Red Hat build of Cryostat 3

Cryostat 入门

Last Updated: 2024-07-03

法律通告

摘要

Red Hat build of Cryostat 是 OpenShift Container Platform 上的红帽产品。使用 Cryostat 入门指南提供了此产品的概述，并解释了如何安装软件并开始使用它。

目录

前言

Red Hat build of Cryostat 是 JDK Flight Recorder (JFR)的容器原生虚拟化，可用于安全监控在 OpenShift Container Platform 集群上运行的工作负载的 Java 虚拟机(JVM)性能。您可以使用 Cryostat 3.0 使用 Web 控制台或 HTTP API 启动、停止、检索、存档、导入和导出容器化应用中的 JVM 的 JFR 数据。

根据您的用例，您可以使用 Cryostat 提供的内置工具直接存储和分析 Red Hat OpenShift 集群上的记录，或者您可以将记录导出到外部监控应用程序，以对记录数据进行更深入分析。

重要

Red Hat build of Cryostat 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。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。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中有问题的语言。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。详情请查看 CTO Chris Wright 的信息。

第 1 章 CRYOSTAT 概述

Cryostat 是一个基于 JDK Flight Recorder (JFR)的容器原生 Java 应用程序，可用于监控在 Red Hat OpenShift 集群上运行的容器化工作负载的 Java 虚拟机(JVM)性能。

您可以在托管容器化 Java 应用程序的 Red Hat OpenShift 项目中在容器中部署 Cryostat。您可以创建与用于运行容器化工作负载的 JVM 实例对应的 JVM 目标。您可以将 Cryostat 连接到 JVM 目标，以记录和分析有关堆和非堆内存使用情况、线程数、垃圾回收和其他性能指标的数据。

您可以使用 Cryostat 中包含的工具实时监控 JVM 的性能，捕获 JDK Flight Recorder (JFR)记录和快照、生成自动分析报告，并使用 Grafana 仪表板视觉化您记录的性能数据。

Cryostat Web 控制台和 HTTP API 提供了分析容器内的 JVM 性能数据的方法，而无需依赖外部监控应用程序。但是，当需要对集群环境外的数据进行分析时，您还可以将 Cryostat 中的记录从 Cryostat 导出到 JDK Mission Control (JMC)的外部实例中。

Cryostat 支持基于角色的访问控制(RBAC)作为 OpenShift Container Platform 的标准功能。

您可以使用 Operator Lifecycle Manager (OLM)在 Red Hat OpenShift 项目中安装 Cryostat。

您还可以从红帽生态系统目录下载最新的 Cryostat 组件镜像。红帽生态系统目录中为 Cryostat 3.0 存在以下容器镜像：

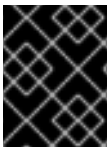
- Cryostat
- Red Hat build of Cryostat Operator
- Red Hat build of Cryostat Operator 捆绑包
- Cryostat 报告
- Cryostat Grafana 仪表板
- Cryostat DB
- Cryostat 存储
- JFR 数据源

其他资源

- Operator Lifecycle Manager (OLM) (OpenShift Container Platform)
- 容器镜像（红帽生态系统目录）

第 2 章 使用 RED HAT BUILD OF CRYOSTAT OPERATOR 在 RED HAT OPENSHIFT 上安装 CRYOSTAT

您可以使用 Operator Lifecycle Manager (OLM)在 Red Hat OpenShift 集群的项目中安装 Red Hat build of Cryostat Operator。您可以使用 Red Hat build of Cryostat Operator 创建单个命名空间或多命名空间 Cryostat 实例。您可以使用可从 Red Hat OpenShift Web 控制台访问的 GUI 来控制这些实例。



重要

如果需要将 Red Hat build of Cryostat Operator 订阅从 Cryostat 2.0 升级到 Cryostat 3.0，您必须将更新频道从 **stable-2.0** 改为 **stable**。

先决条件

- 创建 OpenShift Container Platform 4.12 或更高版本集群。
- 创建具有在项目中安装 Red Hat build of Cryostat Operator 权限的 Red Hat OpenShift 用户帐户。
- 在集群中安装了 Operator Lifecycle Manager (OLM)。
- 使用 Red Hat OpenShift 的 cert-manager Operator 安装了 cert-manager。
  - 如果使用 OpenShift Container Platform 4.12 或更高版本，您可以为 Red Hat OpenShift 安装 cert-manager Operator。如需更多信息，请参阅 Red Hat OpenShift (OpenShift Container Platform)的 cert-manager Operator。
- 使用 Red Hat OpenShift Web 控制台登录到 Red Hat OpenShift。

流程

1. 在您的浏览器中，使用 Web 控制台进入到 **Home** > **Projects**。
2. 选择您要在其中安装 Red Hat build of Cryostat Operator 的项目的名称。
3. 安装 Red Hat build of Cryostat Operator：
   a. 在 Web 控制台的导航菜单中导航到 **Operators** > **OperatorHub**。
   b. 从列表中选择 **Red Hat build of Cryostat Operator**。您可以使用屏幕右上角的搜索框来查找 Red Hat build of Cryostat Operator。
   c. 要在项目中安装 Red Hat build of Cryostat Operator，点 **Install**。
   Red Hat OpenShift Web 控制台会提示您创建 Cryostat 自定义资源(CR)。

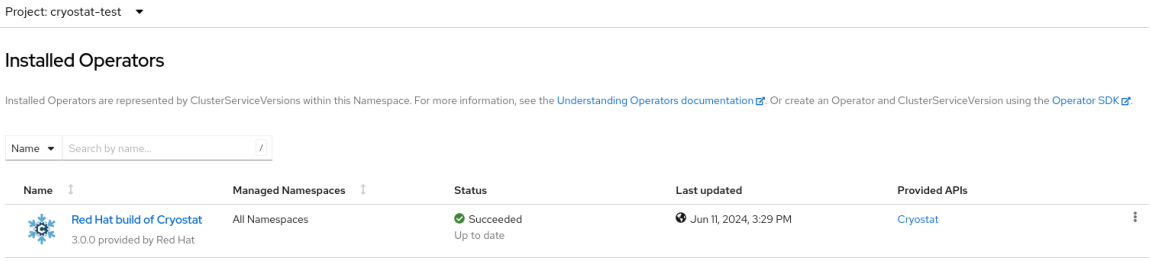
   注意

   从 Cryostat 3.0 开始，在 **安装模式** 区域中，**集群中的所有命名空间（默认）** 单选按钮是唯一可用的选项。

   您可以手动或自动创建 CR。如果要手动创建 CR，请参阅第 4 步。如果要自动创建 CR，请参阅第 5 步。
4. 如果要手动创建 CR，请完成以下步骤：

a. 使用 Web 控制台进入到 **Operators** > **Installed Operators**，然后从安装的 Operator 列表中选择 **Red Hat build of Cryostat Operator**：

