Red Hat Data Grid 8.4

Building and deploying Data Grid clusters with Helm

Create Data Grid clusters on OpenShift

Last Updated: 2024-04-19

Red Hat Data Grid 8.4 Building and deploying Data Grid clusters with Helm

Create Data Grid clusters on OpenShift

Legal Notice

Abstract

Build and deploy Data Grid clusters with Helm.

Table of Contents

RED HAT DATA GRID

Data Grid is a high-performance, distributed in-memory data store.

Schemaless data structure

Flexibility to store different objects as key-value pairs.

Grid-based data storage

Designed to distribute and replicate data across clusters.

Elastic scaling

Dynamically adjust the number of nodes to meet demand without service disruption.

Data interoperability

Store, retrieve, and query data in the grid from different endpoints.

DATA GRID DOCUMENTATION

Documentation for Data Grid is available on the Red Hat customer portal.

- Data Grid 8.4 Documentation
- Data Grid 8.4 Component Details
- Supported Configurations for Data Grid 8.4
- Data Grid 8 Feature Support
- Data Grid Deprecated Features and Functionality

DATA GRID DOWNLOADS

Access the Data Grid Software Downloads on the Red Hat customer portal.

NOTE

You must have a Red Hat account to access and download Data Grid software.

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.

CHAPTER 1. DEPLOYING DATA GRID CLUSTERS AS HELM CHART RELEASES

Build, configure, and deploy Data Grid clusters with Helm. Data Grid provides a Helm chart that packages resources for running Data Grid clusters on OpenShift.

Install the Data Grid chart to create a Helm release, which instantiates a Data Grid cluster in your OpenShift project.

1.1. INSTALLING THE DATA GRID CHART THROUGH THE OPENSHIFT CONSOLE

Use the OpenShift Web Console to install the Data Grid chart from the Red Hat developer catalog. Installing the chart creates a Helm release that deploys a Data Grid cluster.

Prerequisites

- Have access to OpenShift.

Procedure

1. Log in to the OpenShift Web Console.
2. Select the **Developer** perspective.
3. Open the **Add** view and then select **Helm Chart** to browse the Red Hat developer catalog.
4. Locate and select the Data Grid chart.
5. Specify a name for the chart and select a version.
6. Define values in the following sections of the Data Grid chart:
   - **Images** configures the container images to use when creating pods for your Data Grid cluster.
   - **Deploy** configures your Data Grid cluster.

     TIP

     To find descriptions for each value, select the **YAML view** option and access the schema. Edit the yaml configuration to customize your Data Grid chart.
7. Select **Install**.

Verification

1. Select the **Helm** view in the **Developer** perspective.
2. Select the Helm release you created to view details, resources, and other information.

1.2. INSTALLING THE DATA GRID CHART ON THE COMMAND LINE

Use the command line to install the Data Grid chart on OpenShift and instantiate a Data Grid cluster. Installing the chart creates a Helm release that deploys a Data Grid cluster.

Prerequisites

- Install the **helm** client.
- Add the OpenShift Helm Charts repository.
- Have access to an OpenShift cluster.
- Have an **oc** client.

Procedure

1. Create a values file that configures your Data Grid cluster.
   For example, the following values file creates a cluster with two nodes:

```
$ cat > infinispan-values.yaml<<EOF
#Build configuration
images:
  server: registry.redhat.io/datagrid/datagrid-8-rhel8:latest
  initContainer: registry.access.redhat.com/ubi8-micro
#Deployment configuration
deploy:
  #Add a user with full security authorization.
  security:
    batch: "user create admin -p changeme"
  #Create a cluster with two pods.
  replicas: 2
  #Specify the internal Kubernetes cluster domain.
  clusterDomain: cluster.local
EOF
```

2. Install the Data Grid chart and specify your values file.

```
$ helm install infinispan openshift-helm-charts/redhat-data-grid --values infinispan-values.yaml
```

TIP

Use the **--set** flag to override configuration values for the deployment. For example, to create a cluster with three nodes:

```
--set deploy.replicas=3
```

Verification

Watch the pods to ensure all nodes in the Data Grid cluster are created successfully.

```
$ oc get pods -w
```

1.3. UPGRADING DATA GRID HELM RELEASES

Modify your Data Grid cluster configuration at runtime by upgrading Helm releases.

