Cloudera Flow Management 2.1.7

Preparing for upgrade to NiFi 2

Date published: 2019-06-26
Date modified: 2024-10-02

CLOUDERA

https://docs.cloudera.com/

Legal Notice

Contents

Apache NiFi and NiFi in Cloudera Flow Management

NiFi 2 marks a major version of Apache NiFi, introducing several exciting new features, including a powerful Python API for developing native NiFi components using Python. While this version brings significant enhancements, the transition from NiFi 1.x to NiFi 2.x comes with numerous breaking changes, potentially making the upgrade process complex and challenging.

Cloudera has made the decision to move a large number of components downstream that were removed from the open-source Apache NiFi project. These changes aim to provide better support for Cloudera customers by aligning with specific Cloudera technologies and minimizing disruptive changes.

For example:

- **Hive components**:

  NiFi in Cloudera Flow Management includes Hive components built against Cloudera's version of Hive, which has diverged from Apache Hive, ensuring compatibility with Cloudera's ecosystem.
- **Kafka components**:

  Cloudera has retained Kafka 2.6 components in Cloudera Flow Management to facilitate smoother transitions from legacy Kafka components to the new Kafka architecture based on controller services, reducing the risk of breaking changes.

Breaking changes in NiFi 2

Learn about the breaking changes Apache NiFi 2 will bring to Cloudera Flow Management.

Flow design level breaking changes

Moved components

In NiFi 2, some components have been relocated within the Apache NiFi repository, resulting in changes to the bundle coordinates for the associated NAR files.

Unfortunately, no pre-upgrade actions can fully prevent these breaking changes. You will need to update the flow.json.gz file with new coordinates. Cloudera's upcoming NiFi Migration Tooling is designed to automate as many changes as possible to help the upgrade process. Some changes will still require manual handling, so it is highly recommended to run the pre-upgrade check script to identify any potential issues or impacted components before proceeding with the upgrade.

JoltTransformJSON processor

For Previous coordinates

```
…
            "type": "org.apache.nifi.processors.standard.JoltT
ransformJSON",
            "bundle": {
              "group": "org.apache.nifi",
              "artifact": "nifi-standard-nar",
              "version": "1.27.0"
            }
…
```

For New coordinates

```
…
```

```
            "type": "org.apache.nifi.processors.jolt.JoltTrans
formJSON",
            "bundle": {
              "group": "org.apache.nifi",
              "artifact": "nifi-jolt-nar",
              "version": "2.0.0"
            }
…
```

JoltTransformRecord processor

For Previous coordinates

```
…
            "type": "org.apache.nifi.processors.jolt.record.Jo
ltTransformRecord",
            "bundle": {
              "group": "org.apache.nifi",
              "artifact": "nifi-jolt-record-nar",
              "version": "1.27.0"
            }
…
```

For New coordinates

```
…
            "type": "org.apache.nifi.processors.jolt.JoltTrans
formRecord",
            "bundle": {
              "group": "org.apache.nifi",
              "artifact": "nifi-jolt-nar",
              "version": "2.0.0"
            }
…
```

FileParameterProvider renamed to KubernetesSecretParameterProvider

For Previous coordinates

```
…
            "type": "org.apache.nifi.parameter.FileParameterPr
ovider",
            "bundle": {
              "group": "org.apache.nifi",
              "artifact": "nifi-standard-nar",
              "version": "1.27.0"
            }
…
```

For New coordinates

```
…
            "type": "org.apache.nifi.parameter.KubernetesSecre
tParameterProvider",
            "bundle": {
              "group": "org.apache.nifi",
              "artifact": "nifi-standard-nar",
              "version": "2.0.0"
            }
```

```
...
```

Removed key components

- Kafka processors:

  All Kafka processors in Apache NiFi have been removed and got replaced by new components using a controller service-based approach. This change is a significant breaking change, as it does not allow for a non-breaking upgrade. To ease this transition, Cloudera has preserved the Kafka 2.6 processors without altering the bundle coordinates. However, adjustments to the Kerberos configuration will still be necessary (see the details below). This approach provides time to transition to the new Kafka components while on NiFi 2.
- Hive components:

  All Hive-related components in Apache NiFi have been removed. Cloudera has introduced specific components downstream to align with the Hive version distributed as part of CDP. You will need to perform proper bundle coordinate updates in the flow.json.gz file to migrate to these new components. Both the old and new components are available in the latest NiFi 1.x releases, so you are advised to switch to the new components while still using NiFi 1.x.

Components with Kerberos configuration changes

Kerberos authentication in NiFi requires presenting a Kerberos credential which can be in one of the following forms:

- Principal + Keytab: The keytab, stored on disk, contains the client's secret key. This credential is used by the application to authenticate and obtain the Ticket Granting Ticket (TGT).
- Principal + Password: The client's secret key is derived from a password, so no keytab is stored on disk. Otherwise, this method functions similarly to the keytab-based approach.
- Principal + Ticket cache: The TGT must be acquired externally and stored in a ticket cache, which the application uses. The application itself is unaware of the keytab or password and is not responsible for handling TGT acquisition.