图 2.1. 在安装的 Operator 列表中查看 Red Hat build of Cryostat operator



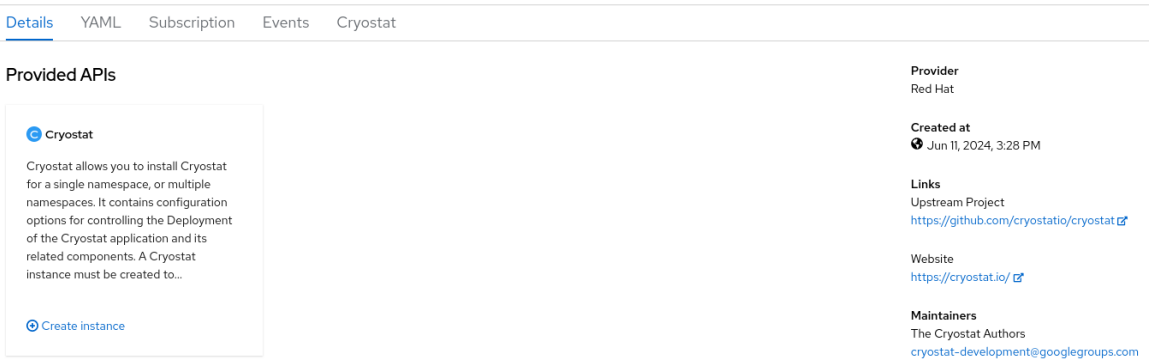
b. 点 **Details** 标签页。

c. 要创建 Cryostat 实例，请转至 **Provided APIs** 部分。然后，在 **Cryostat** 下，单击 **Create instance**。

注意

从 Cryostat 3.0 开始，**Cryostat** API 可让您创建单命名空间和多命名空间 Cryostat 实例。

图 2.2. 选择 Red Hat build of Cryostat Operator 提供的 Cryostat API



d. 点 **Form view** 单选按钮或 **YAML view** 单选按钮。如果要在 YAML 配置文件中输入您的信息，点 **YAML 视图**。

e. 为您要创建的 Cryostat 实例指定唯一名称。

f. *可选*：在 Labels 字段中，为您要部署的 Operand 工作负载指定标签或注解。

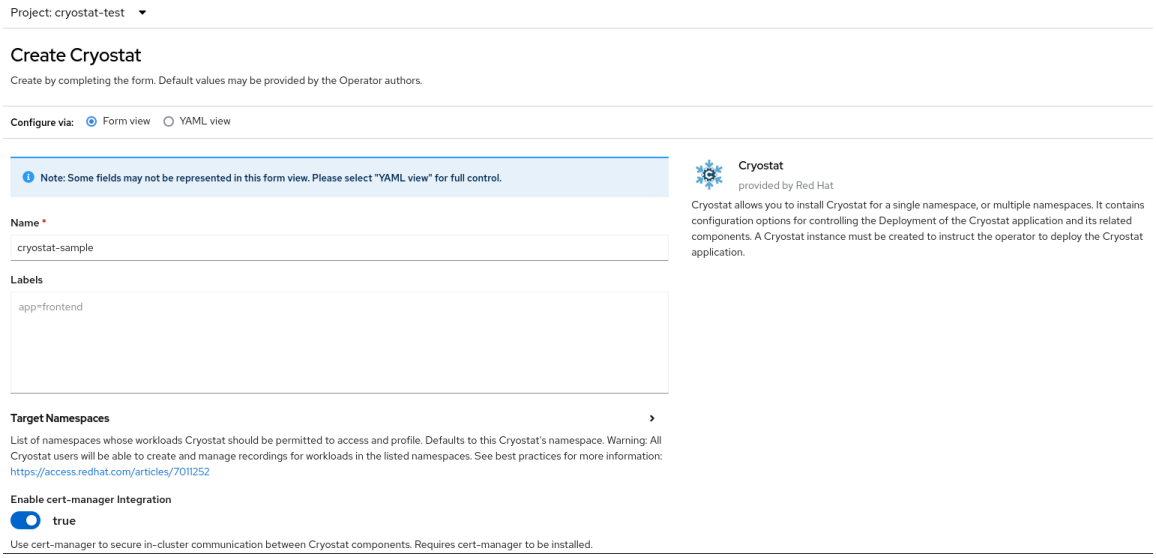
g. 在 **Target Namespaces** 字段中，选择您要允许此 Cryostat 实例访问并使用的命名空间。另外，您可以选择安装 Cryostat 的同一命名空间，也可以选择不同的命名空间。要添加额外的命名空间，请点击 **+Add Target Namespace**。

重要

可以访问 Cryostat 实例的用户可以访问该 Cryostat 实例可见的任何命名空间中的所有目标应用程序。因此，当部署多命名空间 Cryostat 实例时，您必须考虑要选择哪些命名空间进行监控，哪些命名空间要安装 Cryostat，以及您要授予哪些用户访问权限。

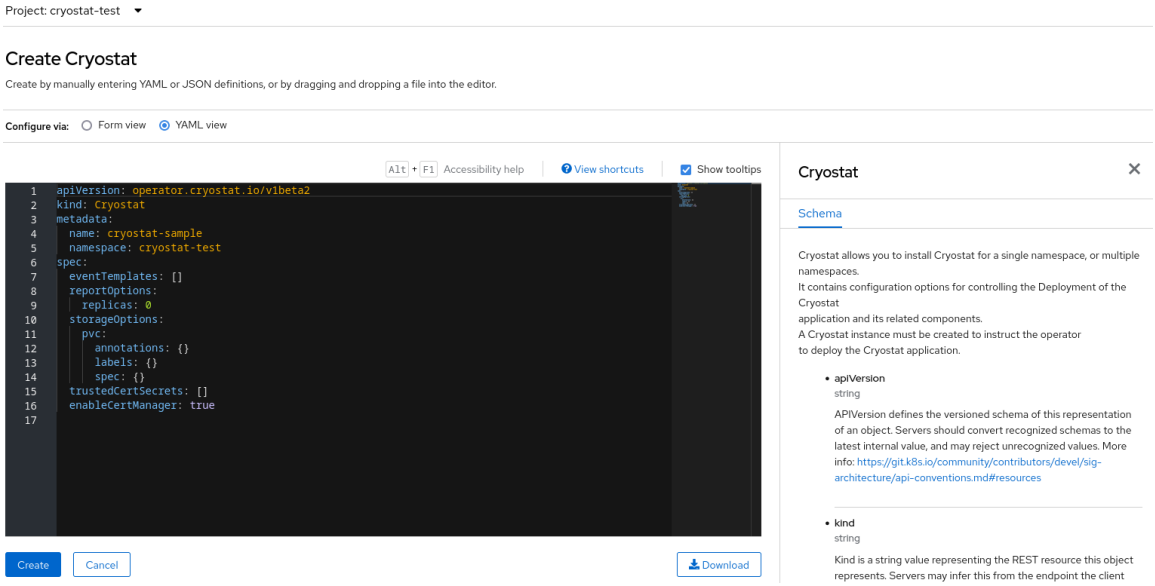
您还可以为您的部署指定额外的配置选项：

图 2.3. 使用 web 控制台中的表单创建 Cryostat 实例



另外，您可以使用 YAML 模板来创建实例，并指定其他配置选项，而不使用以下格式：

图 2.4. 使用 web 控制台中的 YAML 模板创建 Cryostat 实例



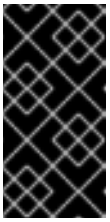
5. 如果要使用自动提示选项创建 CR，请按照提示的说明操作，然后完成以下步骤：