Prerequisites

- Deploy the Data Grid chart.
- Have a **helm** client.
- Have an **oc** client.

Procedure

1. Modify the values file for your Data Grid deployment as appropriate.
2. Use the **helm** client to apply your changes, for example:

```
$ helm upgrade infinispan openshift-helm-charts/redhat-data-grid --values infinispan-values.yaml
```

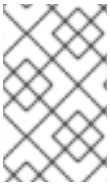
Verification

Watch the pods rebuild to ensure all changes are applied to your Data Grid cluster successfully.

```
$ oc get pods -w
```

1.4. UNINSTALLING DATA GRID HELM RELEASES

Uninstall a release of the Data Grid chart to remove pods and other deployment artifacts.



NOTE

This procedure shows you how to uninstall a Data Grid deployment on the command line but you can use the OpenShift Web Console instead. Refer to the OpenShift documentation for specific instructions.

Prerequisites

- Deploy the Data Grid chart.
- Have a **helm** client.
- Have an **oc** client.

Procedure

1. List the installed Data Grid Helm releases.

```
$ helm list
```

2. Use the **helm** client to uninstall a release and remove the Data Grid cluster:

```
$ helm uninstall <helm_release_name>
```

3. Use the **oc** client to remove the generated secret.

```
$ oc delete secret <helm_release_name>-generated-secret
```

1.5. DEPLOYMENT CONFIGURATION VALUES

Deployment configuration values let you customize Data Grid clusters.

TIP

You can also find field and value descriptions in the Data Grid chart README.

Field	Description	Default value
deploy.clusterDomain	Specifies the internal Kubernetes cluster domain.	**cluster.local**
deploy.replicas	Specifies the number of nodes in your Data Grid cluster, with a pod created for each node.	**1**
deploy.container.extraJvmOpts	Passes JVM options to Data Grid Server.	No default value.
deploy.container.libraries	Libraries to be downloaded before server startup. Specify multiple, space-separated artifacts represented as URLs or as Maven coordinates. Archive artifacts in .tar, .tar.gz or .zip formats will be extracted.	No default value.
deploy.container.storage.ephemeral	Defines whether storage is ephemeral or permanent.	The default value is **false**, which means data is permanent. Set the value to **true** to use ephemeral storage, which means all data is deleted when clusters shut down or restart.
deploy.container.storage.size	Defines how much storage is allocated to each Data Grid pod.	1Gi

Field	Description	Default value
deploy.container.storage.storageClassName	Specifies the name of a **StorageClass** object to use for the persistent volume claim (PVC).	No default value. By default, the persistent volume claim uses the storage class that has the **storageclass.kubernetes.io/is-default-class** annotation set to **true**. If you include this field, you must specify an existing storage class as the value.
deploy.container.resources.limits.cpu	Defines the CPU limit, in CPU units, for each Data Grid pod.	500m
deploy.container.resources.limits.memory	Defines the maximum amount of memory, in bytes, for each Data Grid pod.	512Mi
deploy.container.resources.requests.cpu	Specifies the maximum CPU requests, in CPU units, for each Data Grid pod.	500m
deploy.container.resources.requests.memory	Specifies the maximum memory requests, in bytes, for each Data Grid pod.	512Mi
deploy.security.secretName	Specifies the name of a secret that creates credentials and configures security authorization.	No default value. If you create a custom security secret then **deploy.security.batch** does not take effect.
deploy.security.batch	Provides a batch file for the Data Grid command line interface (CLI) to create credentials and configure security authorization at startup.	No default value.
deploy.expose.type	Specifies the service that exposes Hot Rod and REST endpoints on the network and provides access to your Data Grid cluster, including the Data Grid Console.	**Route** Valid options are: **""** (empty value), **Route**, **LoadBalancer**, and **NodePort**. Set an empty value (**""**) if you do not want to expose Data Grid on the network.
deploy.expose.nodePort	Specifies a network port for node port services within the default range of 30000 to 32767.	0 If you do not specify a port, the platform selects an available one.