Historically, NiFi supported the following Kerberos configuration options:

- Kerberos Principal + Kerberos Keytab: Component-level properties supporting keytab-based credential type
- Kerberos Credentials Service: A property referencing the KerberosCredentialsService controller service interface with the following implementation: KeytabCredentialsService for keytab-based credential type
- Kerberos Principal + Kerberos Password: Component-level properties supporting password-based credentials
- Kerberos User Service: A property referencing the KerberosUserService controller service interface, with implementations for all credential types:
  - KerberosKeytabUserService for keytab-based credentials
  - KerberosPasswordUserService for password-based credentials
  - KerberosTicketCacheUserService for ticket cache-based credentials.

In NiFi 2, only the Kerberos User Service is retained because it can accommodate all credential types (keytab, password, ticket cache) with a single property on the component. The other configuration options have been removed in NiFi 2. For more information, see NIFI-13510.

Affected components in Cloudera Flow Management 2.1.7:

The table below lists the affected components and their legacy Kerberos properties. It also indicates whether these components are planned to be available in Cloudera Flow Management 4.x and provides additional comments on their migration.

Component Group	Component Type	Component Artifact	Component Name	(1) Kerberos Principal + Kerberos Keytab	(2) Kerberos Credentials Service	(3) Kerberos Principal + Kerberos Password	(4) Kerberos User Service	Future availability in CFM 4	Comment
	org.apache.nifi.accumulo.controllerservices.AccumuloService	nifi-accumulo-services-nar	AccumuloService	x	x	x	NO		
	org.apache.nifi.dbcp.DBCPConnectionPool	nifi-dbcp-service-nar	DBCPConnectionPool	x		x	x	YES	
	org.apache.nifi.dbcp.HadoopDBCPConnectionPool	nifi-hadoop-dbcp-service-nar	HadoopDBCPConnectionPool	x		x	x	YES	
	org.apache.nifi.schemaregistry.hortonworks.HortonworksSchemaRegistry	nifi-hwx-schema-registry-nar	HortonworksSchemaRegistry	x				NO	Replaced by ClouderaSchemaRegistry
	org.apache.nifi.controller.kudu.KuduLookupService	nifi-kudu-nar	KuduLookupService	x				YES	
	org.apache.nifi.controller.livy.LivySessionController	nifi-livy-nar	LivySessionController	x				YES	
	org.apache.nifi.processors.kudu.PutKudu	nifi-kudu-nar	PutKudu		x	x	x	YES	
	org.apache.nifi.atlas.reporting.ReportLineageToAtlas	nifi-atlas-nar	ReportLineageToAtlas	x				YES	
CDP Object Store processors	org.apache.nifi.processors.hadoop.ListCDPObjectStore	nifi-cdf-objectstore-nar	ListCDPObjectStore	x	x			YES	
	org.apache.nifi.processors.hadoop.FetchCDPObjectStore	nifi-cdf-objectstore-nar	FetchCDPObjectStore					YES	
	org.apache.nifi.processors.hadoop.PutCDPObjectStore	nifi-cdf-objectstore-nar	PutCDPObjectStore					YES	
	org.apache.nifi.processors.hadoop.DeleteCDPObjectStore	nifi-cdf-objectstore-nar	DeleteCDPObjectStore					YES	
Hadoop processors	org.apache.nifi.processors.hadoop.ListHDFS	nifi-hadoop-nar	ListHDFS	x	x	x	x	YES	
	org.apache.nifi.processors.hadoop.FetchHDFS	nifi-hadoop-nar	FetchHDFS					YES	

	org.apache.nifi.processors.hadoop.PutHDFS	nifi-hadoop-nar	PutHDFS						YES	
	org.apache.nifi.processors.hadoop.DeleteHDFS	nifi-hadoop-nar	DeleteHDFS						YES	
	org.apache.nifi.processors.hadoop.GetHDFS	nifi-hadoop-nar	GetHDFS						YES	
	org.apache.nifi.processors.hadoop.MoveHDFS	nifi-hadoop-nar	MoveHDFS						YES	
	org.apache.nifi.processors.hadoop.inotify.GetHDFSEvents	nifi-hadoop-nar	GetHDFSEvents						YES	
	org.apache.nifi.processors.hadoop.GetHDFSFileInfo	nifi-hadoop-nar	GetHDFSFileInfo						YES	
	org.apache.nifi.processors.hadoop.GetHDFSSequenceFile	nifi-hadoop-nar	GetHDFSSequenceFile						YES	
	org.apache.nifi.processors.hadoop.CreateHadoopSequenceFile	nifi-hadoop-nar	CreateHadoopSequenceFile						YES	
	org.apache.nifi.processors.parquet.FetchParquet	nifi-parquet-nar	FetchParquet						YES	
	org.apache.nifi.processors.parquet.PutParquet	nifi-parquet-nar	PutParquet						YES	
	org.apache.nifi.processors.orc.PutORC	nifi-hive3-nar	PutORC						NO	Replaced by PutClouderaORC
HBase services	org.apache.nifi.hbase.HBase_1_1_2_ClientService	nifi-hbase_1_1_2-client-service-nar	HBase_1_1_2_ClientService			x	x		NO	HBase_2_ClientService should be used instead
	org.apache.nifi.hbase.HBase_2_ClientService	nifi-hbase_2-client-service-nar	HBase_2_ClientService						YES	
Hive components	org.apache.nifi.dbcp.hive.HiveConnectionPool	nifi-hive-nar	HiveConnectionPool			x	x		NO	Replaced by ClouderaHiveConnectionPool
	org.apache.nifi.dbcp.hive.Hive3ConnectionPool	nifi-hive3-nar	Hive3ConnectionPool						NO	Replaced by ClouderaHiveConnectionPool
	org.apache.nifi.processors.hive.PutHiveStreaming	nifi-hive-nar	PutHiveStreaming						NO	Replaced by PutClouderaHiveStreaming
	org.apache.nifi.processors.hive.PutHive3Streaming	nifi-hive3-nar	PutHive3Streaming			x	x		NO	Replaced by PutClouderaHiveStreaming