   a. 点 **Form view** 单选按钮或 **YAML view** 单选按钮。如果要在 YAML 配置文件中输入您的信息，点 **YAML 视图**。

   b. 为您要创建的 Cryostat 实例指定唯一名称。

   c. *可选*：在 Labels 字段中，为您要部署的 Operand 工作负载指定标签或注解。

   d. 在 **Target Namespaces** 字段中，选择您要允许此 Cryostat 实例访问并使用的命名空间。另外，您可以选择安装 Cryostat 的同一命名空间，也可以选择不同的命名空间。要添加额外的命名空间，请点击 **+Add Target Namespace**。

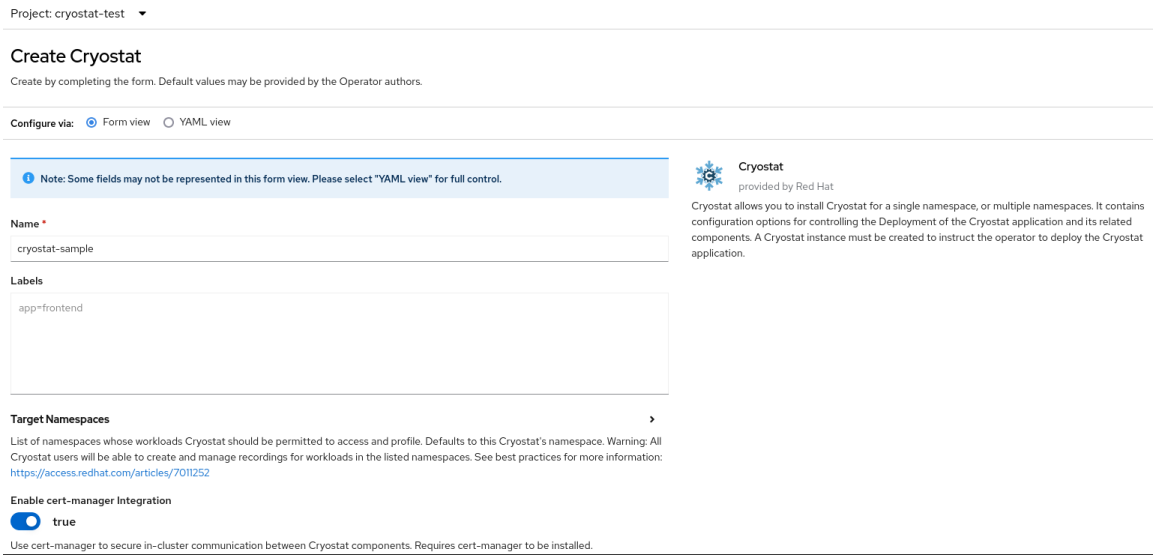


重要

可以访问 Cryostat 实例的用户可以访问该 Cryostat 实例可见的任何命名空间中的所有目标应用程序。因此，当部署多命名空间 Cryostat 实例时，您必须考虑要选择哪些命名空间进行监控，哪些命名空间要安装 Cryostat，以及您要授予哪些用户访问权限。

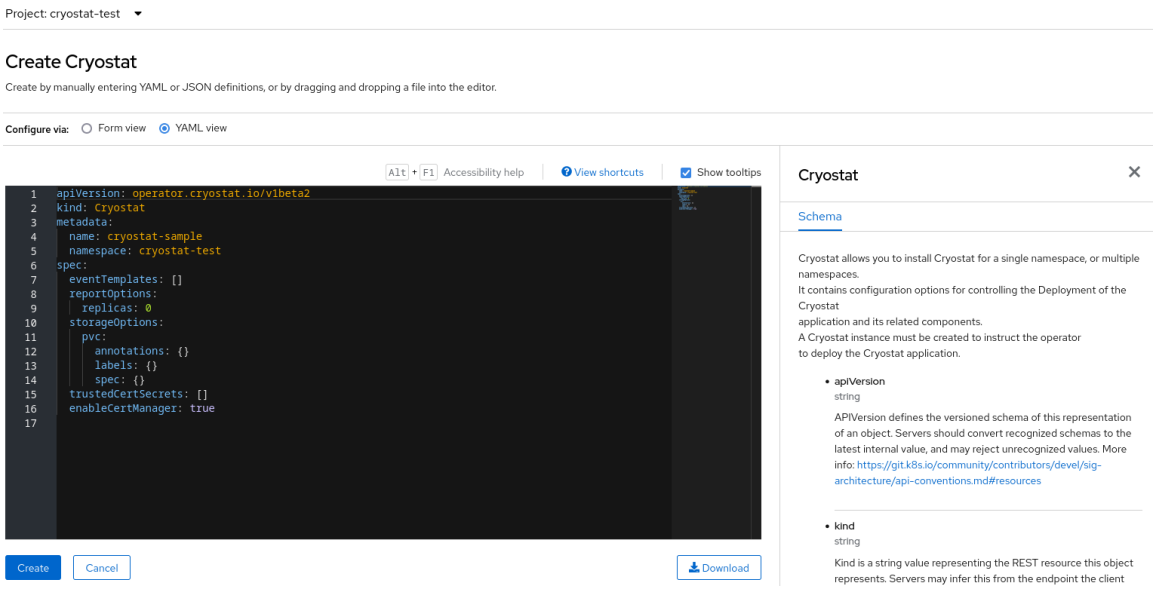
您还可以为您的部署指定额外的配置选项：

图 2.5. 使用 web 控制台中的表单创建 Cryostat 实例



另外，您可以使用 YAML 模板来创建实例，并指定其他配置选项，而不使用以下格式：

图 2.6. 使用 web 控制台中的 YAML 模板创建 Cryostat 实例



6. 要为 Cryostat 实例启动创建过程，请点击 **Create**。
   您必须等待 Cryostat 实例的所有资源就绪，然后才能访问它。

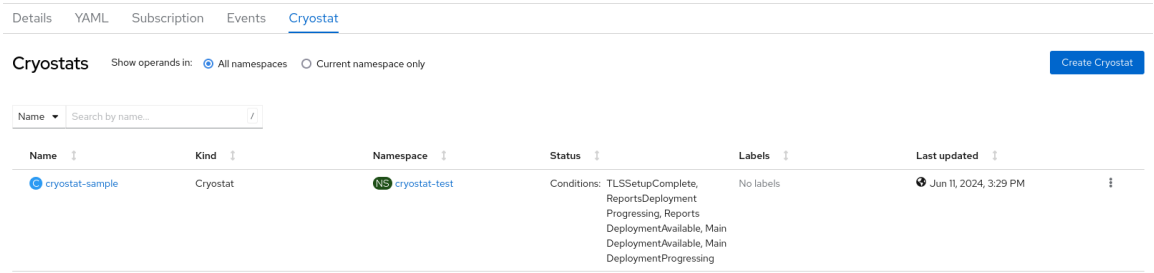
验证

1. 在 Web 控制台的导航菜单中，点 **Operators**，然后点 **Installed Operators**。
2. 在安装的 operator 表中，选择 **Red Hat build of Cryostat Operator**。

