “□□
>
> Windows installation customization controls how the client is installed. It does not by itself create or configure a desktop-client account. If you want to configure an account after installation, see Non-interactive account provisioning or Interactive wizard preconfiguration.

Features

The MSI installer provides several features that can be installed or removed individually, which you can also control via the command line. If you are automating the installation, run the following command:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi
```

The command installs the client into the default location with the default features enabled.

If you want to disable, for example, desktop shortcut icons, you can change the command to the following:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi REMOVE=DesktopShortcut
```

See the following table for a list of available features:

Feature	Enabled by default	Description	Property to disable
Client	Yes, required	The actual client	
DesktopShortcut	Yes	Adds a shortcut to the desktop	`NO_DESKTOP_SHORTCUT`
StartMenuShortcuts	Yes	Adds a shortcut to the start menu	`NO_START_MENU_SHORTCUTS`
ShellExtensions	Yes	Adds Explorer integration	`NO_SHELL_EXTENSIONS`

Installation

You can also choose to only install the client itself by using the following command:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi ADDDEFAULT=Client
```

For example, if you want to install everything except the `DesktopShortcut` and the `ShellExtensions` feature, you have two possibilities:

1. Explicitly name all the features you actually want to install (whitelist), where `Client` is always installed anyway:

   ```
   msiexec /passive /i Nextcloud-x.y.z-x64.msi ADDDEFAULT=StartMenuShortcuts
   ```

2. Pass the `NO_DESKTOP_SHORTCUT` and `NO_SHELL_EXTENSIONS` properties:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi NO_DESKTOP_SHORTCUT="1"
NO_SHELL_EXTENSIONS="1"
```

> “□□
>
> The Nextcloud `.msi` remembers these properties, so you do not need to specify them again on upgrades.

> “□□
>
> You cannot use these to change the installed features after installation. If you want to do that, see the next section.

Changing installed features

You can change the installed features later by using the `REMOVE` and `ADDDEFAULT` properties.

To add the desktop shortcut later, run the following command:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi ADDDEFAULT="DesktopShortcut"
```

To remove it, run the following command:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi REMOVE="DesktopShortcut"
```

Windows keeps track of the installed features, and `REMOVE` or `ADDDEFAULT` affects only the specified features.

> “□□
>
> You cannot specify `REMOVE` on initial installation, as it will disable all features.

Installation folder

You can adjust the installation folder by specifying the `INSTALLDIR` property:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi INSTALLDIR="C:\Program Files\Non Standard Nextcloud Client Folder"
```

Be careful when using PowerShell instead of `cmd.exe`; getting the whitespace escaping right can be tricky.

Specifying `INSTALLDIR` like this only works on first installation; you cannot simply re-run the `.msi` with a different path. If you still need to change it, uninstall the client first and then reinstall it to the new location.

Disabling automatic updates

To disable automatic updates, pass the `SKIPAUTOUPDATE` property:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi SKIPAUTOUPDATE="1"
```

Launch after installation

To launch the client automatically after installation, pass the `LAUNCH` property:

```
msiexec /i Nextcloud-x.y.z-x64.msi LAUNCH="1"
```

This option also removes the checkbox that lets users decide whether to launch the client during non-passive or non-quiet installations.

> “□□
>
> This option does not have any effect without a GUI.

No reboot after installation

The client schedules a reboot after installation to make sure the Explorer extension is correctly loaded or unloaded. If you are handling reboots yourself, you can set the `REBOOT` property:

```
msiexec /i Nextcloud-x.y.z-x64.msi REBOOT=ReallySuppress
```

This makes `msiexec` exit with error `ERROR_SUCCESS_REBOOT_REQUIRED` (3010). If your deployment tooling interprets this as an actual error and you want to avoid that, you may want to set `DO_NOT_SCHEDULE_REBOOT` instead:

```
msiexec /i Nextcloud-x.y.z-x64.msi DO_NOT_SCHEDULE_REBOOT="1"
```

Non-interactive account provisioning

The Nextcloud Desktop client supports non-interactive account provisioning from the command line.

This is intended for deployment automation and other managed-installation scenarios where an administrator wants to create a desktop-client account without requiring the user to go through the graphical setup wizard.

In a typical deployment workflow, you deploy the desktop client package first, then invoke the desktop client executable with provisioning parameters as part of your automation.

When the required parameters are provided, the client attempts to create an account and store it in the client's normal configuration so that subsequent launches behave as if the account had been added through the graphical user interface.

If account creation succeeds, the client exits with code `0`. If account creation fails, the client exits with code `1`.

Detailed status and failure information is written to the desktop client logs.

> “□□
>
> This workflow is distinct from interactive wizard preconfiguration. If you want to prefill or constrain the setup wizard instead of directly provisioning an account, see Interactive wizard preconfiguration.

Required parameters

The following parameters are required for non-interactive account provisioning:

`--userid`
The user ID to configure in the desktop client.

`--apppassword`
The app password to use for authentication.

App passwords should be used instead of a user's regular login password. See the server documentation for the login flow and app password generation process.

`--serverurl`
The base URL of the Nextcloud server to use for the account.

Optional parameters

`--localdirpath`
The local path to use for the sync folder.

If omitted, the desktop client chooses its normal default sync-folder location for the platform.

If this option is provided and the target directory already exists and is not empty, account provisioning fails.

If this option is provided and the directory does not yet exist, the client attempts to create it.

`--remotedirpath`
The remote path to sync.

If omitted, the default is `/` (the account root on the server).