Kafka_1_0 processors	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-1-0-nar	ConsumeKafka_1_0	x				NO	ConsumeKafka_2_6 should be used instead
	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-1-0-nar	PublishKafka_1_0					NO	PublishKafka_2_6 should be used instead
	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-1-0-nar	ConsumeKafkaRecord_1_0	x				NO	ConsumeKafkaRecord_2_6 should be used instead
	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-1-0-nar	PublishKafkaRecord_1_0					NO	PublishKafkaRecord_2_6 should be used instead
Kafka_2_0 processors	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-2-0-nar	ConsumeKafka_2_0	x	x			NO	ConsumeKafka_2_6 should be used instead
	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-2-0-nar	PublishKafka_2_0					NO	PublishKafka_2_6 should be used instead
	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-2-0-nar	ConsumeKafkaRecord_2_0					NO	ConsumeKafkaRecord_2_6 should be used instead
	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-2-0-nar	PublishKafkaRecord_2_0					NO	PublishKafkaRecord_2_6 should be used instead
Kafka_2_6 processors	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-2-6-nar	ConsumeKafka_2_6	x	x		x	YES	
	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-2-6-nar	PublishKafka_2_6					YES	
	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-2-6-nar	ConsumeKafkaRecord_2_6					YES	
	org.apache.nifi.processors.kafka.pubsub	nifi-kafka-2-6-nar	PublishKafkaRecord_2_6					YES	
Kafka2CDP processors	org.apache.nifi.processors.kafka.pubsub	nifi-cdf-kafka-2-nar	ConsumeKafka2CDP	x			x	YES	
	org.apache.nifi.processors.kafka.pubsub	nifi-cdf-kafka-2-nar	PublishKafka2CDP					YES	
	org.apache.nifi.processors.kafka.pubsub	nifi-cdf-kafka-2-nar	ConsumeKafkaRecord2CDP					YES	

	org.apache.nifi.processors.kafka.pubsub.PublishKafkaRecord_2_6	nifi-cdf-kafka-2-nar	PublishKafkaRecord2CDP					YES	
Solr processors	org.apache.nifi.processors.solr.GetSolr	nifi-solr-nar	GetSolr		x	x	x	YES	
	org.apache.nifi.processors.solr.QuerySolr	nifi-solr-nar	QuerySolr					YES	
	org.apache.nifi.processors.solr.PutSolrContentStream	nifi-solr-nar	PutSolrContentStream					YES	
	org.apache.nifi.processors.solr.PutSolrRecord	nifi-solr-nar	PutSolrRecord					YES	

Flow migration for Kerberos configuration changes

To ensure compatibility with NiFi 2, you will need to migrate your flow by creating a new Kerberos UserService controller service based on the old controller service or component level properties. This process involves transitioning from legacy component-level properties to the updated service and clearing outdated configurations.

Migration steps for **Kerberos Principal + Kerberos Keytab**:

1. Create service: Set up a KerberosKeytabUserService using the existing Kerberos Principal and Kerberos Keytab component-level properties.
2. Update reference: Link the new service to the component's Kerberos User Service property.
3. Remove legacy properties: Clear the old property values from Kerberos Principal and Kerberos Keytab.

Migration steps for **Kerberos Credentials Service**:

1. Create service: Set up a KerberosKeytabUserService based on the properties from the KeytabCredentialsService.
2. Update reference: Point the component's Kerberos User Service property to the new service.
3. Remove legacy service: Clear the old Kerberos Credentials Service reference property and delete the outdated service.

Migration steps for **Kerberos Principal + Kerberos Password**:

1. Create service: Set up a KerberosPasswordUserService using the Kerberos Principal and Kerberos Password component-level properties.
2. Update reference: Link the new service to the component’s Kerberos User Service property.
3. Remove legacy properties: Clear the old property values from Kerberos Principal and Kerberos Password.

Components requiring initial code-level changes

For certain components, the Kerberos User Service property is not yet available. These components will require an initial code-level update before migration.

- Kafka_1_0 processors
- Kafka_2_0 processors
- CDPObjectStore processors
- ReportLineageToAtlas
- KuduLookupService
- LivySessionController

Migration to Cloudera-specific components

As part of the migration, some components will transition to new, Cloudera-specific types. The Kerberos property migration should be included in this transition.

- HortonworksSchemaRegistry => ClouderaSchemaRegistry
- Hive[3]ConnectionPool => ClouderaHiveConnectionPool
- PutHive[3]Streaming => PutClouderaHiveStreaming

Scripted components

In NiFi 2, support for certain languages in scripted components has been removed. The affected languages include ECMAScript, Lua, Ruby, and Python. Cloudera recommends to switch to Groovy or leverage the new Python API feature for developing processors. The following components are impacted by this change:

- org.apache.nifi.processors.script.ExecuteScript
- org.apache.nifi.processors.script.InvokeScriptedProcessor
- org.apache.nifi.processors.script.ScriptedFilterRecord
- org.apache.nifi.processors.script.ScriptedPartitionRecord
- org.apache.nifi.processors.script.ScriptedTransformRecord
- org.apache.nifi.processors.script.ScriptedValidateRecord
- org.apache.nifi.lookup.script.ScriptedLookupService
- org.apache.nifi.record.script.ScriptedReader
- org.apache.nifi.record.script.ScriptedRecordSetWriter
- org.apache.nifi.record.sink.script.ScriptedRecordSink
- org.apache.nifi.lookup.script.SimpleScriptedLookupService
- org.apache.nifi.reporting.script.ScriptedReportingTask

Important: As of now, the Python API in NiFi 2 is limited to creating processors. Writing controller services or reporting tasks using Python is not supported.