3. 选择 **Cryostat** 选项卡。
   您的 Cryostat 实例在实例表中打开，并列出以下条件：

   - **TLSSetupComplete** 设置为 **true**。
   - **MainDeploymentAvailable** 设置为 **true**。
   - 可选：如果您启用了报告生成器服务，则会显示 **ReportsDeploymentAvailable**，并将它设为 **true**。

   图 2.7. OpenShift 上 Cryostat 实例的 Status 列下设置为 True 的条件示例



4. *可选：* 从 Cryostat 表中选择您的 **Cryostat** 实例。进入 **Cryostat Conditions** 表，您可以在其中查看每个条件的更多信息。

   图 2.8. 列出每个条件及其条件的 Cryostat Conditions 表示例

Cryostat Conditions

Type	Status	Updated	Reason	Message
TLSSetupComplete	True	Jun 11, 2024, 3:29 PM	AllCertificatesReady	All certificates for Cryostat components are ready.
ReportsDeploymentProgressing	True	Jun 11, 2024, 3:29 PM	NewReplicaSetAvailable	ReplicaSet "cryostat-sample-reports-68b9fbf54c" has successfully progressed.
ReportsDeploymentAvailable	True	Jun 11, 2024, 3:30 PM	MinimumReplicasAvailable	Deployment has minimum availability.
MainDeploymentAvailable	True	Jun 11, 2024, 3:31 PM	MinimumReplicasAvailable	Deployment has minimum availability.
MainDeploymentProgressing	True	Jun 11, 2024, 3:29 PM	NewReplicaSetAvailable	ReplicaSet "cryostat-sample-5bdc9655ff" has successfully progressed.

其他资源

- 在不同集群配置中设置 Cryostat 的最佳实践 （红帽知识库）
- 使用 Web 控制台访问 Cryostat

第 3 章 配置 JAVA 应用程序

要启用 Cryostat 收集、存储和分析 Java Flight Recorder (JFR)数据，有关 Java 虚拟机(JVM)上运行的目标应用程序的数据，您必须配置应用程序，以便 Cryostat 可以检测并连接到它们。

您可以使用以下方法配置应用程序：

- 通过使用 Cryostat 代理组件进行检测和连接，该代理作为 Java Instrumentation Agent 实施，并充当 JVM 上运行的应用程序的插件
- 通过将应用程序配置为允许 Java 管理扩展(JMX)连接，并使用 OpenShift Service 检测和 JMX 进行连接
- 通过使用 Cryostat 代理检测和 JMX 进行连接

Cryostat 代理

Cryostat 代理提供了一个 HTTP API，Cryostat 服务器可用作应用程序 JMX 端口的替代选择。通过将正确配置的 Cryostat 代理附加到您部署的工作负载应用程序中，您可以使用完整的 Cryostat 功能集，而无需目标应用程序公开 JMX 端口。

注意

在 Red Hat build of Cryostat 2.4 之前，Cryostat 代理提供了一个只读 HTTP API，它只支持有限的 JFR 操作。

与 JMX 端口相比，Cryostat 代理的 HTTP API 可以提供以下优点：

- 由于 API 面区域减少，安全性更高
- 由于 Cryostat 代理的双角色作为 Cryostat 发现插件，部署灵活性

如果 Cryostat 代理检测到也会在应用程序上配置 JMX，代理会将自身发布到带有代理 HTTP API 定义和 JMX URL 定义的 Cryostat 服务器。在这种情况下，您可以使用您喜欢的任何配置选项。

JVM 的动态附加

从 Cryostat 3.0 开始，Cryostat 代理可以动态地附加正在运行的应用程序 JVM，而无需应用程序重启。这个动态附加功能有以下要求：

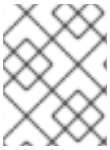
- 您必须确保代理的 JAR 文件被复制到 JVM 的文件系统中（例如，使用 **oc cp** 命令）。
- 您必须能够在同一主机上或同一应用程序（例如，使用 **oc exec** 命令）将代理作为单独的进程运行。

动态附加功能支持临时一次性性能分析或故障排除工作流，您可能不需要在每次 JVM 启动时附加代理。动态附加还适合于无法或不想为附加代理的唯一目的重新配置应用程序的情况。由于代理可以在不需要应用重启的情况下附加到正在运行的 JVM，因此这也意味着没有应用停机时间。

JVM 的静态附件

您可以启用工作负载应用的 JVM，在 JVM 启动时加载和初始化 Cryostat 代理。这个静态附加方法要求您将应用程序配置为将路径为 Cryostat 代理的 JAR 文件（如，**-javaagent:/deployment/app/lib/cryostat-agent.jar**）传递-javaagent JVM 标志。在 Cryostat 代理的基本初始化完成后，您的工作负载应用程序的正常启动过程会正常开始。

根据您的工作负载应用程序，静态附加方法可能需要设置一个或多个环境变量，或者在 **argLine** 参数中添加参数。然而，在某些情况下，您可能需要重新配置、重建并重新部署应用程序。静态附加还需要重启应用程序 JVM，这可能会导致应用程序停机。



注意

在 Cryostat 3.0 之前，通过 **-javaagent** JVM 标志进行静态附加是启用 Cryostat 代理在目标应用程序上附加和运行的唯一方法。

将代理的 JAR 文件包含到工作负载应用程序中的选项

根据您的要求，您可以以不同的方式将 Cryostat 代理的 JAR 文件包含在工作负载应用程序中：

- 对于正在运行的 JVM 的动态一次性连接，您可以使用 **oc cp** 命令将代理的 JAR 文件复制到 JVM 的文件系统中。
- 对于在启动时对 JVM 的静态附件，您可以使用以下选项之一：
  - 最简单的选项是将 JAR 文件添加到 **pom.xml** 或 **build.gradle** 文件中的应用依赖项中。您的构建工具(Maven 或 Gradle)会下载 JAR 文件，以包含在您的应用构建输出中。
  - 您可以使用 Maven 插件（如 **maven-dependency-plugin** ）提供对应用构建输出中下载和包含 JAR 文件的更精细的控制。
  - 您可以创建一个包含 JAR 文件的 PersistentVolume 存储卷。然后，重新配置应用的 **Deployment/DeploymentConfig** 以挂载 PersistentVolume 和 use **-javaagent:/path/to/persistentvolume/cryostat-agent.jar**。完成此任务的确切方法取决于您在 OpenShift 集群中启用的 PersistentVolume 供应商类型。