Field	Description	Default value
deploy.expose.host	Optionally specifies the hostname where the Route is exposed.	No default value.
deploy.expose.annotations	Adds annotations to the service that exposes Data Grid on the network.	No default value.
deploy.logging.categories	Configures Data Grid cluster log categories and levels.	No default value.
deploy.podLabels	Adds labels to each Data Grid pod that you create.	No default value.
deploy.svcLabels	Adds labels to each service that you create.	No default value.
deploy.resourceLabels	Adds labels to all Data Grid resources including pods and services.	No default value.
deploy.makeDataDirWritable	Allows write access to the **data** directory for each Data Grid Server node.	**false** If you set the value to **true**, Data Grid creates an initContainer that runs **chmod -R** on the **/opt/infinispan/server/data** directory to change permissions.
deploy.securityContext	Configures the securityContext used by the StatefulSet pods.	**{}** This can be used to change the group of mounted file systems. Set **securityContext.fsGroup** to **185** if you need to explicitly match the group owner for **/opt/infinispan/server/data** to the default Data Grid's group
deploy.monitoring.enabled	Enable or disable monitoring using **ServiceMonitor**.	**false** Users must have **monitoring-edit** role assigned by the admin to deploy the Helm chart with **ServiceMonitor** enabled.
deploy.nameOverride	Specifies a name for all Data Grid cluster resources.	Helm Chart release name.

Field	Description	Default value
deploy.infinispan	Data Grid Server configuration.	Data Grid provides default server configuration. For more information about configuring server instances, see *Data Grid Server configuration values*.

CHAPTER 2. CONFIGURING DATA GRID SERVERS

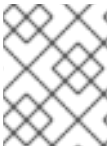
Apply custom Data Grid Server configuration to your deployments.

2.1. CUSTOMIZING DATA GRID SERVER CONFIGURATION

Apply custom **deploy.infinispan** values to Data Grid clusters that configure the Cache Manager and underlying server mechanisms like security realms or Hot Rod and REST endpoints.

IMPORTANT

You must always provide a complete Data Grid Server configuration when you modify **deploy.infinispan** values.



NOTE

Do not modify or remove the default "metrics" configuration if you want to use monitoring capabilities for your Data Grid cluster.

Procedure

Modify Data Grid Server configuration as required:

- Specify configuration values for the Cache Manager with **deploy.infinispan.cacheContainer** fields.
  For example, you can create caches at startup with any Data Grid configuration or add cache templates and use them to create caches on demand.
- Configure security authorization to control user roles and permissions with the **deploy.infinispan.cacheContainer.security.authorization** field.
- Select one of the default JGroups stacks or configure cluster transport with the **deploy.infinispan.cacheContainer.transport** fields.
- Configure Data Grid Server endpoints with the **deploy.infinispan.server.endpoints** fields.
- Configure Data Grid Server network interfaces and ports with the **deploy.infinispan.server.interfaces** and **deploy.infinispan.server.socketBindings** fields.
- Configure Data Grid Server security mechanisms with the **deploy.infinispan.server.security** fields.

2.2. DATA GRID SERVER CONFIGURATION VALUES

Data Grid Server configuration values let you customize the Cache Manager and modify server instances that run in OpenShift pods.