Custom components

If your NiFi environment includes custom components or NARs developed for NiFi 1.x, they are unlikely to be compatible with NiFi 2. To ensure compatibility, you must update your dependencies to align with NiFi 2 and rebuild your NARs using Java 21. This update is essential for a successful transition to the new version.

Flow controller level breaking changes

Transition from variables to parameters

Variables and the variable registry are removed in NiFi 2 due to their inherent limitations, such as requiring expression language support to reference a variable and the inability to store sensitive values. You can use parameter contexts instead, which have been significantly enhanced over recent years. For example, the addition of the Parameter Context Provider allows for sourcing parameter values from external stores (like HashiCorp Vault or cloud provider vaults).

This change is one of the most impactful in NiFi 2, which will require rework on existing data flows. However, it also presents an opportunity to optimize the organization of parameters, allowing you to split them into multiple parameter contexts and use inheritance when sharing parameters across different use cases.

Cloudera will provide automated tools within the NiFi Migration Tooling to assist with transitioning from variables to parameters.

Removal of XML templates

The concept of XML templates is being phased out in NiFi 2. Historically, these templates were stored in memory as well as in the flow definition files (flow.xml.gz and flow.json.gz). This caused significant issues for some NiFi users, especially those managing numerous large templates with

thousands of components. Removing templates from NiFi will enhance stability and reduce memory usage.

If you use templates in your NiFi 1.x clusters, you should export your existing templates as JSON definitions or version them into a NiFi Registry instance to prepare for this change. Using NiFi Registry is the recommended best practice for version control, sharing, and reusing flow definitions.

If your template is a process group:

1. Drag and drop the template onto the canvas.
2. Right-click it, and choose to export it as a flow definition (JSON file) or start version control in your NiFi Registry, if you have one configured.

If your template is not a process group (just a flow with components):

1. Drag and drop a process group onto the canvas.
2. Go into that process group and drag and drop your template within that process group.
3. Go back to the parent process group containing your template and export it as a flow definition or start version control on it.

Cloudera will provide automated tools within the NiFi Migration Tooling to help manage the migration of templates.

Discontinuation of event driven thread pool

The Event Driven Scheduling Strategy, which was available for some processors in previous versions of NiFi, is being removed in NiFi 2. This feature was experimental and did not demonstrate significant performance improvements.

If you are using this scheduling strategy, you will need to update your components to use the time driven scheduling strategy instead. You can identify components using the Event Driven strategy by searching for “event” in the NiFi search bar.

NiFi framework breaking changes in NiFi 2

Java 21 compatibility

Java 21 is the minimum Java version required with NiFi 2. While most versions of NiFi 1.x may work with Java 21, it is not officially supported to run NiFi 1.x with Java 21 for extended periods in preparation for the upgrade. The transition to Java 21 should happen as part of the upgrade process to NiFi 2. You have to ensure that your environment is ready for Java 21 before initiating the upgrade.

Transition from flow.xml.gz to flow.json.gz

In NiFi 2, flow.xml.gz, which has been a cornerstone of flow configuration storage, is completely phased out and replaced by flow.json.gz. While the two files have coexisted in many NiFi 1.x releases, NiFi 2 will exclusively use JSON-based flow representations. Before upgrading, it is recommended to back up your flow.json.gz file. During the upgrade, only work with this JSON file if any updates are needed.

Repository encryption removal

The ability to encrypt repositories has been removed in NiFi 2.

NiFi Toolkit changes

The NiFi Toolkit will also undergo changes in NiFi 2, with some features being removed.

Dynamic parameter retrieval with parameter providers

In NiFi 2, the values associated with parameters retrieved by parameter providers are no longer stored within the flow.json.gz file. Instead, these values are retrieved on-demand or during NiFi's startup and are only stored in memory. This change enhances security and reduces the size of flow files, but it also means that parameter values will need to be available from their source each time NiFi is started or when the parameters are accessed. It is important

to ensure that the external sources for these parameters are reliable and accessible to avoid disruptions during NiFi operations. For more information, see NIFI-13560.

Pre-upgrade check for NiFi

Starting with Cloudera Flow Management 2.1.7.1000 (Service Pack 1), you can run a pre-upgrade check to help you prepare for the transition to NiFi 2.