当 Cryostat 代理成功添加到应用程序容器中并载入后，应用程序的 **stdout** 和 **控制台日志** 会开始显示 Cryostat 代理中的日志消息。

代理配置属性

您可以通过以下两种方式之一为 Cryostat 代理指定配置属性：

- 在应用上使用 JVM 系统属性标志（例如，**-Dcryostat.agent.api.writes-enabled=true**）。
- 通过将所有字母大写字母组成并将任何标点替换为下划线（例如，**CRYOSTAT_AGENT_API_WRITES_ENABLED=true**）来使用环境变量。

您必须配置以下属性，以便 Cryostat 代理能够成功运行：

cryostat.agent.baseuri	这将指定 Cryostat 代理向（即内部 OpenShift Service 对象）公告自己（如 **https://my-cryostat.my-namespace.svc.cluster.local** ）的 Cryostat 服务器后端的 URL 位置。
cryostat.agent.callback	这将指定 Cryostat 代理实例或应用程序本身的 URL 位置。Cryostat 使用此 URL 执行健康检查并从代理请求数据。您可以使用 OpenShift/Kubernetes Downward API 动态决定这一点。如需更多信息，请参阅有关 **status.podIP** 的 Kubernetes Downward API 文档。

根据您的设置要求，您还可以配置以下代理属性：

cryostat.agent.api.writes-enabled	这表明 Cryostat 代理是否允许写操作。这默认设置为 **false**。如果您希望 Cryostat 代理接受启动、停止或删除 JFR flight 记录的请求，您必须将此属性设置为 **true**。 **注意** 即使此属性设置为 **false**，代理仍然可以履行请求，以列出动态记录或下载单个记录文件。
cryostat.agent.webserver.port	这将指定代理用来绑定其 HTTP API 的 HTTP 端口号（默认为 9977）。如果这与应用程序或其他工具代理使用的现有端口冲突，您必须指定不同的端口号。
cryostat.agent.app.name	这指定了一个标签，用于标识此 Cryostat 代理实例附加到哪个应用程序（默认情况下，cry **ostat-agent**）。您可以使用 Downward API **metadata.name** 或 **metadata.labels['app']** 字段。如需更多信息，请参阅 Kubernetes Downward API 文档Kubernetes Downward API 文档。

远程 Java 管理扩展(JMX)连接

JMX 是 JVM 上的标准功能，您可以使用它监控和管理 JVM 上运行的目标应用程序。要使 Cryostat 使用 JMX，您必须在启动 JVM 时启用和配置 JMX，因为 Cryostat 需要目标应用程序来公开 JMX 端口。

Cryostat 通过此 JMX 端口与目标应用程序通信，以启动和停止 JFR 记录并通过网络拉取 JFR 数据，启用 Cryostat 存储和分析这个 JFR 数据。远程监控需要安全性，以确保未授权的人员无法访问应用程序。Cryostat 会提示您输入您的凭证，然后 Cryostat 可以访问任何应用程序的 JFR 记录。

Cryostat 代理和 JMX 混合

您可以将目标应用程序配置为使用混合方法，其中使用 Cryostat 代理和 JMX。使用此方法，您可以使用 Cryostat 代理来检测目标应用程序和 JMX 来公开 JFR 数据到 Cryostat，从而获得更大的灵活性。

例如，您可以使用代理来检测应用程序，而无需依赖特定的端口号，并使用 JMX 连接根据需要启动和停止 JFR flight 记录。

3.1. 启动 CRYOSTAT 代理作为到 JVM 动态附加的独立进程

如果您希望 Cryostat 代理动态附加到已在运行的应用程序 JVM，您可以将代理作为独立 Java 进程启动。

注意

只有在您想要使用动态附加功能时，这个过程才相关，这允许 Cryostat 代理附加到临时一次性运行的 JVM。如果您希望工作负载的 JVM 在 JVM 启动时加载和初始化 Cryostat 代理，请参阅使用 Cryostat 代理配置应用程序。

先决条件

- 使用 **oc cp** 命令将代理的 JAR 文件复制到 JVM 的文件系统。

流程

- 输入以下命令：

```
$ java -jar target/<agent_jar_file> <pid>
```

在前面的命令中，将 <*agent_jar_file* > 替换为代理的 JAR 文件名，并将 *<pid* > 替换为您要附加到的 JVM 的进程 ID (PID)。

例如：

```
$ java -jar target/cryostat-agent-0.4.0.jar 1234
```

运行上述命令时，代理进程使用其附加提供程序来查找指定的 PID。如果找到指定的 PID，代理进程会附加到此 PID，并尝试将代理的 JAR 文件加载到此 JVM 中，然后引导到正常的代理启动进程。

基于 PID 值的代理启动行为

根据 PID 值，请考虑以下代理启动行为准则：

- 如果您指定了无效的 PID，代理将无法成功启动。
- 如果您将通配符星号(*)指定为 PID 值，代理的 JAR 文件将尝试引导到它找到的每个 JVM。
- 如果将 **0** 指定为 PID 值，或者没有指定任何 PID 值，代理会检查是否只有一个 candidate JVM 可用。如果只有一个 JVM 可用，代理会尝试引导到此 JVM。如果多个 JVM 或没有 JVM 可用，代理将无法成功启动。

late-binding 配置选项

当将 Cryostat 代理作为独立进程启动时，您也可以使用命令行选项在代理启动程序中指定额外的 late-binding 配置选项。

例如：

```
$ java -jar target/cryostat-agent-0.4.0.jar \
-Dcryostat.agent.baseuri=http://cryostat.local \
--smartTrigger=[ProcessCpuLoad>0.2]~profile \
@/deployment/app/moreAgentArgs \
1234
```

有关可用选项及其行为的更多信息，请运行 **java -jar target/cryostat-agent-0.4.0.jar -h** help 命令。在注入的代理尝试读取配置值之前，您在主机 JVM 上设置了通过 **-D** 指定的系统属性。这与在主机 JVM 进程本身上设置这些系统属性或对等环境变量的影响相同。

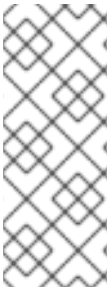
3.2. 使用 CRYOSTAT 代理配置应用程序

您可以使用 Cryostat 代理（作为 Java Instrumentation Agent 实施）来配置目标应用程序，以便 Cryostat 可以检测应用程序，收集数据，并将数据发送到 Cryostat 以进行分析。您还可以选择启用 Cryostat 代理，以接受来自 Cryostat 服务器的请求，以启动、停止和删除 JFR 记录。

