Red Hat Data Grid 8.1

在 OpenShift 上运行 Data Grid

在 OpenShift 中配置并运行 Data Grid 服务

Last Updated: 2023-12-15

Red Hat Data Grid 8.1 在 OpenShift 上运行 Data Grid

在 OpenShift 中配置并运行 Data Grid 服务

法律通告

摘要

Data Grid Operator 提供操作智能，并减少了在 OpenShift 上部署 Data Grid 的管理复杂性。

目录

RED HAT DATA GRID

Data Grid 是一个高性能分布式内存数据存储。

无架构数据结构

将不同对象存储为键值对的灵活性。

基于网格的数据存储

旨在在集群中分发和复制数据。

弹性扩展

动态调整节点数量，以便在不中断服务的情况下满足需求。

数据互操作性

从不同端点在网格中存储、检索和查询数据。

DATA GRID 文档

红帽客户门户网站中提供了 Data Grid 的文档。

- Data Grid 8.1 文档
- Data Grid 8.1 组件详情
- Data Grid 8.1 支持的配置
- Data Grid 8 功能支持
- 数据中心已弃用的功能和功能

DATA GRID 下载

访问红帽客户门户上的 Data Grid 软件下载。

注意

您必须有一个红帽帐户才能访问和下载数据中心软件。

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。有关更多详情，请参阅我们的首席技术官 Chris Wright 提供的消息。

第 1 章 安装 DATA GRID OPERATOR

将 Data Grid Operator 安装到 OpenShift 命名空间中，以创建和管理 Data Grid 集群。

1.1. 在 RED HAT OPENSHIFT 上安装 DATA GRID OPERATOR

在 OpenShift 上创建 Data Grid Operator 的订阅，以便您可以安装不同的 Data Grid 版本并接收自动更新。

自动更新首先应用到 Data Grid Operator，然后用于每个 Data Grid 节点。Data Grid Operator 一次更新集群一个节点，安全地关闭每个节点，然后在进入下一节点前，使用更新的版本重新上线。

先决条件

- 访问在 OpenShift 上运行的 **OperatorHub**。一些 OpenShift 环境（如 OpenShift Container Platform）可能需要管理员凭证。
- 如果您计划将其安装到特定命名空间中，则具有 Data Grid Operator 的 OpenShift 项目。

流程

1. 登录 OpenShift Web 控制台。
2. 导航到 **OperatorHub**。
3. 查找并选择 Data Grid Operator。
4. 选择 **Install** and continue to **Create Operator Subscription**。
5. 指定订阅的选项。

   安装模式

   您可以将 Data Grid Operator 安装到 **特定命名空间** 或 **所有命名空间中**。

   更新频道

   获取 Data Grid Operator 8.1.x 的更新。

   批准策略

   自动安装来自 8.1.x 频道的更新，或在安装前需要批准。
6. 选择 **Subscribe** 来安装 Data Grid Operator。
7. 进入 **Installed Operators** 以验证 Data Grid Operator 安装。

1.2. 从命令行安装 DATA GRID OPERATOR

作为通过 OpenShift 上的 **OperatorHub** 安装 Data Grid Operator 的替代选择，请使用 **oc** 客户端来创建订阅。

先决条件

- 有一个 **oc** 客户端。

流程

1. 设置项目。

   a. 为 Data Grid Operator 创建一个项目。

   b. 如果您希望 Data Grid Operator 只控制特定的 Data Grid 集群，请为该集群创建一个项目。

```
$ oc new-project ${INSTALL_NAMESPACE} 1
$ oc new-project ${WATCH_NAMESPACE} 2
```

   1 创建一个安装 Data Grid Operator 的项目。

   2 如果您不希望 Data Grid Operator 监控所有项目，可以选择为特定 Data Grid 集群创建项目。

2. 创建 **OperatorGroup** 资源。

   控制所有 Data Grid 集群

```
$ oc apply -f - << EOF
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
 name: datagrid
 namespace: ${INSTALL_NAMESPACE}
EOF
```

   控制特定 Data Grid 集群

```
$ oc apply -f - << EOF
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
 name: datagrid
 namespace: ${INSTALL_NAMESPACE}
spec:
 targetNamespaces:
 - ${WATCH_NAMESPACE}
EOF
```

3. 为 Data Grid Operator 创建订阅。

```
$ oc apply -f - << EOF
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
 name: datagrid-operator
 namespace: ${INSTALL_NAMESPACE}
spec:
 channel: 8.1.x
 installPlanApproval: Automatic 1
 name: datagrid
 source: redhat-operators
 sourceNamespace: openshift-marketplace
EOF
```

1 如果要从 8.1.x 频道手动批准更新，请指定 **Manual**。

4. 验证安装。

```
$ oc get pods -n ${INSTALL_NAMESPACE}
NAME                              READY   STATUS
infinispan-operator-<id>          1/1     Running
```

第 2 章 DATA GRID OPERATOR 入门

借助 Data Grid Operator，您可以创建、配置和管理 Data Grid 集群。

先决条件

- 安装 Data Grid Operator。
- 有一个 **oc** 客户端。

2.1. INFINISPAN 自定义资源(CR)

Data Grid Operator 添加了一个类型为 **Infinispan** 的新自定义资源(CR)，可让您将 Data Grid 集群作为 OpenShift 中的复杂单元处理。

Data Grid Operator 会监视您用于实例化和配置 Data Grid 集群并管理 OpenShift 资源（如 StatefulSets 和 Services）的 **Infinispan** 自定义资源(CR)。这样，**Infinispan** CR 是 OpenShift 上 Data Grid 的主要接口。

最小 **Infinispan** CR 如下：

```
apiVersion: infinispan.org/v1 1
kind: Infinispan 2
metadata:
  name: example-infinispan 3
spec:
  replicas: 2 4
```

1. 声明 **Infinispan** API 版本。
2. 声明 **Infinispan** CR。
3. 将 Data Grid 命名为 cluster。
4. 指定 Data Grid 集群中的节点数。

2.2. 创建 DATA GRID 集群

使用 Data Grid Operator 创建由两个或多个 Data Grid 节点组成的集群。

流程

1. 在 **Infinispan** CR 中指定集群中带有 **spec.replicas** 的 Data Grid 节点数量。
   例如，创建一个 **cr_minimal.yaml** 文件，如下所示：

```
$ cat > cr_minimal.yaml<<EOF
apiVersion: infinispan.org/v1
kind: Infinispan
metadata:
  name: example-infinispan
```

```
spec:
  replicas: 2
EOF
```

2. 应用 **Infinispan** CR。

```
$ oc apply -f cr_minimal.yaml
```

3. 观察 Data Grid Operator 创建 Data Grid 节点。

```
$ oc get pods -w

NAME                    READY  STATUS            RESTARTS   AGE
example-infinispan-1      0/1    ContainerCreating  0          4s
example-infinispan-2      0/1    ContainerCreating  0          4s
example-infinispan-3      0/1    ContainerCreating  0          5s
infinispan-operator-0     1/1    Running            0          3m
example-infinispan-3      1/1    Running            0          8s
example-infinispan-2      1/1    Running            0          8s
example-infinispan-1      1/1    Running            0          8s
```

后续步骤

尝试更改 **replicas** 值： 并观察 Data Grid Operator 扩展集群或缩减。

2.3. 验证 DATA GRID 集群

查看日志消息，以确保 Data Grid 节点接收集群视图。

流程

- 执行以下操作之一：
  - 从日志中检索集群视图。

```
$ oc logs example-infinispan-0 | grep ISPN000094

INFO  [org.infinispan.CLUSTER] (MSC service thread 1-2) \
ISPN000094: Received new cluster view for channel infinispan: \
[example-infinispan-0|0] (1) [example-infinispan-0]

INFO  [org.infinispan.CLUSTER] (jgroups-3,example-infinispan-0) \
ISPN000094: Received new cluster view for channel infinispan: \
[example-infinispan-0|1] (2) [example-infinispan-0, example-infinispan-1]
```

  - 检索 Data Grid Operator 的 **Infinispan** CR。

```
$ oc get infinispan -o yaml
```

响应表示 Data Grid pod 已收到集群视图：

```
conditions:
    - message: 'View: [example-infinispan-0, example-infinispan-1]'
      status: "True"
```

```
        type: wellFormed
```

提示

对自动化脚本使用 **oc wait** 和 **wellFormed** 条件。

```
$ oc wait --for condition=wellFormed --timeout=240s infinispan/example-infinispan
```

第 3 章 设置 DATA GRID SERVICES

使用 Data Grid Operator 创建 Cache 服务或 Data Grid 服务节点的集群。

3.1. 服务类型

服务是有状态的应用程序，基于数据网格服务器镜像，提供灵活、强大的内存中数据存储。

缓存服务

如果您希望一个具有最小配置的易失性低延迟数据存储，请使用 Cache 服务。缓存服务节点：

- 数据存储需求上线或缩减时，自动缩放以满足容量。
- 同步分发数据以确保一致性。
- 在集群中复制缓存中的每个条目。
- 将缓存条目存储为 off-heap，并将驱除用于 JVM 效率。
- 确保与默认分区处理配置保持一致。