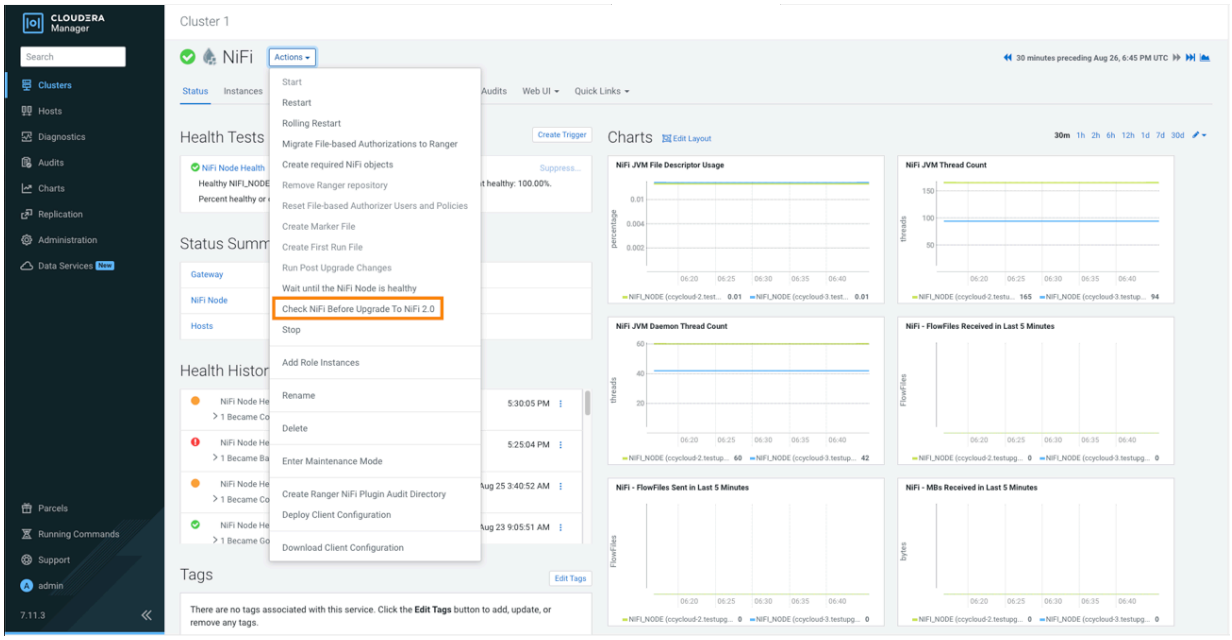
The pre-upgrade check script is specifically designed to highlight necessary steps and make you aware of any breaking changes that will not be automatically handled by Cloudera's NiFi Migration Tooling. Its primary purpose is to ensure a smooth upgrade by informing you of potential issues before you start your NiFi 2 upgrade.

Pre-upgrade check for NiFi in Cloudera Manager

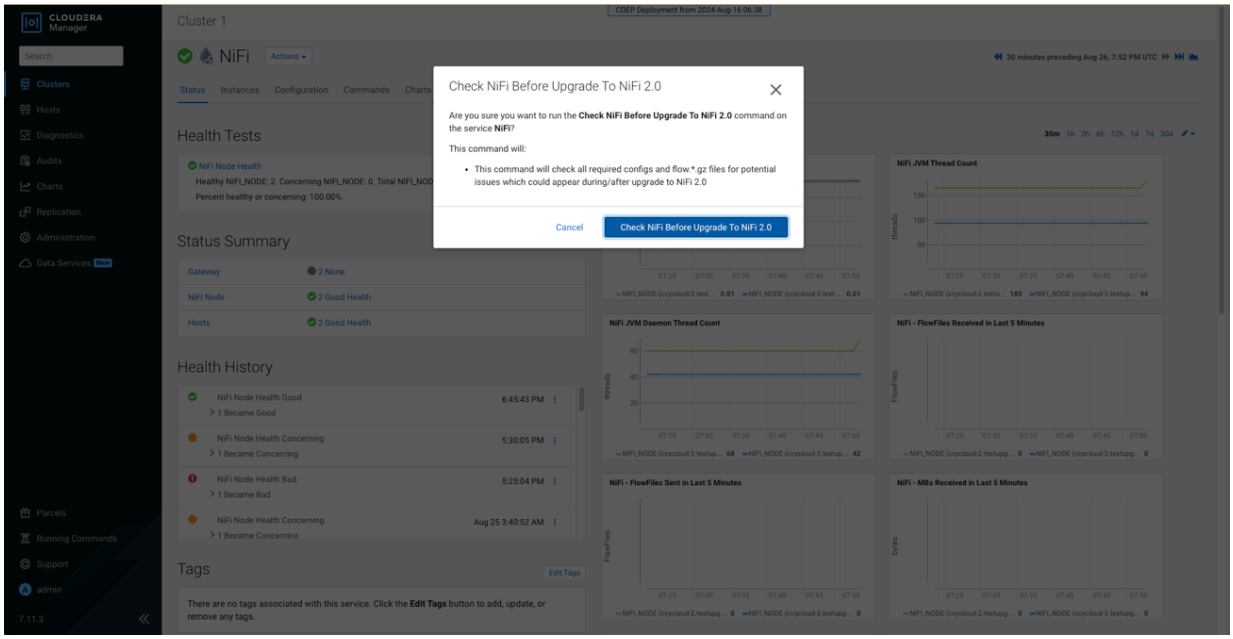
Cloudera Manager allows you to easily run a pre-upgrade check for NiFi. This process helps identify any potential issues before upgrading. Follow the below steps to perform the pre-upgrade check directly from Cloudera Manager.