Red Hat build of Cryostat 3.0 分发了 Cryostat 代理的 JAR 文件的两个不同变体。根据您的设置要求，您可以使用以下一种代理 JAR 文件：

- 一个一体化的"shaded" JAR 文件，该文件是自包含的，包括代理代码及其所有依赖项
  此"shaded" JAR 文件提供了在现有应用程序中包括的最方便的 Cryostat 代理形式，因为您只需要包含一个额外的代理 JAR 文件。这是类似代理和工具的常见分发模式。
- 包含没有依赖项的代理代码的标准 JAR 文件

如果您知道代理和工作负载应用程序之间存在依赖关系冲突，则这种类型的 JAR 文件很有用。如果您打算应用自己的策略，以提供每个依赖项的正确版本以满足代理和应用程序的要求，您可以使用独立 JAR 文件。



注意

之前的版本提供了 Cryostat 代理的一个发行版，它是一个"shaded""JAR 文件。以下流程描述了如何安装 Cryostat 3.0 代理的"shaded" JAR 文件分发。

如 配置 Java 应用程序：Cryostat 代理 中所述，Cryostat 3.0 代理支持不同的选项，以将代理的 JAR 文件包含在工作负载应用程序中。以下流程描述了如何将 "shaded" JAR 文件添加到 **pom.xml** 或 **build.gradle** 文件中的应用程序依赖项中。

先决条件

- 登录到您的 Cryostat web 控制台。
- 安装了 JDK 版本 11 或更高版本。

流程

1. 安装 Cryostat 代理。根据您的应用程序构建选择以下选项之一：
   - 使用 Maven：
     使用 Cryostat 代理 JAR 文件信息更新应用程序 **pom.xml** 文件。

     pom.xml 示例

```
<project>
  ...
  <repositories>
    <repository>
      <id>redhat-maven-repository</id>
      <url>https://maven.repository.redhat.com/earlyaccess/all/</url>
    </repository>
  </repositories>
  ...
  <build>
    <plugins>
      <plugin>
        <artifactId>maven-dependency-plugin</artifactId>
        <version>3.3.0</version>
        <executions>
          <execution>
            <phase>prepare-package</phase>
            <goals>
              <goal>copy</goal>
            </goals>
            <configuration>
              <artifactItems>
                <artifactItem>
                  <groupId>io.cryostat</groupId>
                  <artifactId>cryostat-agent</artifactId>
                  <version>0.4.0.redhat-xxxxx</version>
                  <classifier>shaded</classifier>
                </artifactItem>
```

```
            </artifactItems>
            <stripVersion>true</stripVersion>
          </configuration>
        </execution>
      </executions>
    </plugin>
  </plugins>
  ...
 </build>
 ...
</project>
```

注意

在前面的示例中，将 **0.4.0.redhat-*xxxxx*** 替换为 Cryostat 代理的最新构建版本（例如 **0.4.0.redhat-00001**）。有关 Cryostat 代理的最新构建版本的详情，请参考 Red Hat Maven 存储库。

下次构建应用程序时，Cryostat 代理 JAR 文件位于 **target/dependency/cryostat-agent-shaded.jar**。

- 使用 Gradle：
  更新 **build.gradle** 文件。

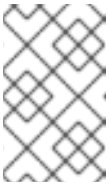
  build.gradle 文件示例

```
repositories {
 …
maven {
    url "https://maven.repository.redhat.com/earlyaccess/all/"
    credentials {
       username "myusername"
        password "mytoken"
    }
 }
}
```

  如何将代理 JAR 文件打包到应用中取决于用于构建的 Gradle 插件。例如，如果您使用 Jib 插件，请按如下所示更新 **build.gradle** 文件：

  build.gradle 文件示例

```
plugins {
 id 'java'
 id 'application'
 id 'com.google.cloud.tools.jib' version '3.3.1'
 id 'com.ryandens.javaagent-jib' version '0.5.0'
}
…
dependencies {
 …
 javaagent 'io.cryostat:cryostat-agent:0.4.0.redhat-xxxxx:shaded'
```



注意

在前面的示例中，将 **0.4.0.redhat-*xxxxx*** 替换为 Cryostat 代理的最新构建版本（例如 **0.4.0.redhat-00001**）。有关 Cryostat 代理的最新构建版本的详情，请参考 Red Hat Maven 存储库。

2. 更新 Docker 文件。以下示例使用 **JAVA_OPTS** 环境变量传递相关的 JVM 信息。

Example

```
...
COPY target/dependency/cryostat-agent.jar /deployments/app/
...
ENV JAVA_OPTS="-javaagent:/deployments/app/cryostat-agent-shaded.jar"
```

3. 重建特定于应用程序的容器镜像。

```
docker build -t docker.io/myorg/myapp:latest -f src/main/docker/Dockerfile
```

4. 要提供您需要配置 Cryostat 代理的 JVM 系统属性或环境变量，请推送更新的镜像，然后修改应用程序部署。

Example

```
apiVersion: apps/v1
kind: Deployment
...
spec:
  ...
  template:
    ...
    spec:
      containers:
        - name: sample-app
          image: docker.io/myorg/myapp:latest
          env:
            - name: CRYOSTAT_AGENT_APP_NAME
              value: "myapp"
              # Replace this with the Kubernetes DNS record
              # for the Cryostat Service
            - name: CRYOSTAT_AGENT_BASEURI
              value: "http://cryostat.mynamespace.mycluster.svc:4180"
            - name: POD_IP
              valueFrom:
                fieldRef:
                  fieldPath: status.podIP
            - name: CRYOSTAT_AGENT_CALLBACK
              value: "http://$(POD_IP):9977" (1)
              # Replace "abcd1234" with a plain-text authentication token
            - name: CRYOSTAT_AGENT_AUTHORIZATION (2)
              value: "Bearer abcd1234"
            - name: CRYOSTAT_AGENT_API_WRITES_ENABLED (3)
              value: true
          ports:
```

```
        - containerPort: 9977
          protocol: TCP
      resources: {}
    restartPolicy: Always
status: {}
```

- <1>: 端口号 **9977** 是代理为服务 Cryostat 请求的内部 Web 服务器公开的默认 HTTP 端口。如果与安装代理的目标应用程序冲突，您可以更改此端口号。
- <2>： **CRYOSTAT_AGENT_AUTHORIZATION** 值显示代理在 API 请求中包含的凭证，以公告其自身存在或推送 JFR 数据。您还可以为此目的创建一个 Kubernetes **服务帐户**，并将 **abcd1234** 替换为与服务帐户关联的纯文本身份验证令牌。
- <3>：默认情况下 **CRYOSTAT_AGENT_API_WRITES_ENABLED** 变量设为 **false**。如果您希望 Cryostat 代理接受来自 Cryostat 服务器的请求以启动、停止或删除 JFR flight 记录，您必须将此变量设置为 **true**。