Data Grid Server configuration

```
deploy:
  infinispan:
    cacheContainer:
    # [USER] Add cache, template, and counter configuration.
    name: default
```

```
# [USER] Specify `security: null` to disable security authorization.
security:
  authorization: {}
transport:
  cluster: ${infinispan.cluster.name:cluster}
  node-name: ${infinispan.node.name:}
  stack: kubernetes
server:
  endpoints:
    # [USER] Hot Rod and REST endpoints.
    - securityRealm: default
      socketBinding: default
    # [METRICS] Metrics endpoint for cluster monitoring capabilities.
    - connectors:
        rest:
          restConnector:
            authentication:
              mechanisms: BASIC
      securityRealm: metrics
      socketBinding: metrics
  interfaces:
  - inetAddress:
      value: ${infinispan.bind.address:127.0.0.1}
    name: public
  security:
    credentialStores:
    - clearTextCredential:
        clearText: secret
      name: credentials
      path: credentials.pfx
    securityRealms:
    # [USER] Security realm for the Hot Rod and REST endpoints.
    - name: default
      # [USER] Comment or remove this properties realm to disable authentication.
      propertiesRealm:
        groupProperties:
          path: groups.properties
        groupsAttribute: Roles
        userProperties:
          path: users.properties
      # [METRICS] Security realm for the metrics endpoint.
    - name: metrics
      propertiesRealm:
        groupProperties:
          path: metrics-groups.properties
          relativeTo: infinispan.server.config.path
        groupsAttribute: Roles
        userProperties:
          path: metrics-users.properties
          plainText: true
          relativeTo: infinispan.server.config.path
    socketBindings:
      defaultInterface: public
      portOffset: ${infinispan.socket.binding.port-offset:0}
      socketBinding:
        # [USER] Socket binding for the Hot Rod and REST endpoints.
```

```
        - name: default
          port: 11222
          # [METRICS] Socket binding for the metrics endpoint.
        - name: metrics
          port: 11223
```

Data Grid cache configuration

```
deploy:
  infinispan:
    cacheContainer:
      distributedCache:
        name: "mycache"
        mode: "SYNC"
        owners: "2"
        segments: "256"
        capacityFactor: "1.0"
        statistics: "true"
        encoding:
          mediaType: "application/x-protostream"
        expiration:
          lifespan: "5000"
          maxIdle: "1000"
        memory:
          maxCount: "1000000"
          whenFull: "REMOVE"
        partitionHandling:
          whenSplit: "ALLOW_READ_WRITES"
          mergePolicy: "PREFERRED_NON_NULL"
    #Provide additional Cache Manager configuration.
  server:
    #Provide configuration for server instances.
```

Cache template

```
deploy:
  infinispan:
    cacheContainer:
      distributedCacheConfiguration:
        name: "my-dist-template"
        mode: "SYNC"
        statistics: "true"
        encoding:
          mediaType: "application/x-protostream"
        expiration:
          lifespan: "5000"
          maxIdle: "1000"
        memory:
          maxCount: "1000000"
          whenFull: "REMOVE"
    #Provide additional Cache Manager configuration.
  server:
    #Provide configuration for server instances.
```

Cluster transport

```
deploy:
  infinispan:
    cacheContainer:
      transport:
        #Specifies the name of a default JGroups stack.
        stack: kubernetes
    #Provide additional Cache Manager configuration.
  server:
    #Provide configuration for server instances.
```

Additional resources

- Data Grid Server Guide
- Configuring Data Grid

CHAPTER 3. CONFIGURING AUTHENTICATION AND AUTHORIZATION

Control access to Data Grid clusters by adding credentials and assigning roles with different permissions.

3.1. DEFAULT CREDENTIALS

Data Grid adds default credentials in a **<helm_release_name>-generated-secret** secret.

Username	Description
developer	User that has the **admin** role with full access to Data Grid resources.
monitor	Internal user that has the **monitor** role with access to Data Grid metrics through port **11223**.

Additional resources

- Data Grid Security Guide

3.1.1. Retrieving credentials

Get Data Grid credentials from authentication secrets.

Prerequisites

- Install the Data Grid Helm chart.
- Have an **oc** client.