Procedure

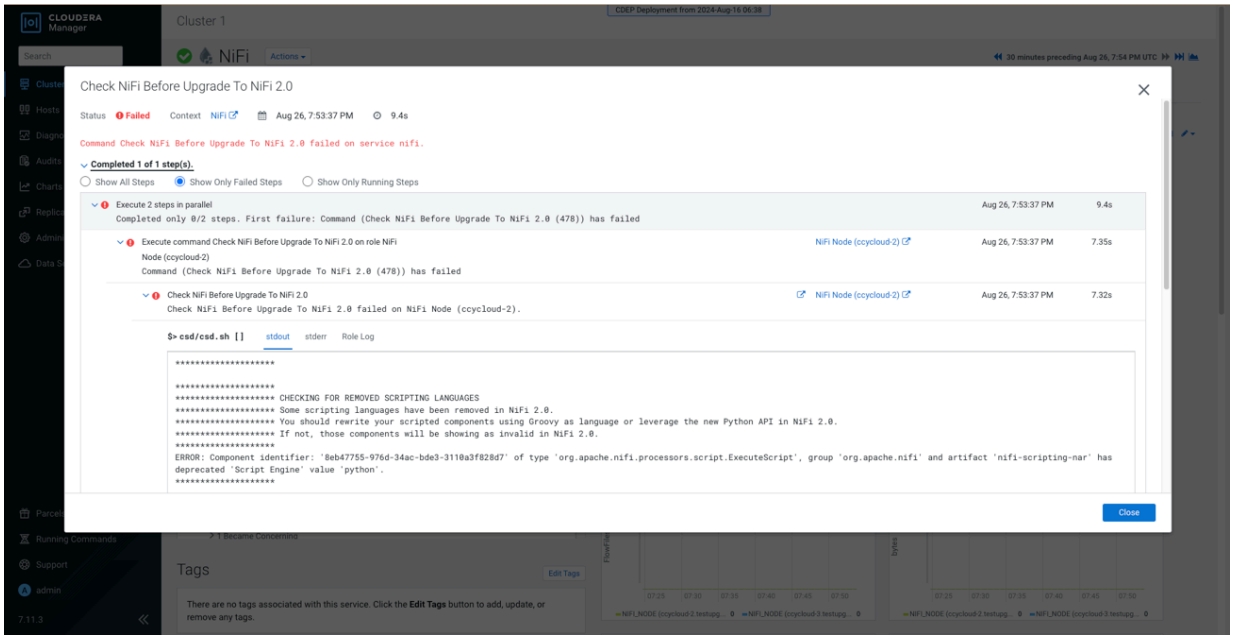
1. From Cloudera Manager, click the Clusters tab in the left-hand navigation.
2. Click NiFi in the list of services to display the NiFi service page.
3. Click the Actions drop-down next to the service name and select the Check NiFi Before Upgrade To NiFi 2.0 option.



4. Click Check NiFi Before Upgrade To NiFi 2.0 again.



The script runs across all NiFi nodes, providing a comprehensive list of items to address before starting an upgrade to NiFi 2.