重要

因为缓存服务节点具有易失性，所以在使用 **Infinispan** CR 对集群应用更改或更新 Data Grid 版本时会丢失所有数据。

Data Grid 服务

如果要，请使用 Data Grid 服务：

- 使用跨站点复制在全局集群中备份数据。
- 使用任何有效配置创建缓存。
- 添加基于文件的缓存存储，将数据保存在持久性卷中。
- 使用 Data Grid 搜索和其他高级功能。

3.2. 创建缓存服务节点

默认情况下，Data Grid Operator 使用缓存服务节点创建 Data Grid 集群。

流程

1. 创建 **Infinispan** CR。

```
apiVersion: infinispan.org/v1
kind: Infinispan
metadata:
  name: example-infinispan
spec:
  replicas: 2
  service:
    type: Cache 1
```

```
-
```

1 创建节点缓存服务节点。这是 **Infinispan** CR 的默认设置。

2. 应用 **Infinispan** CR 以创建集群。

3.2.1. 配置自动扩展

如果您使用缓存服务节点创建集群，Data Grid Operator 可以根据默认缓存的内存用量自动扩展或缩减节点。

Data Grid Operator 监控缓存服务节点上的默认缓存。当您向缓存中添加数据时，内存用量会增加。当检测到集群需要额外的容量时，Data Grid Operator 会创建新节点而不是撤离条目。同样，如果它检测到内存用量低于特定阈值，则 Data Grid Operator 会关闭节点。

重要

自动扩展仅适用于默认缓存。如果您计划在集群中添加其他缓存，则不应在 **Infinispan** CR 中包含 **autoscale** 字段。在这种情况下，您应该使用驱除来控制每个节点上的数据容器的大小。

流程

1. 将 **spec.autoscale** 资源添加到 **Infinispan** CR 中，以启用自动扩展。
2. 使用 **autoscale** 字段为集群配置内存用量阈值和节点数。

```
spec:
  ...
  service:
    type: Cache
  autoscale:
    maxMemUsagePercent: 70 1
    maxReplicas: 5 2
    minMemUsagePercent: 30 3
    minReplicas: 2 4
```

1 配置每个节点上的内存用量的最大阈值（百分比）。当 Data Grid Operator 检测到集群中的任何节点都达到阈值时，它会创建一个新节点（如果可能）。如果 Data Grid Operator 无法创建新节点，则它在内存用量达到 100% 时执行驱除。

2 定义集群的最大节点数。

3 将最小阈值（百分比）配置为集群中内存用量。当 Data Grid Operator 检测到内存用量低于最小值时，它会关闭节点。

4 定义集群的最小节点数。

3. 应用更改。

3.2.2. 配置所有者数

所有者数量控制每个缓存条目在 Data Grid 集群中复制多少个副本。Cache 服务节点的默认值为两个，它会复制每个条目以防止数据丢失。

流程

1. 在 **Infinispan** CR 中使用 **spec.service.replicationFactor** 资源指定所有者数量，如下所示：

```
spec:
  ...
  service:
    type: Cache
    replicationFactor: 3 (1)
```

(1) 为每个缓存条目配置三个副本。

2. 应用更改。

3.2.3. 缓存服务资源

```
apiVersion: infinispan.org/v1
kind: Infinispan
metadata:
  # Names the cluster.
  name: example-infinispan
spec:
  # Specifies the number of nodes in the cluster.
  replicas: 4
  service:
    # Configures the service type as Cache.
    type: Cache
    # Sets the number of replicas for each entry across the cluster.
    replicationFactor: 2
  # Enables and configures automatic scaling.
  autoscale:
    maxMemUsagePercent: 70
    maxReplicas: 5
    minMemUsagePercent: 30
    minReplicas: 2
  # Configures authentication and encryption.
  security:
    # Defines a secret with custom credentials.
    endpointSecretName: endpoint-identities
    # Adds a custom TLS certificate to encrypt client connections.
    endpointEncryption:
        type: Secret
        certSecretName: tls-secret
  # Sets container resources.
  container:
    extraJvmOpts: "-XX:NativeMemoryTracking=summary"
    cpu: "2000m"
    memory: 1Gi
  # Configures logging levels.
  logging:
    categories:
      org.infinispan: trace
      org.jgroups: trace
  # Configures how the cluster is exposed on the network.
```

```
  expose:
    type: LoadBalancer
  affinity:
    podAntiAffinity:
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 100
        podAffinityTerm:
          labelSelector:
            matchLabels:
              app: infinispan-pod
              clusterName: example-infinispan
              infinispan_cr: example-infinispan
          topologyKey: "kubernetes.io/hostname"
```

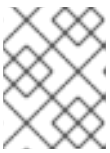
3.3. 创建 DATA GRID 服务节点

要使用自定义缓存定义以及 Data Grid 功能（如跨站点复制），请创建 Data Grid 服务节点集群。

流程

1. 在 **Infinispan** CR 中指定 **DataGrid** 作为 **spec.service.type** 的值。

```
apiVersion: infinispan.org/v1
kind: Infinispan
metadata:
  name: example-infinispan
spec:
  replicas: 2
  service:
    type: DataGrid
```



注意

您无法在创建节点后更改 **spec.service.type** 字段。要更改服务类型，您必须删除现有节点并创建新节点。

2. 使用任何其他 Data Grid 服务资源配置节点。

3. 应用 **Infinispan** CR 以创建集群。

3.3.1. Data Grid 服务资源

```
apiVersion: infinispan.org/v1
kind: Infinispan
metadata:
  # Names the cluster.
  name: example-infinispan
spec:
  # Specifies the number of nodes in the cluster.
  replicas: 6
  service:
    # Configures the service type as Data Grid.
    type: DataGrid
```

```
  # Configures storage resources.
  container:
    storage: 2Gi
    storageClassName: my-storage-class
  # Configures cross-site replication.
  sites:
    local:
    name: azure
    expose:
      type: LoadBalancer
    locations:
    - name: azure
      url: openshift://api.azure.host:6443
      secretName: azure-token
    - name: aws
      url: openshift://api.aws.host:6443
      secretName: aws-token
# Configures authentication and encryption.
security:
  # Defines a secret with custom credentials.
  endpointSecretName: endpoint-identities
  # Adds a custom TLS certificate to encrypt client connections.
  endpointEncryption:
      type: Secret
      certSecretName: tls-secret
# Sets container resources.
container:
  extraJvmOpts: "-XX:NativeMemoryTracking=summary"
  cpu: "1000m"
  memory: 1Gi
# Configures logging levels.
logging:
  categories:
    org.infinispan: debug
    org.jgroups: debug
    org.jgroups.protocols.TCP: error
    org.jgroups.protocols.relay.RELAY2: fatal
# Configures how the cluster is exposed on the network.
expose:
  type: LoadBalancer
# Configures affinity and anti-affinity strategies.
affinity:
  podAntiAffinity:
    preferredDuringSchedulingIgnoredDuringExecution:
    - weight: 100
      podAffinityTerm:
        labelSelector:
          matchLabels:
            app: infinispan-pod
            clusterName: example-infinispan
            infinispan_cr: example-infinispan
        topologyKey: "kubernetes.io/hostname"
```

3.4. 为数据网格资源添加标签

将键/值标签附加到 Data Grid Operator 创建和管理的 pod 和服务。这些标签可帮助您识别对象之间的关系，以更好地组织和监控数据网格资源。

注意

红帽订阅标签自动应用到 Data Grid pod。

流程

1. 打开 **Infinispan** CR 进行编辑。
2. 添加您想要 Data Grid Operator 的任何标签，以附加到带有 **metadata.annotations** 的资源。
3. 使用 **metadata.labels** 为您的标签添加值。

```
apiVersion: infinispan.org/v1
kind: Infinispan
metadata:
  annotations:
    # Add labels that you want to attach to services.
    infinispan.org/targetLabels: svc-label1, svc-label2
    # Add labels that you want to attach to pods.
    infinispan.org/podTargetLabels: pod-label1, pod-label2
  labels:
    # Add values for your labels.
    svc-label1: svc-value1
    svc-label2: svc-value2
    pod-label1: pod-value1
    pod-label2: pod-value2
    # The operator does not attach these labels to resources.
    my-label: my-value
    environment: development
```

4. 应用 **Infinispan** CR。

其他资源

- 标签和选择器
- 标签：Kubernetes 用户指南

第 4 章 调整容器规格

您可以分配 CPU 和内存资源，指定 JVM 选项，并为 Data Grid 节点配置存储。

4.1. JVM、CPU 和内存资源

```
spec:
  ...
  container:
    extraJvmOpts: "-XX:NativeMemoryTracking=summary" 1
    cpu: "1000m" 2
    memory: 1Gi 3
```

1 指定 JVM 选项。

2 将主机 CPU 资源分配给节点，以 CPU 单位计算。

3 将主机内存资源分配给节点，以字节为单位。

当 Data Grid Operator 创建 Data Grid 集群时，它使用 **spec.container.cpu** 和 **spec.container.memory**：

