

## Using Parameter Providers

Date published: 2019-06-26

Date modified: 2025-10-08



# Legal Notice

© Cloudera Inc. 2025. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

# Contents

|                                                                 |          |
|-----------------------------------------------------------------|----------|
| <b>What are Parameter Providers?</b>                            | <b>4</b> |
| <b>Example for using Parameter Providers</b>                    | <b>4</b> |
| Creating and configuring a Parameter Provider                   | 4        |
| Fetching Parameters                                             | 5        |
| Creating a Parameter Context from a Parameter Group             | 6        |
| Updating Parameter sensitivity                                  | 7        |
| Updating Parameter Context when the external source has changed | 8        |
| Using Parameter Context inheritance to combine Parameters       | 8        |

## What are Parameter Providers?

Parameter Providers is a new extension point that is added to the existing NiFi Parameter Context feature that can help to populate flow parameters on demand. Learn the features associated with Parameter Providers and how you can use them in your NiFi environment.

You can create Parameter Contexts from Parameters fetched from an external source. Using Parameter Providers allows automatic creation of Parameter Contexts from external sources, for example file-based Kubernetes secrets, environment variables, or HashiCorp Vault secrets engines.

Parameter Providers:

- Provide a feature in the Controller Settings that enable you to generate Parameter Contexts from external sources.
- Enable you to keep your provided Parameter Contexts up to date with the external source by running a Fetch Parameters operation.
- Provide an extension point for developing new custom Parameter Providers that can be deployed in a NiFi Archive (NAR).
- Provide a CLI command, `nifi fetch-params`, to fetch and apply parameters, allowing a scripted approach to keep Parameter Contexts up to date.

Parameter values are stored (encrypted, if sensitive) inside the flow. Parameter Providers are a mechanism that automate the creation of Parameter Contexts and facilitate keeping them updated. They do not replace the framework mechanism to resolve parameter values during Processor/Controller Service execution and they do not pull parameters directly from the external source at the time of usage in the flow.



**Note:** Parameter Providers do not have an automatic mechanism to refresh parameter values from the NiFi UI. Fetching and applying parameters is a potentially disruptive operation, since it can involve stopping and starting large portions of the flow. This kind of activity could be scripted using the CLI command `fetch-params`, but should be done with the potential for flow disruption in mind.

For more information about Parameter Contexts, see *Parameter Contexts in the Apache NiFi User Guide*.

### Related Information

[Parameter Contexts, Apache NiFi User Guide](#)

## Example for using Parameter Providers

In this example you can see how a Parameter Provider is created and configured, how it is used to fetch Parameters from an external source, and how a Parameter Context is created and updated using the Parameters fetched from the external source.

## Creating and configuring a Parameter Provider

Learn how to create and configure Parameter Providers through the NiFi Controller Settings.

### Procedure

1. Select Controller Settings from the top-right Global menu in the NiFi UI.  
The **NiFi Settings** dialog opens.
2. Go to the PARAMETER PROVIDERS tab.
3. Click **+** in the top-right corner of the NiFi Settings dialog to add a new Parameter Provider.

4. Select one of the listed types, in this example, FileParameterProvider, and click Add.

The Parameter Provider is created.

The FileParameterProvider created in this example lets you supply parameters in key-value files inside a Parameter Group directory.

5. Click  to edit it.
6. Enter the absolute path of the Parameter Groups on your file system in the Parameter Group Directories property.
7. Set the Parameter Value Encoding property to "Plain text".

**Configure Parameter Provider** | FileParameterProvider 1.18.0

SETTINGS

PROPERTIES

COMMENTS

✔

+

Required field

✔

+

| Property                    | Value             |
|-----------------------------|-------------------|
| Parameter Group Directories | ? /tmp/parameters |
| Parameter Value Byte Limit  | ? 256 B           |
| Parameter Value Encoding    | ? Plain text      |

8. Click APPLY to save the Parameter Provider configuration.

The Fetch Parameters icon becomes available if the path set for the Parameter Group Directories property is readable.

### Results

The Parameter Provider, in this example, FileParameterProvider, is created and configured.