5. Review this list thoroughly to ensure a smooth upgrade process.

Example

Here is an example of the output:

```
Tue Sep 24 19:37:22 UTC 2024
JAVA_HOME=/usr/java/jdk1.8.0_232-cloudera
Using -XX:OnOutOfMemoryError=/opt/cloudera/cm-agent/service/common/killpa
rent.sh as CSD_JAVA_OPTS
Using /var/run/cloudera-scm-agent/process/60-nifi-NIFI_NODE-PreUpgradeCheck
 as conf dir
Using scripts/service-commands/pre_upgrade_check.sh as process script
```

```
CONF_DIR=/var/run/cloudera-scm-agent/process/60-nifi-NIFI_NODE-PreUpgrade
Check
CMF_CONF_DIR=
Script pre_upgrade_check.sh started work!
CMF_PACKAGE_DIR = /opt/cloudera/cm-agent/service
NIFI_WORKING_DIRECTORY = /var/lib/nifi
Path to csv file = /var/run/cloudera-scm-agent/process/60-nifi-NIFI_NODE-P
reUpgradeCheck/scripts/common-scripts/resources/CFM4-Deprecated-components.c
sv
Path to flow.json.gz = /var/lib/nifi/flow.json.gz
Path to bootstrap.conf = /var/lib/nifi/config_backup/bootstrap.conf
Custom java home =
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Check Before Upgrade to CFM with NiFi 2.0
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Using Python 3 or newer version (3.8.12)
Script pre_upgrade_check.py started work!
Passed arguments:
Path to json flow file = /var/lib/nifi/flow.json.gz
Path to deprecated components CSV file = /var/run/cloudera-scm-agent/proce
ss/60-nifi-NIFI_NODE-PreUpgradeCheck/scripts/common-scripts/resources/CFM4-D
eprecated-components.csv
Path to components with kerberos properties CSV file = /var/run/cloudera-sc
m-agent/process/60-nifi-NIFI_NODE-PreUpgradeCheck/scripts/common-scripts/res
ources/NiFi-1.x-components-with-legacy-Kerberos-properties.csv
Path to bootstrap conf = /var/lib/nifi/config_backup/bootstrap.conf
********************
Validating arguments:
You are running pre upgrade check before upgrade to NiFi 2.0 for NiFi mana
ged by Cloudera Manager.
Path to flow.json.gz file is valid!
Path to deprecated components CSV file is valid!
Path to components with kerberos properties CSV file is valid!
Path to bootstrap.conf file is valid!
********************
Parsing bootstrap.conf:
********************

Reading and parsing /var/run/cloudera-scm-agent/process/60-nifi-NIFI_NODE-
PreUpgradeCheck/scripts/common-scripts/resources/CFM4-Deprecated-components.
csv:
********************

Reading and parsing /var/run/cloudera-scm-agent/process/60-nifi-NIFI_NODE-
PreUpgradeCheck/scripts/common-scripts/resources/NiFi-1.x-components-with-l
egacy-Kerberos-properties.csv:
********************
Reading and parsing flow.json.gz:
********************

********************
******************** CHECKING THAT UPGRADE PATH IS SUPPORTED
******************** (It is required to be on CFM 2.1.7+ before upgrading to
 CFM 4.0.0+)
********************
Get current CFM version from bootstrap.conf and check it's valid to upgrade
to CFM with NiFi 2.0 (CFM-4.0.0.0 or newer)
Property value for 'working.dir' = /opt/cloudera/parcels/CFM-2.1.7.1000-42/N
IFI
Current CFM version could be upgraded to CFM with NiFi 2.0 (CFM-4.0.0.0 or n
ewer).
```

```
********************
********************
******************** CHECKING THAT JAVA 21 IS USED
******************** (Java 21 is the minimum Java version required for NiFi
 and NiFi Registry 2.0)
********************
Get 'java' property value from bootstrap.conf
Property value for 'java' = java
Getting java home from JAVA_HOME in env variables.
Executing command: /usr/java/jdk1.8.0_232-cloudera/bin/java -version 2>&1 |
 awk -F[\"\.] -v OFS=. 'NR==1{print $2}'
Command output: 1

WARN: Your current Java version is too low for upgrading to NiFi 2.0 (CFM-
4.0.0.0). Please install Java version 21 on each host with NiFi and follow t
he upgrade documentation for NiFi 2.0 (CFM-4.0.0.0). NOTE: Do not change the
 JDK for the current NiFi. It should only be changed during the upgrade to N
iFi 2.0 (CFM-4.0.0.0).

********************
******************** CHECKING FOR REMOVED COMPONENTS
******************** Some components have been removed in NiFi 2.0.
******************** You should switch to the recommended option before up
grading.
******************** If not, those components will be showing as ghost comp
onents in NiFi.
********************
ERROR: Component identifier: '8d74d462-7c52-35d1-ad1b-665696503ea2' of type
 'org.apache.nifi.processors.gettcp.GetTCP', group 'org.apache.nifi' and art
ifact 'nifi-tcp-nar' is no longer supported.
ERROR: Component identifier: 'f78dba93-e6ec-38d7-b5e4-fdbcd09fd831' of type
 'org.apache.nifi.processors.pulsar.pubsub.PublishPulsar', group 'io.streamn
ative.connectors' and artifact 'nifi-pulsar-nar' is removed, but it is plann
ed to be reintroduced later. Please check the available CFM-4.x releases.
ERROR: Component identifier: '2e2edf20-3efc-388a-ac96-943b717e5252' of type
 'org.apache.nifi.processors.twitter.GetTwitter', group 'org.apache.nifi' a
nd artifact 'nifi-social-media-nar' is no longer supported. Please consider
using 'ConsumeTwitter' instead.
ERROR: Component identifier: 'a0766b6b-140e-31a8-a1fb-80f10459f870' of type
'org.apache.nifi.kerberos.KeytabCredentialsService', group 'org.apache.nifi'
 and artifact 'nifi-kerberos-credentials-service-nar' is no longer supported
. Please consider using 'KerberosKeytabUserService' instead.
********************

********************
******************** CHECKING FOR REMOVED SCRIPTING LANGUAGES
******************** Some scripting languages have been removed in NiFi 2.0.
******************** You should rewrite your scripted components using Gr
oovy as language or leverage the new Python API in NiFi 2.0.
******************** If not, those components will be showing as invalid in
 NiFi 2.0.
********************
ERROR: Component identifier: 'e0334fb0-738b-3000-bdfc-fee3e59f539d' of type
'org.apache.nifi.processors.script.InvokeScriptedProcessor', group 'org.apac
he.nifi' and artifact 'nifi-scripting-nar' has deprecated 'Script Engine' va
lue 'ruby'.
ERROR: Component identifier: 'e9d304ef-6c58-39ea-aa27-c01a6d5718fe' of type
'org.apache.nifi.processors.script.ExecuteScript', group 'org.apache.nifi' a
nd artifact 'nifi-scripting-nar' has deprecated 'Script Engine' value 'pytho
n'.
********************

********************
******************** CHECKING FOR VARIABLES
```

```
******************** Variables no longer exist in NiFi 2.0 and are replaced
 by Parameters and Parameter Contexts.
********************
ERROR: Component identifier: 'e65e16f2-d5ba-32b4-a04e-1ae97c44da02' contains
 2 variable(s).
ERROR: Component identifier: '1428d513-770b-3c26-94c9-9641d57050c1' contains
 3 variable(s).
********************

********************
******************** CHECKING FOR EVENT DRIVEN SCHEDULING STRATEGY
******************** Event Driven Scheduling Strategy no longer exists. Use
Time Driven scheduling strategy instead.
********************
No component is configured with Event Driven scheduling strategy.
********************

********************
******************** CHECKING FOR TEMPLATES
******************** Templates no longer exist in NiFi 2.0. The use of NiFi
 Registry is recommended for versioning flows.
********************
ERROR: Component identifier: '6c387684-c2f2-3bbe-b485-8b6593cda6ba' with '
componentType' 'TEMPLATE'.
********************
********************
******************** CHECKING FOR INCORRECT KERBEROS CONFIGURATION
******************** NiFi components with Kerberos configuration that are a
vailable in NiFi 2.0 should be configured with the 'Kerberos User Service'
********************
ERROR: Component identifier: 'a716b969-9163-3e70-a162-b819b9ec4b63' of type
'org.apache.nifi.processors.kafka.pubsub.ConsumeKafka_2_6', group 'org.apach
e.nifi' and artifact 'nifi-kafka-2-6-nar' has configured Kerberos properties
. Please configure Kerberos using 'KerberosUserService', as this is the only
 configuration supported in NiFi 2.0.
ERROR: Component identifier: '0d650d47-5136-303c-a694-678a027b27cf' of type
 'org.apache.nifi.processors.parquet.PutParquet', group 'org.apache.nifi' a
nd artifact 'nifi-parquet-nar' has configured Kerberos properties. Please co
nfigure Kerberos using 'KerberosUserService', as this is the only configurat
ion supported in NiFi 2.0.
ERROR: Component identifier: '86c88c6a-369f-3209-9a79-dd6168c28b48' of type
 'org.apache.nifi.processors.hadoop.PutHDFS', group 'org.apache.nifi' and
artifact 'nifi-hadoop-nar' has configured Kerberos properties. Please config
ure Kerberos using 'KerberosUserService', as this is the only configuration
supported in NiFi 2.0.
********************
********************************************************************************
************************
Script finished work successfully
ERROR: You should not upgrade until the above errors are taken care of!
```