- 确保 OpenShift 有足够的容量来运行 Data Grid 节点。默认情况下，Data Grid Operator 从 OpenShift 调度程序请求 **512Mi 内存和** 0.5。
- 约束节点资源使用情况。**Data Grid Operator 将 cpu 和 memory 的值设置为资源限值。**

垃圾回收日志记录

默认情况下，Data Grid Operator 不会记录垃圾回收(GC)信息。您可以选择添加以下 JVM 选项，将 GC 消息定向到 stdout：

```
extraJvmOpts: "-Xlog:gc*:stdout:time,level,tags"
```

4.2. 存储资源

默认情况下，Data Grid Operator 为缓存服务和数据网格服务节点分配 1Gi 存储。您可以为 Data Grid 服务节点配置存储资源，但不能配置缓存服务节点。

```
spec:
  ...
  service:
    type: DataGrid
```

```
container:
  storage: 2Gi (1)
  storageClassName: my-storage-class (2)
```

(1) 为 Data Grid 服务节点配置存储大小。

(2) 指定用于持久性卷声明的 StorageClass 对象的名称。如果包含此字段，则必须指定现有存储类作为值。如果没有包括此字段，持久性卷声明将使用将 storageclass.kubernetes.io/is-default-class 注解设置为 true 的存储类。

持久性卷声明

Data Grid Operator 在以下位置挂载持久性卷：/opt/infinispan/server/data

注意

持久性卷声明使用 ReadWriteOnce (RWO) 访问模式。

第 5 章 停止并启动 DATA GRID 集群

使用 Data Grid Operator 停止并启动 Data Grid 集群。

缓存定义

Cache 服务和 Data Grid 服务都将永久缓存定义存储在持久性卷中，以便它们在集群重启后仍然可用。

data

如果您添加了基于文件的缓存存储，则数据网格服务节点可在集群关闭过程中将所有缓存条目写入持久性存储。

5.1. 关闭 DATA GRID 集群

关闭缓存服务节点会删除缓存中的所有数据。对于 Data Grid 服务节点，您应该为 Data Grid 服务节点配置存储大小，以确保持久性卷可以保存所有数据。

如果可用容器存储小于 Data Grid 服务节点可用的内存量，则数据网格会在关闭过程中将以下异常写入日志和数据丢失：

```
WARNING: persistent volume size is less than memory size. Graceful shutdown may not work.
```

流程

- 将 replicas 的值设置为 0 并应用更改。

```
spec:
  replicas: 0
```

5.2. 重启 DATA GRID 集群

在关闭后，完成以下步骤以重启 Data Grid 集群。

先决条件

对于 Data Grid 服务节点，您必须在关闭前重启具有相同节点的集群。例如，您可以关闭 6 个节点的集群。重启该集群时，必须将 6 指定为 spec.replicas 的值。

这允许 Data Grid 恢复集群中的数据分布。当集群中的所有节点都运行时，您可以添加或删除节点。

您可以找到 Data Grid 集群的正确节点数量，如下所示：

```
$ oc get infinispan example-infinispan -o=jsonpath='{.status.replicasWantedAtRestart}'
```

流程

- 将 spec.replicas 的值设置为集群的相应节点数，例如：

```
spec:
  replicas: 6
```

第 6 章 配置网络对数据网格的访问

公开数据网格集群，以便您可以访问数据网格控制台、Data Grid 命令行界面(CLI)、REST API 和 Hot Rod 端点。

6.1. 获取内部连接的服务

默认情况下，Data Grid Operator 创建一个服务，可从 OpenShift 上运行的客户端提供对 Data Grid 集群的访问。

此内部服务具有与 Data Grid 集群相同的名称，例如：

```
metadata:
  name: example-infinispan
```

流程

- 检查内部服务是否可用，如下所示：

```
$ oc get services

NAME                TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)
example-infinispan ClusterIP   192.0.2.0       <none>        11222/TCP
```

其他资源

- 网络服务

6.2. 通过负载平衡器公开数据网格

使用负载均衡器服务将 Data Grid 集群提供给在 OpenShift 外部运行的客户端。

注意

要使用未加密的 Hot Rod 客户端连接访问数据网格，您必须使用负载均衡器服务。

流程

1. 在 Infinispan CR 中包含 spec.expose。

2. 使用 spec.expose.type 将 LoadBalancer 指定为服务类型。

```
spec:
  ...
  expose:
    type: LoadBalancer (1)
    nodePort: 30000 (2)
```

(1) 通过端口 11222 上的负载均衡器服务公开网络上的 Data Grid。

(2) （可选）定义负载均衡器服务将流量转发到的节点端口。

3. 应用更改。

4. 验证 -external 服务是否可用。

```
$ oc get services | grep external

NAME                        TYPE           CLUSTER-IP    EXTERNAL-IP   PORT(S)
example-infinispan-external  LoadBalancer    192.0.2.24    hostname.com  11222/TCP
```

6.3. 通过节点端口公开数据网格

使用节点端口服务在网络上公开 Data Grid 集群。

流程

1. 在 Infinispan CR 中包含 spec.expose。

2. 使用 spec.expose.type 将 NodePort 指定为服务类型。

```
spec:
  ...
  expose:
    type: NodePort (1)
    nodePort: 30000 (2)
```

1. 通过节点端口服务在网络上公开数据网格。

2. 定义 Data Grid 被公开的端口。如果您没有定义端口，则平台会选择一个端口。

3. 应用更改。

4. 验证 -external 服务是否可用。

```
$ oc get services | grep external

NAME                        TYPE          CLUSTER-IP      EXTERNAL-IP   PORT(S)
example-infinispan-external  NodePort      192.0.2.24      <none>        11222:30000/TCP
```

6.4. 通过路由公开数据网格

使用带有 passthrough 加密的 OpenShift Route 使 Data Grid 集群在网络上可用。

流程

1. 在 Infinispan CR 中包含 spec.expose。

2. 使用 spec.expose.type 将 Route 指定为服务类型。

3. （可选）使用 spec.expose.host 添加主机名。

```
spec:
  ...
  expose:
```

```
type: Route 1
host: www.example.org 2
```

1 通过 OpenShift 路由在网络上公开数据网格。

2 （可选）指定公开 Data Grid 的主机名。

4. 应用更改。

5. 验证路由是否可用。

```
$ oc get routes

NAME                CLASS    HOSTS   ADDRESS   PORTS   AGE
example-infinispan  <none>   *                 443     73s
```

路由端口

当您创建路由时，它会在网络上公开一个端口，该端口接受客户端连接并将流量重定向到侦听端口 11222 的 Data Grid 服务。

路由可用的端口取决于您是否使用加密。

端口	描述
80	加密被禁用。
443	加密已启用。

第 7 章 保护数据网格连接

使用身份验证和加密保护客户端连接，以防止网络入侵和保护您的数据。

7.1. 配置身份验证

应用程序用户需要凭证才能访问 Data Grid 集群。您可以使用默认生成的凭证或自行添加。

7.1.1. 默认凭证

Data Grid Operator 生成存储在名为 example-infinispan-generated-secret的身份验证 secret 中的 base64 编码默认凭证

用户名	描述
developer	默认应用程序用户。
operator	与 Data Grid 集群交互的内部用户。

7.1.2. 检索凭证

从身份验证 secret 获取凭证以访问 Data Grid 集群。

流程

- 从身份验证 secret 检索凭证，如下例所示：

```
$ oc get secret example-infinispan-generated-secret
```

Base64-decode 凭证。

```
$ oc get secret example-infinispan-generated-secret \
-o jsonpath="{.data.identities\.yaml}" | base64 --decode

credentials:
- username: developer
  password: dIRs5cAAsHIeeRIL
- username: operator
  password: uMBo9CmEdEduYk24
```

7.1.3. 添加自定义凭证

使用自定义凭据配置对 Data Grid 集群端点的访问。

流程

1. 使用要添加的凭证创建一个 identity .yaml 文件。

```
credentials:
- username: testuser
  password: testpassword
- username: operator
  password: supersecretoperatorpassword
```

重要

identity.yaml 必须包含 operator 用户。

2. 从 identity .yaml 创建身份验证 secret。

```
$ oc create secret generic --from-file=identities.yaml connect-secret
```

3. 在 Infinispan CR 中使用 spec.security.endpointSecretName 指定身份验证 secret，然后应用更改。

```
spec:
  ...
  security:
    endpointSecretName: connect-secret (1)
```

(1) 指定包含您的凭证的身份验证 secret 的名称。

修改 spec.security.endpointSecretName 会触发集群重启。您可以监控 Data Grid 集群，因为 Data Grid Operator 应用更改：

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

7.2. 配置加密

使用 Red Hat OpenShift 服务证书或自定义 TLS 证书加密客户端和 Data Grid 节点之间的连接。

7.2.1. 使用 Red Hat OpenShift Service Certificates 加密

Data Grid Operator 会自动生成由 Red Hat OpenShift 服务 CA 签名的 TLS 证书。然后，Data Grid Operator 将证书和密钥存储在 secret 中，以便您可以检索它们并用于远程客户端。