Example: if the server contains folders such as `/Photos`, `/Documents`, and `/Music`, specifying `/Music` creates a sync connection for that remote subfolder instead of the entire account root.

`--isvfsenabled`
Controls whether the created sync connection should use virtual files.

Use `1` to enable virtual files and `0` to disable them.

Virtual file support depends on the client build, operating system, and runtime availability of the required virtual-file functionality. Only enable this option on platforms and builds where virtual files are supported.

`--confdir`
Overrides the configuration directory used by the client process.

This is a general client option rather than a provisioning-specific option. It is only needed if you intentionally want to use a non-default configuration directory.

If you use this option during provisioning, subsequent desktop-client launches must use the same configuration directory, or the provisioned account may not appear in later GUI sessions.

Behavior

When started with the required provisioning parameters, the desktop client attempts to:

1. validate the provided command-line parameters,
2. create the local sync folder if needed,
3. authenticate to the server using the provided app password,
4. verify server access, and
5. store the resulting account in the client's normal configuration.

On later launches, the configured account is available as a normal desktop-client account, provided the client is started with the same configuration context.

> “□□
>
> The desktop client does not mount the server into the local path like a network filesystem. It creates a sync connection. An empty local folder does not by itself mean that provisioning failed.

Common failure cases

Account provisioning can fail for reasons including:

- one or more required parameters are missing or invalid,
- an account for the same user and server already exists,
- the specified local sync folder already exists and is not empty,
- the specified local sync folder could not be created,
- the server URL is incorrect,
- the app password is invalid,
- the authenticated request is redirected unexpectedly,
- the server denies access to the configured account, or
- the server returns an invalid response to the authenticated WebDAV request.

In these cases, the client exits with code `1` and writes more detailed information to the logs.

Examples

Windows:

```
"C:\Program Files\Nextcloud\nextcloud.exe" ^
  --userid admin ^
  --apppassword Jliy12356785jxnHa2ZCiZ9MX48ncECwDso95Pq3a5HABjY34ZvhZiXrPfpKWUg7aOHAX5 ^
  --serverurl https://cloud.example.com ^
  --localdirpath "D:\Nextcloud-sync-folder" ^
  --remotedirpath /Music ^
  --isvfsenabled 1
```

Linux:

```
nextcloud \
  --userid admin \
  --apppassword Jliy12356785jxnHa2ZCiZ9MX48ncECwDso95Pq3a5HABjY34ZvhZiXrPfpKWUg7aOHAX5 \
  --serverurl https://cloud.example.com \
  --localdirpath "/home/admin/Nextcloud-sync-folder" \
  --remotedirpath /Music \
  --isvfsenabled 0
```

macOS:

```
nextcloud \
  --userid admin \
  --apppassword Jliy12356785jxnHa2ZCiZ9MX48ncECwDso95Pq3a5HABjY34ZvhZiXrPfpKWUg7aOHAX5 \
  --serverurl https://cloud.example.com \
  --localdirpath "/Users/admin/Nextcloud-sync-folder" \
  --remotedirpath /Music \
  --isvfsenabled 1
```

Recommendations

For reliable deployment automation:

- use app passwords rather than regular user passwords,
- ensure that the server URL is the correct base URL for the Nextcloud instance,
- use a new or empty local sync directory,
- validate that the target remote path exists and is accessible to the user,
- enable `--isvfsenabled` only on supported platforms and builds, and
- avoid `--confdir` unless you intentionally need a non-default configuration location.

Interactive wizard preconfiguration

If you want to automate the Account Setup Wizard so that users can skip entering the server URL and local sync folder path in the UI, you can use command-line parameters.

When you specify both, the desktop client's Account Setup Wizard will jump straight to opening a browser for account authentication or connection without the need to enter any of the connection details.

The local sync folder will also be selected to the one you specify instead of using the default path.

The following parameters are supported:

`--overridelocaldir`
Specify a local directory to be used in the account setup wizard.

Example: `/home/nextcloud-sync-folder`

`--overrideserverurl`
Specify a server URL to use for the force override in the account setup wizard.

Example: `https://cloud.example.com`

Behavior

These options affect the behavior of the interactive setup wizard. They do not directly create an account in the same way as non-interactive account provisioning.

Use this approach when you want the user to complete setup interactively, but want to prefill, constrain, or guide the process.

Examples

Windows:

```
"C:\Program Files\Nextcloud\nextcloud.exe" --overridelocaldir "D:/work/nextcloud-sync-folder"
--overrideserverurl https://cloud.example.com
```

Linux and macOS:

```
nextcloud --overridelocaldir "/home/<user>/nextcloud-sync-folder" --overrideserverurl
https://cloud.example.com
```

Important distinction from non-interactive provisioning

Use wizard preconfiguration when you still want the user to complete setup in the graphical interface.

Use non-interactive account provisioning when you want deployment automation to create the account directly.

In other words:

- non-interactive provisioning creates the account automatically;
- wizard preconfiguration influences the setup wizard, but still relies on interactive completion.

Summary

The Nextcloud Desktop client supports multiple administrator-facing deployment and setup workflows.

Choose the workflow that best matches your environment:

- use advanced Windows deployment options to control installation behavior on managed Windows systems,
- use non-interactive account provisioning to create desktop-client accounts automatically during deployment,
- use interactive wizard preconfiguration to guide users through a constrained or prefilled first-run setup process.

□□□ #3
□ cocobook-sync □□□ 2025-01-28 22:14:19 UTC
□ cocobook-sync □□□ 2026-04-19 23:29:30 UTC