Procedure

- Retrieve default credentials from the **<helm_release_name>-generated-secret** or custom credentials from another secret with the following command:

```
$ oc get secret <helm_release_name>-generated-secret \
-o jsonpath="{.data.identities-batch}" | base64 --decode
```

3.2. ADDING CUSTOM USER CREDENTIALS OR CREDENTIALS STORE

Create Data Grid user credentials and assign roles that grant security authorization for cluster access.

Procedure

- Create credentials by specifying the **user create** command in the **deploy.security.batch** field.

  User with implicit authorization

```
deploy:
  security:
    batch: 'user create admin -p changeme'
```

User with a specific role

```
deploy:
  security:
    batch: 'user create personone -p changeme -g deployer'
```

3.2.1. User roles and permissions

Data Grid uses role-based access control to authorize users for access to cluster resources and data. For additional security, you should grant Data Grid users with appropriate roles when you add credentials.

Role	Permissions	Description
admin	ALL	Superuser with all permissions including control of the Cache Manager lifecycle.
deployer	ALL_READ, ALL_WRITE, LISTEN, EXEC, MONITOR, CREATE	Can create and delete Data Grid resources in addition to **application** permissions.
application	ALL_READ, ALL_WRITE, LISTEN, EXEC, MONITOR	Has read and write access to Data Grid resources in addition to **observer** permissions. Can also listen to events and execute server tasks and scripts.
observer	ALL_READ, MONITOR	Has read access to Data Grid resources in addition to **monitor** permissions.
monitor	MONITOR	Can view statistics for Data Grid clusters.

Additional resources

- Data Grid Security Guide

3.2.2. Adding credentials store

Create Data Grid credentials store to avoid exposing passwords in clear text in the server configuration ConfigMap. See Section 4.1, “Enabling TLS encryption” for a use case.

Procedure

1. Create credentials store by specifying a **credentials add** command in the **deploy.security.batch** field.

   Add a password to a store

```
deploy:
  security:
    batch: 'credentials add keystore -c password -p secret --path="credentials.pfx"'
```

2. Credentials store needs then to be added to the server configuration.

   Configure a credential store

```
deploy:
  infinispan:
    server:
      security:
        credentialStores:
          - name: credentials
            path: credentials.pfx
            clearTextCredential:
              clearText: "secret"
```

3.2.3. Adding multiple credentials with authentication secrets

Add multiple credentials to Data Grid clusters with authentication secrets.

Prerequisites

- Have an **oc** client.

Procedure

1. Create an **identities-batch** file that contains the commands to add your credentials.

```
apiVersion: v1
kind: Secret
metadata:
  name: connect-secret
type: Opaque
stringData:
  # The "monitor" user authenticates with the Prometheus ServiceMonitor.
  username: monitor
  # The password for the "monitor" user.
  password: password
  # The key must be 'identities-batch'.
  # The content is "user create" commands for the Data Grid CLI.
  identities-batch: |-
    user create user1 -p changeme -g admin
    user create user2 -p changeme -g deployer
    user create monitor -p password --users-file metrics-users.properties --groups-file metrics-
groups.properties
    credentials add keystore -c password -p secret --path="credentials.pfx"
```

2. Create an authentication secret from your **identities-batch** file.

```
$ oc apply -f identities-batch.yaml
```

3. Specify the authentication secret in the **deploy.security.SecretName** field.

```
deploy:
  security:
    authentication: true
    secretName: 'connect-secret'
```

4. Install or upgrade your Data Grid Helm release.

3.3. DISABLING AUTHENTICATION

Allow users to access Data Grid clusters and manipulate data without providing credentials.

IMPORTANT

Do not disable authentication if endpoints are accessible from outside the OpenShift cluster. You should disable authentication for development environments only.

Procedure

1. Remove the **propertiesRealm** fields from the "default" security realm.
2. Install or upgrade your Data Grid Helm release.

3.4. DISABLING SECURITY AUTHORIZATION

Allow Data Grid users to perform any operation regardless of their role.