Manual pre-upgrade check for NiFi managed by Cloudera Manager

You can perform a manual pre-upgrade check from the command line on NiFi nodes even if NiFi is managed by Cloudera Manager. This approach provides administrators with greater control and is ideal for scenarios that require or prefer a manual process. Follow the below steps to perform a pre-upgrade check directly from the command line.

Procedure

1. Switch the user to NiFi using su nifi.

2. Find the location of these files.
   a. pre_upgrade_check.py
   b. CFM4-Deprecated-components.csv
   c. NiFi-1.x-components-with-legacy-Kerberos-properties.csv
3. Identify the ${NIFI_WORKING_DIRECTORY} environment variable, which can be found in stdout.log during NiFi startup.
4. Use Python3 to run the script with the following command, replacing the placeholders with the actual paths.

```
python3 [***/path/to***]/pre_upgrade_check.py \
"${NIFI_WORKING_DIRECTORY}/flow.json.gz" \
"[***/path/to***]/CFM4-Deprecated-components.csv" \
"[***/path/to***]/NiFi-1.x-components-with-legacy-Kerberos-properties.csv"
 \
"[***NIFI_WORKING_DIRECTORY***]/config_backup/bootstrap.conf" \
"${CUSTOM_JAVA_HOME}"
```

5. If necessary, specify the ${CUSTOM_JAVA_HOME} argument.

   This should be set if the JAVA_HOME environment variable for the "nifi" user is different from the JAVA _HOME you are using to run the NiFi service.
6. Once all file paths are updated in the command, run it to start the pre-upgrade check.

Manual pre-upgrade check for standalone NiFi (not managed by Cloudera Manager)

You can perform a manual pre-upgrade check from the command line on NiFi nodes in environments where NiFi is not managed by Cloudera Manager. Follow the below steps to perform a standalone pre-upgrade check directly from the command line.

Procedure

1. Download the pre-upgrade check script and the associated CSV files listing the components deprecated in NiFi 2 and the components with updated Kerberos properties.

   These files are part of the Cloudera Service Descriptor (CSD) package available behind the paywall.

   a) Download the NiFi CSD file and unzip it.
   b) Locate the required files.

   - scripts/service-commands/pre_upgrade_check.py
   - scripts/common-scripts/resources/CFM4-Deprecated-components.csv
   - scripts/common-scripts/resources/NiFi-1.x-components-with-legacy-Kerberos-properties.csv
2. After downloading the files, run the pre-upgrade check script using the following command.

```
su nifi
python3 [***path_to***]/pre_upgrade_check.py \
"[***path_to***]/flow.json.gz" \
"[***path_to***]/CFM4-Deprecated-components.csv" \
"[***path_to***]/NiFi-1.x-components-with-legacy-Kerberos-properties.csv"
 \
"[***path_to***]/bootstrap.conf" \
"" \
"[***path_to***]/nifi-framework-nar-<version>.nar" \
```

```
"[***path_to***]/nifi.sh"
```

Note:

Replace the placeholder paths with the actual locations of your files.

For example:

```
su nifi
python3 /tmp/scripts/service-commands/pre_upgrade_check.py \
/var/lib/nifi/flow.json.gz \
"/tmp/scripts/common-scripts/resources/CFM4-Deprecated-components.csv" \
"/tmp/scripts/common-scripts/resources/NiFi-1.x-components-with-legacy-K
erberos-properties.csv" \
"/tmp/nifi-1.26.0.2.1.7.1000-25/conf/bootstrap.conf" \
"" \
"/tmp/nifi-1.26.0.2.1.7.1000-25/lib/nifi-framework-nar-1.26.0.2.1.7.1000
-25.nar" \
"/tmp/nifi-1.26.0.2.1.7.1000-25/bin/nifi.sh"
```

The script generates a detailed report highlighting any issues that need to be resolved before upgrading to NiFi 2.

3. Review this report thoroughly to ensure a smooth upgrade process.