CEM MiNiFi C++ Agent 1.25.05

Using MiNiFi Agent Container Image

Date published: 2020-10-14
Date modified: 2025-06-30

CLOUDERA

https://docs.cloudera.com/

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

Download the MiNiFi C++ agent image..... 4

Use the container in standalone mode.....4

Use the container as a sidecar container in Kubernetes.....5

Download the MiNiFi C++ agent image

Learn how to deploy the MiNiFi C++ agent in a container. To do so, you need to retrieve the container image.

Deploying the MiNiFi C++ agent in a container is a very common use case, especially when deploying applications on Kubernetes. In such a situation, you may want to deploy the agent as a sidecar pod to collect logs, metrics, and so on. Having the agent registered to EFM is very powerful as you can change the data being collected and processed by the agent in the pods while the application is up and running.

If using Docker, you can retrieve the container image by the following command:

```
 % docker login -u *** -p *** container.repo.cloudera.com
  Login Succeeded
```

Note: The above command expects your Cloudera credentials derived from your license key. The same credentials you would use to access the CEM binaries.

You can then retrieve the latest version of the container using:

```
 % docker pull container.repo.cloudera.com/cloudera/apacheminificpp:latest
  latest: Pulling from cloudera/apacheminificpp
  801bfaa63ef2: Pull complete
  d90265e0d39e: Pull complete
  c5a0aa313d04: Pull complete
  ba189417f08f: Pull complete
  4f4fb700ef54: Pull complete
  Digest: sha256:aa253aa57ab3c6370ae321f41e61b45425d00ada7fe1e865fd665356c
064e53c
  Status: Downloaded newer image for container.repo.cloudera.com/cloudera/ap
acheminificpp:latest
  container.repo.cloudera.com/cloudera/apacheminificpp:latest
```

The container image is about 60 MB in size.

Use the container in standalone mode

Learn how to use the docker container in standalone mode.

The docker container can be used in standalone mode by providing the configuration files as a mounted volume. One option is to mount the configuration directory in the following way:

```
 % docker run -d -v /home/user/minifi-config:/opt/minifi/minifi-current/conf
/ container.repo.cloudera.com/cloudera/apacheminificpp:latest
```

The other option is to only mount the specific configuration files directly:

```
 % docker run -d -v /home/user/minifi-config/config.yml:/opt/minifi/minifi-c
urrent/conf/config.yml -v /home/user/minifi-config/minifi.properties:/opt/mi
nifi/minifi-current/conf/minifi.properties -v /home/user/minifi-config/minif
i-log.properties:/opt/minifi/minifi-current/conf/minifi-log.properties conta
iner.repo.cloudera.com/cloudera/apacheminificpp:latest
```

If the containerized MiNiFi instance is connected to an EFM instance through secure connection, you should also provide the certificates for the container. In case these certificates are found in the certs directory under the /home/user directory, you can mount it in the following way:

```
% docker run -d -v /home/user/minifi-config:/opt/minifi/minifi-current/conf/
 -v /home/user/certs:/certs container.repo.cloudera.com/cloudera/apacheminif
icpp:latest
```

In this case, you have to define the following properties in the minifi.properties file to configure the certificates:

```
nifi.security.client.certificate=/certs/client.pem
nifi.security.client.private.key=/certs/client.key
nifi.security.client.pass.phrase=/certs/password
nifi.security.client.ca.certificate=/certs/ca-cert
```

Use the container as a sidecar container in Kubernetes

Learn how to use the docker container as a sidecar container in Kubernetes.

The MiNiFi container can also be used in a Kubernetes environment, for example, as a logging agent for collecting logs from an application as a sidecar container. In the following example a use case is shown where a NiFi container runs in a pod together with a MiNiFi container as a sidecar collecting NiFi application logs.