如果 Red Hat OpenShift 服务 CA 可用，Data Grid Operator 会将以下 spec.security.endpointEncryption 配置添加到 Infinispan CR 中：

```
spec:
  ...
  security:
    endpointEncryption:
      type: Service
      certServiceName: service.beta.openshift.io 1
      certSecretName: example-infinispan-cert-secret 2
```

1

指定 Red Hat OpenShift Service。

2

以 PEM 格式命名包含服务证书 tls.crt 和密钥 tls.key 的 secret。如果没有指定名称，Data Grid Operator 将使用 < cluster_name>-cert-secret。

注意

服务证书使用 Data Grid 集群的内部 DNS 名称作为通用名称(CN)，例如：

subject: CN = example-infinispan.mynamespace.svc

因此，服务证书只能在 OpenShift 中完全信任。如果要加密与 OpenShift 外部运行的客户端的连接，您应该使用自定义 TLS 证书。

服务证书在一年内有效，并在过期前自动替换。

7.2.2. 检索 TLS 证书

从加密 secret 获取 TLS 证书以创建客户端信任存储。

- 从加密 secret 检索 tls.crt，如下所示：

```
$ oc get secret example-infinispan-cert-secret \
-o jsonpath='{.data.tls\.crt}' | base64 --decode > tls.crt
```

7.2.3. 禁用加密

您可以禁用加密，因此客户端不需要 TLS 证书来建立与 Data Grid 的连接。

重要

Data Grid 不建议在生产环境中禁用加密，其中端点通过 spec.expose.type 在 OpenShift 集群外公开。

流程

- 将 None 设置为 Infinispan CR 中的 spec.security.endpointEncryption.type 字段的值，然后应用更改。

```
spec:
  ...
```

```
security:
  endpointEncryption:
        type: None 1
```

1

禁用 Data Grid 端点的加密。

7.2.4. 使用自定义 TLS 证书

使用自定义 PKCS12 密钥存储或 TLS 证书/密钥对来加密客户端和数据网格集群间的连接。

先决条件

- 创建密钥存储或证书 secret。请参阅：
- 证书 Secret
- 密钥存储 secret

流程

1. 将加密 secret 添加到 OpenShift 命名空间中，例如：

```
$ oc apply -f tls_secret.yaml
```

2. 在 Infinispan CR 中使用 spec.security.endpointEncryption 指定加密 secret，然后应用更改。

```
spec:
  ...
  security:
    endpointEncryption: 1
          type: Secret 2
          certSecretName: tls-secret 3
```

1

加密到 Data Grid 端点的流量。

2

配置 Data Grid 以使用包含加密证书的 secret。

3

将加密 secret 命名为。

7.2.4.1. 证书 Secret

```
apiVersion: v1
kind: Secret
metadata:
  name: tls-secret
type: Opaque
data:
    tls.key:  "LS0tLS1CRUdJTiBQUk ..." 1
    tls.crt: "LS0tLS1CRUdJTiBDRVI ..." 2
```

1

添加以 base64 编码的 TLS 密钥。

2

添加以 base64 编码的 TLS 证书。

7.2.4.2. 密钥存储 secret

```
apiVersion: v1
kind: Secret
metadata:
  name: tls-secret
type: Opaque
stringData:
    alias: server 1
    password: password 2
data:
    keystore.p12:  "MIIKDgIBAzCCCdQGCSqGSIb3DQEHA..." 3
```

1

指定密钥存储的别名。

2

指定密钥存储的密码。

3

添加 base64 编码的密钥存储。

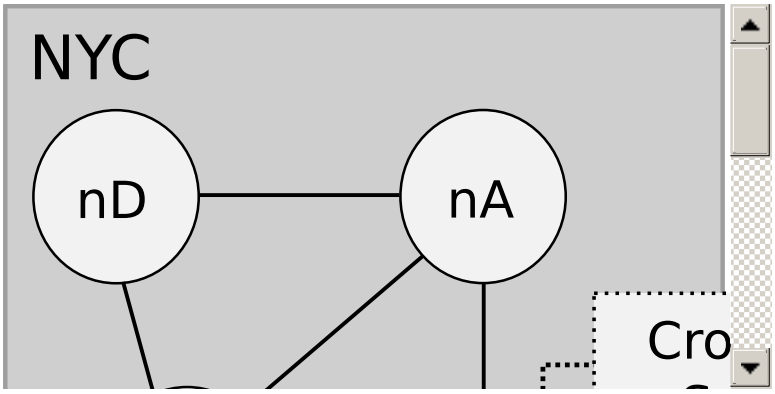
第 8 章 配置跨站点复制

设置全局 Data Grid 集群，以便在站点间备份数据。

8.1. 使用 DATA GRID OPERATOR 跨站点复制

如果您在单独的位置运行 Data Grid 集群，请使用 Data Grid Operator 连接它们，以便您可以在站点间备份数据。

例如，在下图中，Data Grid Operator 在 New York City NYC 的数据中心管理一个 Data Grid 集群。在伦敦的另一台数据中心，LON，Data Grid Operator 还管理一个 Data Grid 集群。



Data Grid Operator 使用 Kubernetes API 在 NYC 和 LON 的 OpenShift Container Platform 集群间建立安全连接。然后，Data Grid Operator 会创建跨站点复制服务，以便数据可以在不同位置备份数据。

每个 Data Grid 集群都有一个站点 master 节点，用于协调所有备份请求。Data Grid Operator 标识站点 master 节点，以便通过跨站点复制服务的所有流量都进入站点 master。

如果当前站点 master 节点离线，则新节点变为站点 master。Data Grid Operator 会自动找到新的站点 master 节点，并更新跨站点复制服务来向它转发备份请求。

8.2. 创建服务帐户令牌

在每个作为备份位置的每个 OpenShift 集群上生成服务帐户令牌。集群使用这些令牌来相互进行身份验证，以便 Data Grid Operator 可以创建跨站点复制服务。

流程

1. 登录 OpenShift 集群。

2. 创建一个服务帐户。

   例如，在 LON 创建服务帐户：

```
$ oc create sa lon
serviceaccount/lon created
```

3. 使用以下命令在服务帐户中添加 view 角色：

```
$ oc policy add-role-to-user view system:serviceaccount:<namespace>:lon
```

4. 在其他 OpenShift 集群上重复上述步骤。

其他资源

在应用程序中使用服务帐户

8.3. 交换服务帐户令牌

在 OpenShift 集群中创建服务帐户令牌后，您可以将其添加到每个备份位置的 secret 中。例如，在 LON 中，您可以为 NYC 添加服务帐户令牌。在 NYC 中，您可以为 LON 添加服务帐户令牌。

先决条件

- 从每个服务帐户获取令牌。

  使用以下命令或从 OpenShift Web 控制台获取令牌：

```
$ oc sa get-token lon

eyJhbGciOiJSUzI1NiIsImtpZCI6IiJ9...
```

流程

1. 登录 OpenShift 集群。

2. 使用以下命令为备份位置添加服务帐户令牌：

```
$ oc create secret generic <token-name> --from-literal=token=<token>
```

例如，登录到 NYC 的 OpenShift 集群，并创建一个 lon-token secret，如下所示：

```
$ oc create secret generic lon-token --from-
literal=token=eyJhbGciOiJSUzI1NiIsImtpZCI6IiJ9...
```

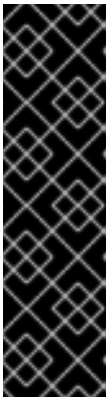
3. 在其他 OpenShift 集群上重复上述步骤。

8.4. 为跨站点复制配置数据网格集群

将 Data Grid 集群配置为备份位置，以便它们可以通过专用的 JGroups 传输通道来复制数据。

先决条件

- 创建包含每个备份位置的服务帐户令牌的 secret。
- 确保所有集群都是 Data Grid 服务节点。
- 确保 OpenShift 项目名称匹配。



重要

要执行跨站点复制，Data Grid Operator 需要 Data Grid 集群具有相同的名称并在匹配的命名空间中运行。

例如，您可以在名为 xsite-cluster 的项目中，在 LON 上创建集群。位于 NYC 的集群还必须在名为 xsite-cluster 的项目中运行。

流程

1. 为每个 Data Grid 集群创建一个 Infinispan CR。

2. 使用 metadata.name 为每个 Data Grid 集群指定匹配名称。

3. 使用 spec.service.sites.local.name 指定本地站点的名称。

4. 使用 spec.service.sites.local.expose.type 设置本地站点的 expose 服务类型。

5. 为每个 Data Grid 集群提供名称、URL 和 secret，它们充当 spec.service.sites.locations 的备份位置。

以下是 LON 和 NYC 的 Infinispan CR 定义示例：

- LON

```
apiVersion: infinispan.org/v1
kind: Infinispan
metadata:
  name: example-infinispan
spec:
  replicas: 3
  service:
    type: DataGrid
    sites:
      local:
        name: LON
        expose:
          type: LoadBalancer
      locations:
        - name: LON
          url: openshift://api.rhdg-lon.openshift-aws.myhost.com:6443
          secretName: lon-token
        - name: NYC
          url: openshift://api.rhdg-nyc.openshift-aws.myhost.com:6443
          secretName: nyc-token
```