### What to do next

You can now move on to fetching parameters. For details, see *Fetching parameters*.

### Related Information

[Fetching Parameters](#)

## Fetching Parameters

Learn how to fetch Parameters from an external source into a Parameter Group.

### Before you begin

You must have created and configured a Parameter Provider. For instructions, see [Creating and configuring a Parameter Provider](#).

### Procedure

1. If you do not have any files containing key-value pairs as parameters inside the directory set in the Parameter Group Directories property of FileParameterProvider, which in this example is the tmp/parameters directory, create some files and add some content.

```
$ print admin > /tmp/parameters/sys.admin.username && \
  print password > /tmp/parameters/sys.admin.password && \
```

```
print value > /tmp/parameters/sys.other
```

In this example, each file represents a Parameter in the parameters Parameter Group, and the contents of each file represents the parameter value.

2. Select Controller Settings from the top-right Global menu in the NiFi UI.  
The NiFi Settings dialog opens.
3. Go to the PARAMETER PROVIDERS tab.
4. Click the Fetch Parameters (down arrow) icon of FileParameterProvider.  
The Fetch Parameters dialog opens.

### Results

A Parameter Group named parameters is listed. This is the name of the directory where the Parameter Groups are stored on your file system. This group can be used to create a Parameter Context. The fetched parameters sys.admin.username, sys.admin.password, and sys.other are displayed.

Fetch Parameters

Name  
FileParameterProvider

Select To Configure A Parameter Group ?

Parameter Group Name ▾

parameters

☐ Create Parameter Context

Fetched Parameters ?

sys.admin.password  
sys.admin.username  
sys.other

Parameter Contexts To Create ?  
None

Parameter Contexts To Update ?  
None

Referencing Components ?  
None

### What to do next

You can now move on to *Creating a Parameter Context from a Parameter Group*.

### Related Information

[Creating a Parameter Context from a Parameter Group](#)

## Creating a Parameter Context from a Parameter Group

Learn how to create a Parameter Context from a Parameter Group that stores Parameters fetched from an external source using a Parameter Provider.

### Before you begin

You must have fetched parameters from an external source. For instructions, see [Fetching Parameters](#).

### Procedure

1. Select Controller Settings from the top-right Global menu in the NiFi UI.  
The NiFi Settings dialog opens.
2. Go to the PARAMETER PROVIDERS tab.
3. Click the Fetch Parameters (down arrow) icon of FileParameterProvider.  
The Fetch Parameters dialog opens.

#### 4. Select Create Parameter Context.

- A star appears next to the parameters Parameter Group. The star indicates that the Parameter Group has an associated Parameter Context.
- Checkboxes appear next to the Parameter names.
- The Parameter Context Name field appears.

Enter a name for your Parameter Context. In this example, it is My Parameters.

### Fetch Parameters

Name  
FileParameterProvider

Select To Configure A Parameter Group ⓘ  

Parameter Group Name ▲

parameters ★

☒ Create Parameter Context

Parameter Context Name  

My Parameters

Select Parameters To Be Set As Sensitive ⓘ  

☒ SELECT ALL
☐ DESELECT ALL

Parameter Name ▲

☒ sys.admin.password  
☐ sys.admin.username  
☐ sys.other

Parameter Contexts To Create ⓘ  
parameters

Parameter Contexts To Update ⓘ  
None

Referencing Components ⓘ  
None

#### 5. Select the Parameters that you want to set as sensitive.

In this example, only the sys.admin.password Parameter needs to be set to sensitive.

#### 6. Click APPLY to create the Parameter Context.

#### 7. Click Close in the Fetch Parameters dialog.

### Results

- The newly created Parameter Context appears under the SETTINGS tab of the Configure Parameter Provider dialog.
- The Parameters appear under the PARAMETERS tab of the Update Parameter Context dialog. Only the values of the non-sensitive Parameters can be read. The Parameters cannot be edited. The Parameter Provider can add, remove, or update Parameters in this Parameter Context.