For simplicity, the NiFi container is configured with default parameters so that no custom configuration is added for that instance. In this example, MiNiFi collects the NiFi logs, and after every minute it compresses and uploads the NiFi logs to an AWS S3 bucket. For this use case, you can use the following configuration:

```
Flow Controller:
  name: minifi-logging
Processors:
- id: 94b8e610-b4ed-1ec9-b26f-c839931bf3e2
  name: TailFile
  class: org.apache.nifi.processors.standard.TailFile
  scheduling strategy: TIMER_DRIVEN
  scheduling period: 5 sec
  auto-terminated relationships list: []
  Properties:
    File to Tail: /nifi-logs/nifi-app.log
    Lookup frequency: 1 min
- id: 261e8cf1-71ba-af86-fb2b-bc95764f91f8
  name: MergeContent
  class: org.apache.nifi.processors.standard.MergeContent
  scheduling strategy: EVENT_DRIVEN
  auto-terminated relationships list:
    - original
  Properties:
    Attribute Strategy: Keep Only Common Attributes
    Maximum number of Bins: 100
    Minimum Group Size: 0
    Max Bin Age: 1 min
    Minimum Number of Entries: 1000000
    Maximum Group Size: 1000000
    Maximum Number of Entries: 1000000
    Merge Strategy: Bin-Packing Algorithm
- id: 69335770-ee29-11eb-9a03-0242ac130003
  name: CompressContent
  class: org.apache.nifi.processors.standard.CompressContent
  scheduling strategy: EVENT_DRIVEN
  auto-terminated relationships list:
```

```
    - failure
  Properties:
    Compression Level: 6
    Compression Format: gzip
    UpdateFileName: false
- id: fe198bd9-2a1c-316e-0000-000000000000
  name: PutS3Object
  class: org.apache.nifi.minifi.azure.processors.PutS3Object
  scheduling strategy: EVENT_DRIVEN
  auto-terminated relationships list:
    - success
  Properties:
    Bucket: test_bucket
    AWS Credentials Provider service: AWSCredentialsService
Controller Services:
- name: AWSCredentialsService
  id: 2094d776-2006-4d02-9bb9-28eac9d0fc95
  class: org.apache.nifi.minifi.aws.controllers.AWSCredentialsService
  Properties:
    Use Default Credentials: 'true' # Can be used in Amazon EKS to retrieve
 credentials from metadata otherwise use your AWS Access Key and Secret Key
Connections:
- id: 99f617e7-49a1-6078-8534-26af7d56ca08
  name: TailFile/success/MergeContent
  source name: TailFile
  source relationship names:
  - success
  destination name: MergeContent
- id: 24d6be1e-ee29-11eb-9a03-0242ac130003
  name: MergeContent/merged/CompressContent
  source name: MergeContent
  source relationship names:
  - merged
  destination name: CompressContent
- id: 67ea5c91-446a-393b-6274-b6fae2f475a2
  name: CompressContent/success/PutS3Object
  source name: CompressContent
  source relationship names:
  - success
  destination name: PutS3Object
Remote Process Groups: []
```

In the pod specification you have to define the application and the MiNiFi containers to mount a shared volume of the logs to be collected. In the following example the NiFi logs are mounted to both NiFi and MiNiFi containers. MiNiFi also mounts the previously detailed configuration from the /root/minifi-config host directory.

```
apiVersion: v1
kind: Pod
metadata:
  name: log-collection-minifi-pod
  namespace: default
spec:
  containers:
  - name: nifi
    image: apache/nifi:latest
    volumeMounts:
    - name: nifi-logs
      mountPath: /opt/nifi/nifi-current/logs
  - name: sidecar-minifi
    image: container.repo.cloudera.com/cloudera/apacheminificpp:latest
    volumeMounts:
    - name: nifi-logs
      mountPath: /nifi-logs
```

```
    - name: minifi-config
      mountPath: /opt/minifi/minifi-current/conf/
  volumes:
  - name: nifi-logs
    emptyDir: {}
  - name: minifi-config
    hostPath:
      path: /root/minifi-config
```

Another option would be to instead of mounting the configuration files from the host directory they can be configured in a Kubernetes configuration map beforehand and this way they can be mounted individually. A configuration map definition template would look something like this:

```
apiVersion: v1
data:
  minifi-log.properties: |
    <minifi log properties file content>
  minifi.properties: |
    <minifi properties file content>
  config.yml: |
    <config file content>
kind: ConfigMap
metadata:
  labels:
    k8s-app: minifi-log-collection
  name: minifi-log-collection-config
  namespace: default
```

With the configuration map defined, you can change the previous pod definition in the following way to use the configuration map volume:

```
apiVersion: v1
kind: Pod
metadata:
  name: log-collection-minifi-pod
  namespace: default
spec:
  containers:
  - name: nifi
    image: apache/nifi:latest
    volumeMounts:
    - name: nifi-logs
      mountPath: /opt/nifi/nifi-current/logs
  - name: sidecar-minifi
    image: container.repo.cloudera.com/cloudera/apacheminificpp:latest
    volumeMounts:
    - name: nifi-logs
      mountPath: /nifi-logs
    - name: minificonfig
      mountPath: /opt/minifi/minifi-current/conf/config.yml
      subPath: config.yml
    - name: minificonfig
      mountPath: /opt/minifi/minifi-current/conf/minifi-log.properties
      subPath: minifi-log.properties
  volumes:
  - name: nifi-logs
    emptyDir: {}
  - configMap:
      defaultMode: 420
      name: minifi-log-collection-config
    name: minificonfig
```

You can find an example of sidecar Kubernetes YAML configuration file here with the previous pod and configuration map definitions to try. The suggested changes are marked with the TODO keyword.