- NYC

```
apiVersion: infinispan.org/v1
kind: Infinispan
metadata:
```

```
  name: example-infinispan
spec:
  replicas: 2
  service:
    type: DataGrid
    sites:
      local:
        name: NYC
        expose:
          type: LoadBalancer
      locations:
        - name: NYC
          url: openshift://api.rhdg-nyc.openshift-aws.myhost.com:6443
          secretName: nyc-token
        - name: LON
          url: openshift://api.rhdg-lon.openshift-aws.myhost.com:6443
          secretName: lon-token
```

6. 为跨站点复制调整日志记录级别，如下所示：

```
...
  logging:
    categories:
      org.jgroups.protocols.TCP: error
      org.jgroups.protocols.relay.RELAY2: fatal
```

上述配置减少了 JGroups TCP 和 RELAY2 协议的日志，以减少有关集群备份操作的大量消息，这可能会导致大量使用容器存储的日志文件。

7. 使用任何其他 Data Grid 服务资源配置节点。

8. 应用 Infinispan CR。

9. 检查节点日志以验证 Data Grid 集群是否形成跨站点视图，例如：

```
$ oc logs example-infinispan-0 | grep x-site

INFO  [org.infinispan.XSITE] (jgroups-5,example-infinispan-0-<id>) ISPN000439: Received new x-site view: [NYC]
INFO  [org.infinispan.XSITE] (jgroups-7,example-infinispan-0-<id>) ISPN000439: Received new x-site view: [NYC, LON]
```

后续步骤

如果您的集群有一个跨站点视图，您可以开始向缓存添加备份位置。

其他资源

- 跨站点复制资源
- 向缓存添加备份位置
- 跨站点复制的数据网格指南

8.4.1. 跨站点复制资源

```
spec:
  ...
  service:
    type: DataGrid (1)
    sites:
      local:
        name: LON (2)
        expose:
          type: LoadBalancer (3)
      locations: (4)
      - name: LON (5)
        url: openshift://api.site-a.devcluster.openshift.com:6443 (6)
        secretName: lon-token (7)
      - name: NYC
        url: openshift://api.site-b.devcluster.openshift.com:6443
        secretName: nyc-token
  logging:
    categories:
      org.jgroups.protocols.TCP: error (8)
      org.jgroups.protocols.relay.RELAY2: fatal (9)
```

(1) 指定数据网格服务。Data Grid 仅支持通过 Data Grid 服务集群进行跨站点复制。

(2) 为 Data Grid 集群命名本地站点。

(3)

指定 LoadBalancer 作为处理备份位置间通信的服务。

4

为所有备份位置提供连接信息。

5

指定与 spec.service.sites.local.name 匹配的备份位置。

6

指定备份位置的 OpenShift API 的 URL。

7

指定包含备份站点的服务帐户令牌的 secret。

8

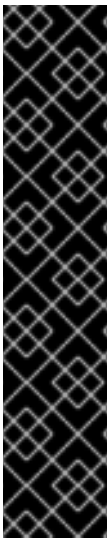
记录 JGroups TCP 协议的错误消息。

9

记录 JGroups RELAY2 协议的致命消息。

第 9 章 使用 DATA GRID OPERATOR 创建缓存

使用 Cache CR 添加带有 Data Grid Operator 的缓存配置，并控制 Data Grid 如何存储数据。



重要

使用 Data Grid Operator 创建缓存作为技术预览提供。

红帽产品服务级别协议(SLA)不支持技术预览功能或功能，且可能无法完成。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

红帽技术预览功能支持范围

使用 Cache CR 时，会应用以下规则：

- 缓存 CR 仅适用于 Data Grid 服务节点。
- 您可以为每个 Cache CR 创建一个缓存。
- 如果您的 Cache CR 同时包含模板和 XML 配置，Data Grid Operator 将使用模板。
- 如果在 OpenShift Web 控制台中编辑缓存，则更改会反映在用户界面中，但不会对 Data Grid 集群产生影响。您不能编辑缓存。要更改缓存配置，您必须首先通过控制台或 CLI 删除缓存，然后重新创建缓存。
- 删除 OpenShift Web 控制台中的 缓存 CR 不会从 Data Grid 集群中删除缓存。您必须通过控制台或 CLI 删除缓存。

9.1. 为 DATA GRID OPERATOR 添加凭证

Data Grid Operator 必须通过 Data Grid 服务集群进行身份验证来创建缓存。您可以在 secret 中添加凭证，以便 Data Grid Operator 可以在创建缓存时访问集群。

以下流程解释了如何在新 secret 中添加凭证。如果您已经有一个包含凭证的自定义 secret，您可以使用该 secret 而不是创建新 secret。

流程

1. 定义 Secret 对象类型，它提供在 StringData 映射中用于访问 Data Grid 服务集群的有效用户凭证。

   例如，创建一个 basic-auth.yaml 文件，它为 developer 用户提供凭证，如下所示：

```
apiVersion: v1
stringData:
  username: developer (1)
  password: G8ZdJvSaY3lOOwfM (2)
kind: Secret
metadata:
  name: basic-auth (3)
type: Opaque
```

   (1) 命名可以创建缓存的用户。

   (2) 指定与用户对应的密码。

   (3) 指定 secret 的名称。

2. 从文件创建 secret，如下例所示：

```
$ oc apply -f basic-auth.yaml
```

9.1.1. 使用自定义凭证 secret

Data Grid Operator 要求凭证作为 secret 中 用户名和密码 键的值存在。如果您有包含 Data Grid 凭证的自定义 secret，但使用不同的密钥名称，您可以在 Cache CR 中覆盖这些名称。

例如，您有一个名为"my-credentials"的 secret，其中包含一个 Data Grid 用户及其密码列表，如下所示：

```
stringData:
  app_user1: spock
  app_user1_pw: G8ZdJvSaY3lOOwfM
  app_user2: jim
  app_user2_pw: zTzz2gVyyF4JsYsH
```

流程

- 在 Cache CR 中，使用 用户名和密码 覆盖自定义密钥名称，如下所示：

```
spec:
  adminAuth:
    username:
      key: app_user1 (1)
      name: my-credentials (2)
    password:
      key: app_user1_pw (3)
      name: my-credentials
```

(1) 使用用户名 覆盖 app_user1 密钥名称。

(2) 指定自定义凭证 secret 的名称。

(3) 使用密码 覆盖 app_user1_pw 键名称。

9.2. 从 XML 创建数据网格缓存

完成以下步骤，使用有效的 infinispan.xml 缓存定义在 Data Grid 服务集群中创建缓存。

先决条件

- 创建一个 secret，其中包含用于访问 Data Grid 集群的有效用户凭证。

流程

1. 创建一个包含您要创建的 XML 缓存定义的 Cache CR。

```
apiVersion: infinispan.org/v2alpha1
kind: Cache
metadata:
  name: mycachedefinition (1)
spec:
  adminAuth: (2)
    secretName: basic-auth
  clusterName: example-infinispan (3)
  name: mycache (4)
  template: <infinispan><cache-container><distributed-cache name="mycache"
mode="SYNC"><persistence><file-store/></persistence></distributed-cache></cache-
container></infinispan> (5)
```

(1) 为 Cache CR 命名。

(2) 指定为凭证提供用户名和密码 密钥或 自定义凭证 secret 覆盖的 secret。

(3) 指定您希望 Data Grid Operator 创建缓存的目标 Data Grid 集群名称。

(4) 在 Data Grid 集群中命名缓存。

(5) 指定用于创建缓存的 XML 缓存定义。请注意，name 属性将被忽略。只有 spec.name 适用于生成的缓存。

2. 应用 Cache CR，例如：

```
$ oc apply -f mycache.yaml
cache.infinispan.org/mycachedefinition created
```

9.3. 从模板创建数据网格缓存

完成以下步骤，使用缓存配置模板在 Data Grid 服务集群中创建缓存。

先决条件

- 创建一个 secret，其中包含用于访问 Data Grid 集群的有效用户凭证。
- 识别您要用于缓存的缓存配置模板。您可以在 Data Grid Console 中找到可用配置模板列表。

流程

1. 创建一个 Cache CR，用于指定您要使用的模板的名称。

   例如，以下 CR 创建名为"mycache"的缓存，它使用 org.infinispan.DIST_SYNC 缓存配置模板：

```
apiVersion: infinispan.org/v2alpha1
kind: Cache
metadata:
  name: mycachedefinition (1)
spec:
  adminAuth: (2)
    secretName: basic-auth
  clusterName: example-infinispan (3)
  name: mycache (4)
  templateName: org.infinispan.DIST_SYNC (5)
```