### What to do next

- You can view the new Parameter Context in the NiFi Parameter Contexts list by clicking its name.
- You can view the details of the Parameter Context in the Update Parameter Context dialog by clicking .
- You can return to the NiFi Settings by clicking the right arrow icon in the listing.

## Updating Parameter sensitivity

You may need to update the sensitivity setting of Parameters of a Parameter Contexts over time.

### Before you begin

- You have created a Parameter Context from a Parameter Provider.
- If the Parameter Context, in this example, "My Parameters", is assigned to a group, and that group has a component that references the sys.other parameter, you cannot update its sensitivity. For this, ensure that you remove any references to it.

You must have created a Parameter Context from a Parameter Provider.

### Procedure

1. Select Controller Settings from the top-right Global menu in the NiFi UI.  
The NiFi Settings dialog opens.
2. Go to the PARAMETER PROVIDERS tab.
3. Click the Fetch Parameters (down arrow) icon of FileParameterProvider.  
The Fetch Parameters dialog opens.
4. Select the Parameter whose sensitivity you want to change.
5. Click APPLY to update the Parameter Context.

### Related Information

[Creating a Parameter Context from a Parameter Group](#)

## Updating Parameter Context when the external source has changed

The Parameters that you fetched from an external source may change in that external source over time. In such cases you need to fetch the Parameters again using the Parameter Providers.

### About this task

Parameters are not automatically synchronized with the external source. To update any linked Parameter Contexts, you can fetch the parameters again, and you need to specify the sensitivity of any new parameters.

### Before you begin

There must be changes to your external source. To simulate the change in the example, delete the sys.other parameter, update the value of the sys.admin.username parameter, and add a new parameter new-parameter.

```
$ rm /tmp/parameters/sys.other && \  
  print admin2 > /tmp/parameters/sys.admin.username && \  
  print test > /tmp/parameters/new-parameter
```

### Procedure

1. Select Controller Settings from the top-right Global menu in the NiFi UI.  
The NiFi Settings dialog opens.
2. Go to the PARAMETER PROVIDERS tab.
3. Click the Fetch Parameters (down arrow) icon of FileParameterProvider.  
The Fetch Parameters dialog opens. An asterisk appears next to newly fetched and changed Parameters. To view detailed information about the change in the Parameter, hover over the asterisk.
4. Set sensitivity for the newly fetched Parameters.
5. Click APPLY to update the Parameter Context.
6. Click Close in the Fetch Parameters dialog.

### Results

The changes that took place in the parameters in the external source are now reflected in the Parameter Context. In this example, sys-other parameter is removed, new-parameter is added, and the value of sys.admin.username is changed. You can view the changes by clicking the Fetch Parameters (down arrow) icon.

## Using Parameter Context inheritance to combine Parameters

You can use Parameter Context inheritance to combine Parameters from different Parameter Providers within one Parameter Context.

It is good practice to create separate Parameter Providers for different sources, and then compose the Parameter Contexts through inheritance. This allows sensitive parameters to originate from a secrets manager, like HashiCorp Vault, and non-sensitive parameters to originate from other sources like Environment Variables or the file system. In the following simple example, the EnvironmentVariableParameterProvider contains non-sensitive Parameters and the FileParameterProvider contains sensitive Parameters. Both Parameter Providers are added as inherited contexts to a new Parameter Context, “Parameters”. For details on inherited Parameter Contexts, see *What is Parameter Context inheritance?*.

Update Parameter Context

SETTINGSPARAMETERSINHERITANCE

Available Parameter Contexts ⓘ

test

Selected Parameter Context ⓘ

Environment Variables✕

File Parameters✕

The new Parameter Context inherits Parameters from both sources:

|                    |                     |   |
|--------------------|---------------------|---|
| JENV_FORCEJAVAHOME | true                | → |
| JENV_FORCEJDKHOME  | true                | → |
| JENV_LOADED        | 1                   | → |
| JENV_SHELL         | zsh                 | → |
| new-parameter      | Sensitive value set | → |
| sys.admin.password | Sensitive value set | → |
| sys.admin.username | Sensitive value set | → |

**Related Information**  
[What is Parameter Context inheritance?](#)