Procedure

1. Set **null** as the value for the **deploy.infinispan.cacheContainer.security** field.

   TIP

   Use the **--set deploy.infinispan.cacheContainer.security=null** argument with the **helm** client.

2. Install or upgrade your Data Grid Helm release.

CHAPTER 4. CONFIGURING ENCRYPTION

Configure encryption for your Data Grid.

4.1. ENABLING TLS ENCRYPTION

Encryption can be independently enabled for endpoint and cluster transport.

Prerequisites

- A secret containing a certificate or a keystore. Endpoint and cluster should use different secrets.
- A credentials keystore containing any password needed to access the keystore. See Adding credentials keystore.

Procedure

1. Set the secret name in the deploy configuration.
   Provide the name of the secret containing the keystore:

```
deploy:
  ssl:
    endpointSecretName: "tls-secret"
    transportSecretName: "tls-transport-secret"
```

2. Enable cluster transport TLS.

```
deploy:
  infinispan:
    cacheContainer:
      transport:
        urn:infinispan:server:15.0:securityRealm: >
          "cluster-transport" 1
    server:
      security:
        securityRealms:
          - name: cluster-transport
            serverIdentities:
              ssl:
                keystore: 2
                  alias: "server"
                  path: "/etc/encrypt/transport/cert.p12"
                  credentialReference: 3
                    store: credentials
                    alias: keystore
                truststore: 4
                  path: "/etc/encrypt/transport/cert.p12"
                  credentialReference: 5
                    store: credentials
                    alias: truststore
```

1 Configures the transport stack to use the specified security-realm to provide cluster encryption.

2 Configure the keystore path in the transport realm. The secret is mounted at **/etc/encrypt/transport**.

3 5 Configures the truststore with the same keystore allowing the nodes to authenticate each other.

4 Alias and password must be provided in case the secret contains a keystore.

3. Enable endpoint TLS.

```
deploy:
  infinispan:
    server:
      security:
        securityRealms:
          - name: default
            serverIdentities:
              ssl:
                keystore:
                  path: "/etc/encrypt/endpoint/keystore.p12" 1
                  alias: "server" 2
                  credentialReference:
                    store: credentials 3
                    alias: keystore 4
```

1 Configure the keystore path in the endpoint realm; secret is mounted at **/etc/encrypt/endpoint**.

2 Alias must be provided in case the secret contains a keystore.

3 4 Any password must be provided with via credentials keystore.

Additional resources

- Data Grid Security Guide

CHAPTER 5. CONFIGURING NETWORK ACCESS

Configure network access for your Data Grid deployment and find out about internal network services.

5.1. EXPOSING DATA GRID CLUSTERS ON THE NETWORK

Make Data Grid clusters available on the network so you can access Data Grid Console as well as REST and Hot Rod endpoints. By default, the Data Grid chart exposes deployments through a Route but you can configure it to expose clusters via Load Balancer or Node Port. You can also configure the Data Grid chart so that deployments are not exposed on the network and only available internally to the OpenShift cluster.

Procedure

1. Specify one of the following for the **deploy.expose.type** field:

Option	Description
Route	Exposes Data Grid through a route. This is the default value.
LoadBalancer	Exposes Data Grid through a load balancer service.
NodePort	Exposes Data Grid through a node port service.
"" (empty value)	Disables exposing Data Grid on the network.

2. Optionally specify a hostname with the **deploy.expose.host** field if you expose Data Grid through a route.
3. Optionally specify a port with the **deploy.expose.nodePort** field if you expose Data Grid through a node port service.
4. Install or upgrade your Data Grid Helm release.

5.2. RETRIEVING NETWORK SERVICE DETAILS

Get network service details so you can connect to Data Grid clusters.

Prerequisites

- Expose your Data Grid cluster on the network.
- Have an **oc** client.