(1) 为 Cache CR 命名。

(2) 指定为凭证提供用户名和密码 密钥或 自定义凭证 secret 覆盖的 secret。

(3) 指定您希望 Data Grid Operator 创建缓存的目标 Data Grid 集群名称。

(4) 将 Data Grid 缓存命名为 instance。

5 指定用于创建缓存的 infinispan.org 缓存配置模板。

2. 应用 Cache CR，例如：

```
$ oc apply -f mycache.yaml
cache.infinispan.org/mycachedefinition created
```

9.4. 向缓存添加备份位置

当您将 Data Grid 集群配置为执行跨站点复制时，您可以在缓存配置中添加备份位置。

流程

1. 为每个站点创建具有相同名称的缓存配置。

   每个站点的缓存配置可以使用不同的缓存模式和备份策略。Data Grid 基于缓存名称复制数据。

2. 配置备份位置，以使用 take-offline 元素自动离线。

   a. 在备份位置使用 min-wait 属性离线前设置时间（以毫秒为单位）。

3. 定义任何其他有效的缓存配置。

4. 将备份位置添加到全局集群中所有站点的指定缓存中。

   例如，如果您将 LON 添加为 NYC 的备份，您应该将 NYC 添加为 LON 的备份。

以下配置示例显示缓存的备份位置：

- NYC

```
<infinispan>
  <cache-container>
    <distributed-cache name="customers">
      <encoding media-type="application/x-protostream"/>
      <backups>
        <backup site="LON" strategy="SYNC">
          <take-offline min-wait="120000"/>
        </backup>
      </backups>
    </distributed-cache>
  </cache-container>
</infinispan>
```

- **LON**

```
<infinispan>
  <cache-container>
    <replicated-cache name="customers">
      <encoding media-type="application/x-protostream"/>
      <backups>
        <backup site="NYC" strategy="ASYNC" >
            <take-offline min-wait="120000"/>
          </backup>
      </backups>
    </replicated-cache>
  </cache-container>
</infinispan>
```

其他资源

- 为跨站点复制配置集群
- 跨站点复制的数据网格指南

9.4.1. 使用 Taking Backup Locations Offline 的性能注意事项

当远程站点不可用时，备份位置可以自动离线。这可防止节点尝试将数据复制到离线备份位置，这可能会对集群造成性能影响，因为它会导致错误。

您可以配置备份位置离线前等待的时长。良好的 thumb 规则是一两分钟。但是，您应该测试不同的等待周期，并评估其性能影响，以确定部署的正确值。

例如，当 OpenShift 终止站点 master pod 时，该备份位置在短时间内不可用，直到 Data Grid Operator 选择一个新的站点 master。在这种情况下，如果最小等待时间不够长，则备份位置离线。然

后，您需要将这些备份位置上线并执行状态传输操作，以确保数据同步。

同样，如果最小等待时间太长，节点 CPU 用量会因为失败的备份尝试而增加，这可能会导致性能下降。

9.5. 添加持久性缓存存储

您可以将单一文件缓存存储添加到 Data Grid 服务节点上，将数据保存到持久性卷中。

您可以使用 persistence 元素将缓存存储配置为 Data Grid 缓存定义的一部分，如下所示：

```
<persistence>
  <file-store/>
</persistence>
```

然后，数据网格会在 /opt/infinispan/server/data 目录中创建单一文件缓存存储 .dat 文件。

流程

- 在缓存配置中添加缓存存储，如下所示：

```
<infinispan>
 <cache-container>
  <distributed-cache name="customers" mode="SYNC">
   <encoding media-type="application/x-protostream"/>
   <persistence>
    <file-store/>
   </persistence>
  </distributed-cache>
 </cache-container>
</infinispan>
```

其他资源

- 存储资源

第 10 章 建立远程客户端连接

从 Data Grid Console、命令行界面(CLI)和远程客户端连接到 Data Grid 集群。

10.1. 客户端连接详情

在连接到 Data Grid 之前，您需要检索以下信息：

- 服务主机名
- 端口
- 身份验证凭证
- TLS 证书（如果使用加密）

服务主机名

服务主机名取决于您如何在网络上公开数据网格，或者您的客户端在 OpenShift 上运行。

对于在 OpenShift 上运行的客户端，您可以使用 Data Grid Operator 创建的内部服务的名称。

对于在 OpenShift 外部运行的客户端，如果您使用负载均衡器，服务主机名是位置 URL。对于节点端口服务，服务主机名是节点主机名。对于路由，服务主机名是自定义主机名或系统定义的主机名。

端口

OpenShift 和负载平衡器上的客户端连接使用端口 11222。

节点端口服务使用 30000 到 60000 范围内的端口。路由使用端口 80 （加密）或 443 （加密）。

其他资源

- 配置网络对数据网格的访问
- 检索凭证
- 检索 TLS 证书

10.2. 创建 DATA GRID CACHES

要在 OpenShift 上运行 Data Grid 时创建缓存，您可以：

- 使用 Cache CR。
- 如果不使用 Cache CR，则一次使用 Data Grid CLI 创建多个缓存。
- 访问 Data Grid 控制台，并以 XML 或 JSON 格式创建缓存，作为 Cache CR 或 Data Grid CLI 的替代方案。
- 仅在需要时，使用 Hot Rod 客户端以编程方式或通过每个缓存属性创建缓存。

其他资源

使用 Data Grid Operator 创建缓存

10.3. 使用 DATA GRID CLI 连接

使用命令行界面(CLI)连接到您的数据网格集群并执行管理操作。

CLI 作为服务器分发的一部分提供，您可以在本地主机上运行以建立与 OpenShift 上 Data Grid 集群的远程连接。

注意

可以在 Data Grid 节点上打开远程 shell 并访问 CLI。

```
$ oc rsh example-infinispan-0
```

但是，以这种方式使用 CLI 消耗分配给容器的内存，这可能会导致内存不足异常。

10.3.1. 使用 Data Grid CLI 创建缓存

使用 CLI 将缓存添加到您的 Data Grid 集群。

先决条件

- 下载服务器分发版本，以便您可以运行 CLI。
- 检索所需的客户端连接详细信息。

流程

1. 使用 XML 或 JSON 格式的缓存配置创建文件，例如：

```
cat > infinispan.xml<<EOF
<infinispan>
  <cache-container>
    <distributed-cache name="mycache">
      <encoding>
        <key media-type="application/x-protostream"/>
        <value media-type="application/x-protostream"/>
      </encoding>
    </distributed-cache>
  </cache-container>
</infinispan>
EOF
```

2. 创建与 Data Grid 集群的 CLI 连接。

```
$ bin/cli.sh -c https://$SERVICE_HOSTNAME:$PORT --trustall
```

将 $SERVICE_HOSTNAME:$PORT 替换为在网络上提供 Data Grid 的主机名和端口。

3. 出现提示时，输入您的 Data Grid 凭证。

4. 使用 create cache 命令和 --file 选项添加缓存。

```
[//containers/default]> create cache --file=infinispan.xml mycache
```

5. 使用 ls 命令验证缓存是否存在。

```
[//containers/default]> ls caches
mycache
```

6. （可选）使用 describe 命令检索缓存配置。

```
[//containers/default]> describe caches/mycache
```

其他资源

- 下载服务器发布
- 使用 Data Grid 命令行界面

10.3.2. 在批处理中创建缓存

使用 Data Grid CLI 添加带有批处理操作的多个缓存。

先决条件

- 下载服务器分发版本，以便您可以运行 CLI。
- 检索所需的客户端连接详细信息。

流程

1. 至少创建一个文件，其缓存配置采用 XML 或 JSON 格式。

2. 创建批处理文件，例如：

```
cat > caches.batch<<EOF
echo "connecting"
connect --username=developer --password=dlRs5cAAsHleeRIL
echo "creating caches..."
create cache firstcache --file=infinispan-one.xml
create cache secondcache --file=infinispan-two.xml
create cache thirdcache --file=infinispan-three.xml
create cache fourthcache --file=infinispan-four.xml
echo "verifying caches"
ls caches
EOF
```

3. 使用 CLI 创建缓存。

```
$ bin/cli.sh -c https://$SERVICE_HOSTNAME:$PORT --trustall -f /tmp/caches.batch
```

将 $SERVICE_HOSTNAME:$PORT 替换为在网络上提供 Data Grid 的主机名和端口。

其他资源

- 下载服务器发布
- 使用 Data Grid 命令行界面

10.4. 访问数据网格控制台

访问控制台以创建缓存、执行管理操作并监控您的数据网格集群。

先决条件

- 在网络上公开 Data Grid，以便您可以通过浏览器访问控制台。
  例如，配置负载均衡器服务或创建路由。

流程

1. 从位于 $SERVICE_HOSTNAME 的任何浏览器访问控制台：$PORT.