3.2.1. 将 Cryostat 代理配置为信任 Cryostat 服务器

当您将 Cryostat 代理与启用了 cert-manager 集成的 Cryostat 实例搭配使用时，Cryostat 代理会通过安全 HTTPS 连接与 Cryostat 通信。在这种情况下，Cryostat Operator 使用 cert-manager 来生成自签名证书颁发机构(CA)证书，该证书存储在 secret 中。您必须将 Cryostat 代理配置为信任此 CA 证书。

流程

1. 创建一个 Cryostat CR，为您的 Cryostat 实例和目标应用程序定义命名空间。
   例如，如果要在 **cryostat** 命名空间中创建 Cryostat CR，该 CR 配置为连接到 **apps** 命名空间中的目标应用程序，请输入以下详情：

```
apiVersion: operator.cryostat.io/v1beta2
kind: Cryostat
metadata:
  name: cryostat-sample
  namespace: cryostat
spec:
  enableCertManager: true
  targetNamespaces:
  - apps
```

2. 在 Cryostat 可用时，要获取目标应用程序命名空间中的 CA 证书 secret，请输入以下命令：

```
$ oc project apps
$ oc get secret "cryostat-ca-$(echo -n 'cryostat/cryostat-sample' | sha256sum | cut -d ' ' -f 1)"
```

   在前面的示例中，将 **apps** 替换为您的目标应用程序的命名空间，并将 **cryostat/cryostat-sample** 替换为 Cryostat 实例的命名空间和名称。另外，请确保 Cryostat 实例的命名空间和名称用正斜杠(/)分隔。

   前面的命令会生成类似如下的输出：

```
NAME                                                                TYPE    DATA   AGE
cryostat-ca-30268177e44252b3f9b7d9bf3a6db48f3a1cd3656700a6830952afc4456c0048   Opaque  1      11s
```

   如上例所示，secret 名称包含一个哈希后缀，以防止与集群中的其他 Cryostat 实例冲突。

3. 通过创建 init 容器将证书导入到信任存储中，将 Cryostat 代理配置为信任 Cryostat 的 CA 证书。

注意

此步骤假设您在 **apps** 命名空间中有一个名为 **my-app** 的目标应用程序，该应用程序安装了 Cryostat 代理，否则会正确配置。

在以下示例中，将任何出现 **my-app** 和 **apps** 替换为目标应用程序的名称和命名空间。

a. 使用在第 2 步中 获取的 secret 名称，为部署中的 Cryostat CA secret 创建卷。
例如：

```
$ oc set volumes deploy/my-app --add --name=cryostat-ca --secret-name=cryostat-ca-30268177e44252b3f9b7d9bf3a6db48f3a1cd3656700a6830952afc4456c0048
```

前面的命令会生成类似如下的输出：

```
deployment.apps/my-app volume updated
```

b. 创建一个 **emptyDir** 卷，以在 init 容器和应用程序容器间共享信任存储：
例如：

```
$ oc set volumes deploy/my-app --add --name=truststore -m /var/run/secrets/io.cryostat/truststore
```

前面的命令会生成类似如下的输出：

```
deployment.apps/my-app volume updated
```

c. 向目标应用程序的部署 YAML 文件中添加一个 init 容器。
例如：

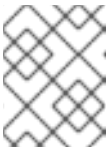
```
initContainers:
  - name: pem-to-truststore
    image: registry.access.redhat.com/ubi8/openjdk-11-runtime:latest
    command:
      - /bin/bash
    args:
      - -c
      - >-
        keytool -import -file
        /var/run/secrets/io.cryostat/cryostat-ca/tls.crt
        -keystore
        /var/run/secrets/io.cryostat/truststore/truststore.jks
        -trustcacerts -noprompt
        -storepass <my password>
    volumeMounts:
      - mountPath: /var/run/secrets/io.cryostat/cryostat-ca
        name: cryostat-ca
      - mountPath: /var/run/secrets/io.cryostat/truststore
        name: truststore
```

在前面的示例中，将 &**lt;my** password> 替换为您要使用的密码。

d. 在应用程序中定义 **javax.ssl.trustStore** Java 系统属性。
例如：

```
env:
  - name: JAVA_OPTS_APPEND
    value: |-
      # …
      -Djavax.net.ssl.trustStore=/var/run/secrets/io.cryostat/truststore/truststore.jks
      -Djavax.net.ssl.trustStorePassword=<my password>
```

在前面的示例中，将 **<my** password> 替换为您在上一步中指定的密码。



注意

如何设置 Java 系统属性取决于应用程序的构建方式以及您使用的基础镜像。前面的示例假设您使用通过 OpenJDK UBI 镜像构建的应用程序。

3.3. 使用 JMX 连接配置应用程序

要使 Cryostat 检测和与目标 Java 应用程序通信，您可以将应用程序配置为允许远程 Java 管理扩展 (JMX)连接。

先决条件

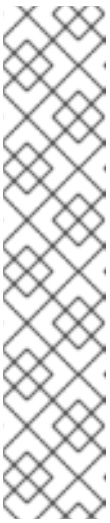
- 登录到您的 Cryostat web 控制台。
- 在项目中创建一个 Cryostat 实例。

流程

1. 要启用远程 JMX 连接，请完成以下步骤：

   a. 在应用程序中，定义以下 Java 系统属性：

```
-Dcom.sun.management.jmxremote.port=<port_num>
```



注意

要添加 **-Dcom.sun.management.jmxremote.port=<port_num** > 属性，而无需重新构建目标应用程序，您可以在应用程序上设置 **JAVA_OPTS_APPEND** 环境变量。**JAVA_OPTS_APPEND** 是一个环境变量，仅用于 Red Hat Universal Base Images (UBI)。

如果您使用 Red Hat UBI 构建应用程序镜像，请在应用程序 Docker 文件中构建时或在运行时设置 **JAVA_OPTS_APPEND** 变量：

```
oc set env deployment <name> JAVA_OPTS_APPEND="..."
```

如果不使用 Red Hat UBI 构建应用程序镜像，请参阅基础镜像的文档，以了解有关如何在构建时或运行时添加 Java 系统属性的信息。

b. 通过允许到应用程序的流量指定应用程序侦听远程 JMX 连接。使用 Red Hat OpenShift Service 并为远程 JMX 端口指定以下值：

service.yaml示例

```
apiVersion: v1
kind: Service
...
spec:
  ports:
    - name: "jfr-jmx"
      port: 9091
      targetPort: 9091
...
```