Procedure

Use one of the following commands to retrieve network service details:

- If you expose Data Grid through a route:

```
$ oc get routes
```

- If you expose Data Grid through a load balancer or node port service:

```
$ oc get services
```

5.3. NETWORK SERVICES

The Data Grid chart creates default network services for internal access.

Service	Port	Protocol	Description
<helm_release_name>	**11222**	TCP	Provides access to Data Grid Hot Rod and REST endpoints.
<helm_release_name>	**11223**	TCP	Provides access to Data Grid metrics.
<helm_release_name>-ping	**8888**	TCP	Allows Data Grid pods to discover each other and form clusters.

You can retrieve details about internal network services as follows:

```
$ oc get services

NAME             TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)
infinispan       ClusterIP   192.0.2.0        <none>        11222/TCP,11223/TCP
infinispan-ping  ClusterIP   None             <none>        8888/TCP
```

CHAPTER 6. CONNECTING TO DATA GRID CLUSTERS

After you configure and deploy Data Grid clusters you can establish remote connections through the Data Grid Console, command line interface (CLI), Hot Rod client, or REST API.

6.1. ACCESSING DATA GRID CONSOLE

Access the console to create caches, perform adminstrative operations, and monitor your Data Grid clusters.

Prerequisites

- Expose your Data Grid cluster on the network.
- Retrieve network service details.

Procedure

- Access Data Grid Console from any browser at **$SERVICE_HOSTNAME:$PORT**. Replace **$SERVICE_HOSTNAME:$PORT** with the hostname and port where Data Grid is available on the network.

6.2. CONNECTING WITH THE COMMAND LINE INTERFACE (CLI)

Use the Data Grid CLI to connect to clusters and create caches, manipulate data, and perform administrative operations.

Prerequisites

- Expose your Data Grid cluster on the network.
- Retrieve network service details.
- Download the native Data Grid CLI distribution from the Data Grid software downloads.
- Extract the **.zip** archive for the native Data Grid CLI distribution to your host filesystem.

Procedure

1. Start the Data Grid CLI with the network service as the value for the **-c** argument, for example:

```
$ {native_cli} -c http://cluster-name-myroute.hostname.net/
```

2. Enter your Data Grid credentials when prompted.
3. Perform CLI operations as required.

   TIP

   Press the tab key or use the **--help** argument to view available options and help text.

4. Use the **quit** command to exit the CLI.

Additional resources

- Using the Data Grid Command Line Interface

6.3. CONNECTING HOT ROD CLIENTS RUNNING ON OPENSHIFT

Access remote caches with Hot Rod clients running on the same OpenShift cluster as your Data Grid cluster.

Prerequisites

- Retrieve network service details.

Procedure

1. Specify the internal network service detail for your Data Grid cluster in the client configuration. In the following configuration examples, **$SERVICE_HOSTNAME:$PORT** denotes the hostname and port that allows access to your Data Grid cluster.

2. Specify your credentials so the client can authenticate with Data Grid.

3. Configure client intelligence, if required.
   Hot Rod clients running on OpenShift can use any client intelligence because they can access internal IP addresses for Data Grid pods.
   The default intelligence, **HASH_DISTRIBUTION_AWARE**, is recommended because it allows clients to route requests to primary owners, which improves performance.

Programmatic configuration

```
import org.infinispan.client.hotrod.configuration.ConfigurationBuilder;
import org.infinispan.client.hotrod.configuration.SaslQop;
import org.infinispan.client.hotrod.impl.ConfigurationProperties;
...

ConfigurationBuilder builder = new ConfigurationBuilder();
    builder.addServer()
            .host("$SERVICE_HOSTNAME")
            .port(ConfigurationProperties.DEFAULT_HOTROD_PORT)
           .security().authentication()
            .username("username")
            .password("changeme")
            .realm("default")
            .saslQop(SaslQop.AUTH)
            .saslMechanism("SCRAM-SHA-512");
```