   将 $SERVICE_HOSTNAME:$PORT 替换为在网络上提供 Data Grid 的主机名和端口。

2. 出现提示时，输入您的 Data Grid 凭证。

10.5. 热 ROD 客户端

hot Rod 是一个二进制 TCP 协议，Data Grid 提供带有远程客户端的高性能数据传输功能。

客户端智能

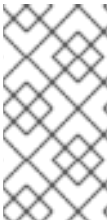
客户端智能指的是 Hot Rod 协议提供的机制，以便客户端可以定位并发送请求到 Data Grid 节点。

在 OpenShift 上运行的热 Rod 客户端可以访问 Data Grid 节点的内部 IP 地址，以便您可以使用任何客户端智能。建议默认智能 HASH_DISTRIBUTION_AWARE，因为它允许客户端将请求路由到主所有者，从而提高性能。

在 OpenShift 外部运行的热 Rod 客户端必须使用 BASIC 智能。

10.5.1. 热 Rod 配置 API

您可以使用 ConfigurationBuilder 接口以编程方式配置 Hot Rod 客户端连接。



注意

$SERVICE_HOSTNAME:$PORT 表示允许访问您的数据网格集群的主机名和端口。您应该使用环境的实际主机名和端口替换这些变量。

在 OpenShift 中

在 OpenShift 上运行的热 Rod 客户端可使用以下配置：

```
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("$SERVICE_HOSTNAME")
            .port(ConfigurationProperties.DEFAULT_HOTROD_PORT)
           .security().authentication()
            .username("username")
            .password("password")
            .realm("default")
            .saslQop(SaslQop.AUTH)
            .saslMechanism("SCRAM-SHA-512")
           .ssl()
            .sniHostName("$SERVICE_HOSTNAME")
            .trustStorePath("/var/run/secrets/kubernetes.io/serviceaccount/service-ca.crt");
```

外部 OpenShift

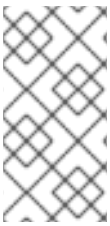
在 OpenShift 外部运行的热 Rod 客户端可使用以下配置：

```
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("password")
            .realm("default")
            .saslQop(SaslQop.AUTH)
            .saslMechanism("SCRAM-SHA-512")
           .ssl()
            .sniHostName("$SERVICE_HOSTNAME")
            .trustStorePath("/path/to/tls.crt");
    builder.clientIntelligence(ClientIntelligence.BASIC);
```

10.5.2. 热 Rod 客户端属性

您可以使用应用程序 classpath 上的 hotrod-client.properties 文件配置 Hot Rod 客户端连接。



注意

$SERVICE_HOSTNAME:$PORT 表示允许访问您的数据网格集群的主机名和端口。您应该使用环境的实际主机名和端口替换这些变量。

在 OpenShift 中

在 OpenShift 上运行的热 Rod 客户端可使用以下属性：

```
# 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

# Encryption
infinispan.client.hotrod.sni_host_name=$SERVICE_HOSTNAME
# Path to the TLS certificate.
# Clients automatically generate trust stores from certificates.
infinispan.client.hotrod.trust_store_path=/var/run/secrets/kubernetes.io/serviceaccount/service-ca.crt
```

外部 OpenShift

在 OpenShift 外部运行的热 Rod 客户端可使用以下属性：

```
# 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

# Encryption
infinispan.client.hotrod.sni_host_name=$SERVICE_HOSTNAME
# Path to the TLS certificate.
# Clients automatically generate trust stores from certificates.
infinispan.client.hotrod.trust_store_path=tls.crt
```

10.5.3. 使用 Hot Rod 客户端创建缓存

您可以使用 Hot Rod 客户端在 OpenShift 上运行的 Data Grid 集群上远程创建缓存。但是，Data Grid 建议使用 Data Grid 控制台、CLI 或带有 Cache CR 而不是 Hot Rod 客户端创建缓存。

以编程方式创建缓存

以下示例演示了如何将缓存配置添加到 ConfigurationBuilder 中，然后使用 RemoteCacheManager 创建它们：

```
import org.infinispan.client.hotrod.DefaultTemplate;
import org.infinispan.client.hotrod.RemoteCache;
import org.infinispan.client.hotrod.RemoteCacheManager;
...

    builder.remoteCache("my-cache")
        .templateName(DefaultTemplate.DIST_SYNC);
    builder.remoteCache("another-cache")
        .configuration("<infinispan><cache-container><distributed-cache name=\"another-
cache\"><encoding media-type=\"application/x-protostream\"/></distributed-cache></cache-
container></infinispan>");
    try (RemoteCacheManager cacheManager = new RemoteCacheManager(builder.build())) {
    // Get a remote cache that does not exist.
    // Rather than return null, create the cache from a template.
    RemoteCache<String, String> cache = cacheManager.getCache("my-cache");
    // Store a value.
    cache.put("hello", "world");
    // Retrieve the value and print it.
    System.out.printf("key = %s\n", cache.get("hello"));
```

本例演示了如何使用 XMLStringConfiguration （） 方法创建名为 CacheWithXMLConfiguration 的缓存，将缓存配置作为 XML 传递：

```
import org.infinispan.client.hotrod.RemoteCacheManager;
import org.infinispan.commons.configuration.XMLStringConfiguration;
...

private void createCacheWithXMLConfiguration() {
   String cacheName = "CacheWithXMLConfiguration";
   String xml = String.format("<infinispan>" +
                              "<cache-container>" +
                              "<distributed-cache name=\"%s\" mode=\"SYNC\">" +
                                "<encoding media-type=\"application/x-protostream\"/>" +
                                "<locking isolation=\"READ_COMMITTED\"/>" +
                                "<transaction mode=\"NON_XA\"/>" +
                                "<expiration lifespan=\"60000\" interval=\"20000\"/>" +
                              "</distributed-cache>" +
                              "</cache-container>" +
                            "</infinispan>"
                            , cacheName);
   manager.administration().getOrCreateCache(cacheName, new
XMLStringConfiguration(xml));
   System.out.println("Cache with configuration exists or is created.");
}
```

使用 Hot Rod 客户端属性

当您为指定缓存调用 cacheManager.getCache （） 调用时，Data Grid 从 Hot Rod 客户端属性创建

它们，而不是返回 null。

在 Hot Rod 客户端属性中添加缓存配置，如下例所示：

```
# Add cache configuration
infinispan.client.hotrod.cache.my-cache.template_name=org.infinispan.DIST_SYNC
infinispan.client.hotrod.cache.another-cache.configuration=<infinispan><cache-container>
<distributed-cache name=\"another-cache\"/></cache-container></infinispan>
infinispan.client.hotrod.cache.my-other-cache.configuration_uri=file:/path/to/configuration.xml
```

10.6. 访问 REST API

Data Grid 提供了一个 RESTful 接口，您可以使用 HTTP 客户端与之交互。

先决条件

- 在网络上公开 Data Grid，以便您可以访问 REST API。
例如，配置负载均衡器服务或创建路由。

流程

- 使用位于 $SERVICE_HOSTNAME:$PORT/rest/v2 的任何 HTTP 客户端访问 REST API。

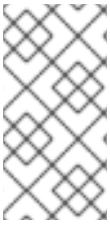
  将 $SERVICE_HOSTNAME:$PORT 替换为在网络上提供 Data Grid 的主机名和端口。

其他资源

- Data Grid REST API

10.7. 在缓存服务节点中添加缓存

缓存服务节点包括带有推荐的设置的默认缓存配置。此默认缓存可让您开始使用 Data Grid，而无需创建缓存。



注意

由于默认缓存提供了推荐的设置，因此您应该只创建缓存作为默认设置的副本。如果您希望多个自定义缓存，您应该创建 Data Grid 服务节点而不是缓存服务节点。

流程

- 访问 Data Grid 控制台，并以 XML 或 JSON 格式提供默认配置的副本。
- 使用 Data Grid CLI 从默认缓存创建副本，如下所示：

```
[//containers/default]> create cache --template=default mycache
```

10.7.1. 默认缓存配置

缓存服务节点的默认缓存如下：

```
<infinispan>
  <cache-container>
    <distributed-cache name="default" 1
                       mode="SYNC" 2
                       owners="2"> 3
      <memory storage="OFF_HEAP" 4
              max-size="<maximum_size_in_bytes>" 5
              when-full="REMOVE" /> 6
      <partition-handling when-split="ALLOW_READ_WRITES" 7
                          merge-policy="REMOVE_ALL"/> 8
    </distributed-cache>
  </cache-container>
</infinispan>
```

1. 将缓存实例命名为 "default"。
2. 使用同步分发在集群中存储数据。
3. 在集群中配置每个缓存条目的两个副本。

4

将缓存条目存储为原生内存(off-heap)中的字节数。

5

定义数据容器的最大大小（以字节为单位）。Data Grid Operator 会在创建节点时计算最大大小。

6

驱除缓存条目来控制数据容器的大小。您可以启用自动扩展，以便 Data Grid Operator 在内存用量而不是删除条目时添加节点。

7

命名冲突解析策略，允许对缓存条目进行读写操作，即使网段所有者位于不同的分区中。

8

指定在 Data Grid 检测到冲突时从缓存中删除条目的合并策略。

第 11 章 使用 PROMETHEUS 监控 DATA GRID

Data Grid 公开了一个指标端点，为 Prometheus 提供统计信息和事件。

11.1. 创建 PROMETHEUS SERVICE MONITOR

定义服务监控实例，以配置 Prometheus 来监控您的 Data Grid 集群。

先决条件

- 在 OpenShift 集群上设置 Prometheus 堆栈。

流程

1. 创建包含 Data Grid 凭证的身份验证 secret，以便 Prometheus 能够与您的 Data Grid 集群进行身份验证。

```
apiVersion: v1
stringData:
  username: developer (1)
  password: dIRs5cAAsHIeeRIL (2)
kind: Secret
metadata:
  name: basic-auth
type: Opaque
```

(1) 指定应用程序用户。developer 是默认值。

(2) 指定对应的密码。

2. 将身份验证 secret 添加到 Prometheus 命名空间中。

```
$ oc apply -f basic-auth.yaml
```

3. 创建服务监控器，将 Prometheus 配置为监控您的 Data Grid 集群。

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    k8s-app: prometheus
  name: datagrid-monitoring (1)
  namespace: infinispan-monitoring (2)
spec:
  endpoints:
    - targetPort: 11222 (3)
      path: /metrics (4)
      honorLabels: true
      basicAuth:
        username:
          key: username
          name: basic-auth (5)
        password:
          key: password
          name: basic-auth
      interval: 30s
      scrapeTimeout: 10s
      scheme: https (6)
      tlsConfig:
        insecureSkipVerify: true
        serverName: example-infinispan (7)
  namespaceSelector:
    matchNames:
      - infinispan (8)
  selector:
    matchLabels:
      app: infinispan-service
      clusterName: example-infinispan (9)
```

(1) 将服务命名为 **monitor** 实例。

(2) 指定 **Prometheus** 堆栈的命名空间。

(3) 为 **Data Grid** 指标端点设置 **11222** 端口。

(4) 设置 **Data Grid** 公开指标的路径。

(5)

使用 Data Grid 凭证指定身份验证 secret。

6

指定 Data Grid 集群使用端点加密。

7

为 Data Grid 加密指定 TLS 证书的通用名称(CN)。如果使用 OpenShift 服务证书，CN 与 Data Grid 集群的 metadata.name 资源匹配。

8

指定 Data Grid 集群的命名空间。

9

指定 Data Grid 集群名称。

4. 将服务监控实例添加到 Prometheus 命名空间中。

```
$ oc apply -f service-monitor.yaml
```

其他资源

- Prometheus Operator
- 使用 OpenShift Cluster Monitoring Stack 监控服务

第 12 章 使用 ANTI-AFFINITY 保证可用性

Kubernetes 包含反关联性功能，可防止工作负载出现单一故障点。

12.1. 反关联性策略

集群中的每个 Data Grid 节点都在集群中运行的 pod 中运行。每个 Red Hat OpenShift 节点在物理主机系统上运行。反关联性的工作原理是将 Data Grid 节点分布到 OpenShift 节点，确保即使出现硬件故障，您的数据网格集群也会保持可用。

Data Grid Operator 提供了两个反关联性策略：

kubernetes.io/hostname

数据源副本 pod 调度到不同的 OpenShift 节点上。

topology.kubernetes.io/zone

数据网格副本 pod 调度到多个区域。

容错

反关联性策略通过不同的方式保证集群可用性。

注意

只有在 OpenShift 节点或区域的数量大于 Data Grid 节点的数量时才适用。

将 pod 调度到不同的 OpenShift 节点上

为以下类型的缓存提供 x 节点故障：

- replicated: x = spec.replicas - 1
- distributed: x = num_owners - 1

在多个区间调度 pod

当 x 区域存在以下类型的缓存时，提供 x 区故障的容错：

- replicated: x = spec.replicas - 1
- distributed: x = num_owners - 1

注意

spec.replicas

定义每个 Data Grid 集群中的 pod 数量。

num_owners

是 cache 配置属性，用于定义缓存中每个条目的副本数。

12.2. 配置 ANTI-AFFINITY

指定 OpenShift 为 Data Grid 集群调度 pod 的位置，以确保可用性。

流程

1. 将 spec.affinity 块添加到 Infinispan CR。
2. 根据需要配置反关联性策略。
3. 应用 Infinispan CR。

其他资源

- 反关联性策略配置

12.3. 反关联性策略配置

在 Infinispan CR 中配置反关联性策略，以控制 OpenShift 调度 Data Grid 副本 pod 的位置。

将 pod 调度到不同的 OpenShift 节点上

如果没有在 Infinispan CR 中配置 spec.affinity 字段，则 Data Grid Operator 会使用反关联性策略：

```
spec:
  affinity:
    podAntiAffinity:
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 100 1
        podAffinityTerm:
          labelSelector:
            matchLabels:
              app: infinispan-pod
              clusterName: <cluster_name>
              infinispan_cr: <cluster_name>
          topologyKey: "kubernetes.io/hostname" 2
```

1

将 hostname 策略设置为首选。

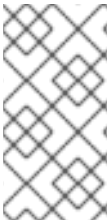
2

将 Data Grid 副本 pod 调度到不同的 OpenShift 节点上。

需要不同的节点

```
spec:
  affinity:
    podAntiAffinity:
      requiredDuringSchedulingIgnoredDuringExecution: 1
      - labelSelector:
          matchLabels:
            app: infinispan-pod
            clusterName: <cluster_name>
            infinispan_cr: <cluster_name>
        topologyKey: "topology.kubernetes.io/hostname"
```

1



注意

为确保您可以在不同的 OpenShift 节点上调度 Data Grid 副本容器集，可用的 OpenShift 节点数量必须大于 spec.replicas 的值。

在多个 OpenShift 区域间调度 pod

以下示例在调度 pod 时首选多个区：

```
spec:
  affinity:
    podAntiAffinity:
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 100 (1)
        podAffinityTerm:
          labelSelector:
            matchLabels:
              app: infinispan-pod
              clusterName: <cluster_name>
              infinispan_cr: <cluster_name>
          topologyKey: "topology.kubernetes.io/zone" (2)
      - weight: 90 (3)
        podAffinityTerm:
          labelSelector:
            matchLabels:
              app: infinispan-pod
              clusterName: <cluster_name>
              infinispan_cr: <cluster_name>
          topologyKey: "kubernetes.io/hostname" (4)
```

(1) 将 zone 策略设置为首选。

(2) 在多个区域间调度 Data Grid 副本 pod。

(3) 将 hostname 策略设置为下一个首选。

(4)

需要多个区域

```
spec:
  affinity:
    podAntiAffinity:
      requiredDuringSchedulingIgnoredDuringExecution: 1
      - labelSelector:
          matchLabels:
            app: infinispan-pod
            clusterName: <cluster_name>
            infinispan_cr: <cluster_name>
        topologyKey: "topology.kubernetes.io/zone"
```

1

仅在调度 Data Grid 副本 pod 时使用 zone 策略。

第 13 章 监控数据网格日志

将日志记录类别设置为不同的消息级别，以监控、调试并对 Data Grid 集群进行故障排除。

13.1. 配置 DATA GRID LOGGING

流程

1. 在 Infinispan CR 中使用 spec.logging 指定日志记录配置，然后应用更改。

```
spec:
  ...
  logging: 1
    categories: 2
      org.infinispan: debug 3
      org.jgroups: debug
```

1. 配置数据网格日志记录。

2. 添加日志记录类别。

3. 命名日志记录类别和级别。

注意

root 日志记录类别是 org.infinispan，默认为 INFO。

2. 根据需要，从 Data Grid 节点检索日志。

```
$ oc logs -f $POD_NAME
```

13.2. 日志级别

日志级别表示消息的性质和严重性。

日志级别	描述
trace	提供有关运行应用程序状态的详细信息。这是最详细的日志级别。
debug	表示单个请求或活动的进度。
info	指明应用程序的整体进度，包括生命周期事件。
warn	指明可能导致错误或降低性能的情况。
错误	指明可能会阻止操作或活动成功但不会阻止应用程序运行的错误条件。

第 14 章 参考

查找有关使用 Data Grid Operator 创建的 Data Grid 服务和集群的信息。

14.1. 网络服务

内部服务

- 允许 Data Grid 节点相互发现并组成集群。
- 提供对同一 OpenShift 命名空间中的客户端对 Data Grid 端点的访问。

Service	端口	协议	描述
<cluster_name>	**11222**	TCP	对 Data Grid 端点的内部访问
<cluster_name>-ping	**8888**	TCP	集群发现

外部服务

提供对 OpenShift 外部或不同命名空间中的客户端对 Data Grid 端点的访问。

注意

您必须使用 Data Grid Operator 创建外部服务。它默认不可用。

Service	端口	协议	描述
<cluster_name>-external	**11222**	TCP	外部访问 Data Grid 端点。

跨站点服务

允许 Data Grid 在不同位置的集群之间备份数据。

Service	端口	协议	描述
<cluster_name>-site	**7900**	TCP	JGroups RELAY2 频道进行跨站点通信。

其他资源

- 创建网络服务