Hot Rod client properties

```
# Connection
infinispan.client.hotrod.server_list=$SERVICE_HOSTNAME:$PORT

# Authentication
infinispan.client.hotrod.use_auth=true
infinispan.client.hotrod.auth_username=developer
infinispan.client.hotrod.auth_password=$PASSWORD
```

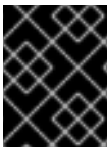
```
infinispan.client.hotrod.auth_server_name=$CLUSTER_NAME
infinispan.client.hotrod.sasl_properties.javax.security.sasl.qop=auth
infinispan.client.hotrod.sasl_mechanism=SCRAM-SHA-512
```

Additional resources

- Hot Rod Java Client Guide

6.3.1. Obtaining IP addresses for all Data Grid pods

You can retrieve a list of all IP addresses for running Data Grid pods.



IMPORTANT

Connecting Hot Rod clients running on OpenShift is the recommended approach as it ensures the initial connection to one of the available pods.

Procedure

Obtain all the IP addresses for a running Data Grid pods in the following ways:

- Using the OpenShift API:
  - Access **${APISERVER}/api/v1/namespaces/<chart-namespace>/endpoints/<helm-release-name>** to retrieve the **endpoints** OpenShift resource associated with the **<helm-release-name>** service.
- Using the OpenShift DNS service:
  - Query the DNS service for the name **<helm-release-name>-ping** to obtain IPs for all the nodes in a cluster.

Additional resources

- Accessing the Kubernetes API from a Pod
- DNS for Services and Pods

6.4. CONNECTING HOT ROD CLIENTS RUNNING OUTSIDE OPENSHIFT

Access remote caches with Hot Rod clients running externally to the OpenShift cluster where you deploy your Data Grid cluster.

Prerequisites

- Expose your Data Grid cluster on the network.
- Retrieve network service details.

Procedure

1. Specify the internal network service detail for your Data Grid cluster in the client configuration.

In the following configuration examples, **$SERVICE_HOSTNAME:$PORT** denotes the hostname and port that allows access to your Data Grid cluster.

2. Specify your credentials so the client can authenticate with Data Grid.

3. Configure clients to use **BASIC** intelligence.

Programmatic configuration

```
import org.infinispan.client.hotrod.configuration.ClientIntelligence;
import org.infinispan.client.hotrod.configuration.ConfigurationBuilder;
import org.infinispan.client.hotrod.configuration.SaslQop;
...

ConfigurationBuilder builder = new ConfigurationBuilder();
    builder.addServer()
            .host("$SERVICE_HOSTNAME")
            .port("$PORT")
           .security().authentication()
            .username("username")
            .password("changeme")
            .realm("default")
            .saslQop(SaslQop.AUTH)
            .saslMechanism("SCRAM-SHA-512");
    builder.clientIntelligence(ClientIntelligence.BASIC);
```

Hot Rod client properties

```
# Connection
infinispan.client.hotrod.server_list=$SERVICE_HOSTNAME:$PORT

# Client intelligence
infinispan.client.hotrod.client_intelligence=BASIC

# Authentication
infinispan.client.hotrod.use_auth=true
infinispan.client.hotrod.auth_username=developer
infinispan.client.hotrod.auth_password=$PASSWORD
infinispan.client.hotrod.auth_server_name=$CLUSTER_NAME
infinispan.client.hotrod.sasl_properties.javax.security.sasl.qop=auth
infinispan.client.hotrod.sasl_mechanism=SCRAM-SHA-512
```

Additional resources

- Hot Rod Java Client Guide

6.5. ACCESSING THE REST API

Data Grid provides a RESTful interface that you can interact with using HTTP clients.

Prerequisites

- Expose your Data Grid cluster on the network.
- Retrieve network service details.

Procedure

- Access the REST API with any HTTP client at **$SERVICE_HOSTNAME:$PORT/rest/v2**. Replace **$SERVICE_HOSTNAME:$PORT** with the hostname and port where Data Grid is available on the network.

Additional resources

- Data Grid REST API