2. 保护远程 JMX 连接：

a. 为应用程序中的远程 JMX 连接启用并配置身份验证和 SSL/TLS：

```
-Dcom.sun.management.jmxremote.port=<port_num>

 # enable JMX authentication
-Dcom.sun.management.jmxremote.authenticate=true

# define users for JMX auth
-Dcom.sun.management.jmxremote.password.file=</path/to/jmxremote.password>

# set permissions for JMX users
-Dcom.sun.management.jmxremote.access.file=</path/to/jmxremote.access>

 # enable JMX SSL
 -Dcom.sun.management.jmxremote.ssl=true

# enable JMX registry SSL
-Dcom.sun.management.jmxremote.registry.ssl=true

# set your SSL keystore
-Djavax.net.ssl.keyStore=</path/to/keystore>

# set your SSL keystore password
-Djavax.net.ssl.keyStorePassword=<password>
```

b. 配置 Cryostat 以信任应用程序 TLS 证书。在与 Cryostat 应用程序相同的命名空间中为应用程序创建一个 secret，并将 Cryostat 配置为引用该 secret。要为证书创建 secret，请运行以下命令：

```
oc create secret generic myapp-cert --from-file=tls.crt=/path/to/cert.pem
```

注意

证书必须采用 **.pem** 文件格式。

c. 创建 Cryostat 实例时，将 secret 添加到可信 TLS 证书列表中。如需更多信息，请参阅配置 TLS 证书。

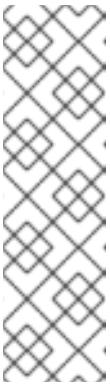
d. 要允许您的应用程序通过密码身份验证以外的方法验证 Cryostat 是否连接到它们，请启用 TLS 客户端身份验证：

```
-Dcom.sun.management.jmxremote.ssl.need.client.auth=true
-Djavax.net.ssl.trustStore=</path/to/truststore>
-Djavax.net.ssl.trustStorePassword=<password>
```

注意

TLS 客户端身份验证需要 Red Hat OpenShift 的 cert-manager operator。

e. 如果您将 TLS 客户端身份验证用于远程 JMX 连接，应用程序信任存储必须包含 Cryostat 证书。Cryostat operator cert-manager 集成为 Cryostat 部署创建一个自签名证书。此证书位于 < **cryostat>-tls** secret 中，其中 <cryostat> 是您创建的 Cryostat 实例的名称。



注意

cert-manager Operator 还将 Java 密钥存储信任存储放在 secret 中。

要在应用程序部署中挂载此信任存储，请运行以下命令，将 "<myapp>" 替换为应用程序部署的名称，将 "<cryostat>" 替换为 Cryostat 实例的名称：

```
oc set volumes deploy <myapp> --add --name=truststore \
  --secret-name=<cryostat>-tls --sub-path=truststore.p12 \
  --mount-path=/var/run/secrets/<myapp>/truststore.p12
```

f. Cryostat operator 生成 truststore 密码，您可以在 < **cryostat>-keystore** secret 中找到。要将它作为环境变量挂载到应用程序部署中，请运行以下命令：

```
oc set env deploy <myapp> --from='secret/<cryostat>-keystore'
```

g. 为容器配置 Java 参数。运行以下命令：

```
-Dcom.sun.management.jmxremote.ssl.need.client.auth=true
-Djavax.net.ssl.trustStore=/var/run/secrets/<myapp>/truststore.p12
-Djavax.net.ssl.trustStorePassword="$(KEYSTORE_PASS)"
```

警告

如果您在测试环境中部署 Cryostat 和您的应用程序，您可能需要在没有 JMX 或 TLS 身份验证的情况下配置目标应用程序。您可以使用以下 Java 系统属性集完成此操作，但这种配置不安全且不推荐。

```
-Dcom.sun.management.jmxremote.port=<port_num>
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.authenticate=false
```

其他资源

- 配置 TLS 证书
- 存储并管理 JMX 凭证

3.4. 使用 CRYOSTAT 代理和 JMX 连接配置应用程序

您可以配置在 Java 虚拟机(JVM)上运行的目标应用程序，以使用 Cryostat 代理和 Java 管理扩展(JMX)连接的组合来检测和与目标应用程序通信。

您可以使用 Cryostat 代理来检测和与目标应用程序通信，并使用 JMX 公开 Java Flight Recorder (JFR)数据。

您必须将 Cryostat 代理配置为与 Cryostat 进行通信，并且代理可以通过 JMX 而不是通过 HTTP 访问。

先决条件

- 登录到您的 Cryostat web 控制台。
- 在项目中创建一个 Cryostat 实例。

流程

1. 安装 Cryostat 代理。对于使用 Maven 的应用构建，请使用 Cryostat 代理 JAR 文件更新应用程序 **pom.xml** 文件。

   pom.xml 文件示例

```
<project>
  ...
  <repositories>
    <repository>
      <id>redhat-maven-repository</id>
      <url>https://maven.repository.redhat.com/earlyaccess/all/</url>
    </repository>
  </repositories>
  ...
  <build>
    <plugins>
      <plugin>
        <artifactId>maven-dependency-plugin</artifactId>
        <version>3.3.0</version>
        <executions>
          <execution>
            <phase>prepare-package</phase>
            <goals>
              <goal>copy</goal>
            </goals>
            <configuration>
              <artifactItems>
                <artifactItem>
                  <groupId>io.cryostat</groupId>
                  <artifactId>cryostat-agent</artifactId>
                  <version>0.4.0.redhat-xxxxx</version>
```

```
              <classifier>shaded</classifier>
            </artifactItem>
          </artifactItems>
          <stripVersion>true</stripVersion>
        </configuration>
      </execution>
    </executions>
  </plugin>
 </plugins>
 ...
</build>
...
</project>
```

注意

在前面的示例中，将 **0.4.0.redhat-*xxxxx*** 替换为 Cryostat 代理的最新构建版本（例如 **0.4.0.redhat-00001**）。有关 Cryostat 代理的最新构建版本的详情，请参考 Red Hat Maven 存储库。

2. 修改应用程序部署：

Example

```
apiVersion: apps/v1
kind: Deployment
...
spec:
 ...
 template:
  ...
  spec:
   containers:
    - name: sample-app
      image: docker.io/myorg/myapp:latest
      env:
       - name: CRYOSTAT_AGENT_APP_NAME
         value: "myapp"
         # Replace this with the Kubernetes DNS record
         # for the Cryostat Service
       - name: CRYOSTAT_AGENT_BASEURI
         value: "http://cryostat.mynamespace.mycluster.svc:4180"
       - name: POD_IP
         valueFrom:
           fieldRef:
             fieldPath: status.podIP
       - name: CRYOSTAT_AGENT_CALLBACK
         value: "http://$(POD_IP):9977"
       - name: CRYOSTAT_AGENT_AUTHORIZATION
         # Replace "abcd1234" with a plain-text authentication token
         value: "Bearer abcd1234"
         # This environment variable is key to the Cryostat agent
         # and JMX "hybrid" setup.
         # Set the Cryostat agent to register itself with Cryostat
         # as reachable through JMX, rather than reachable through HTTP.
```

```
        - name: CRYOSTAT_AGENT_REGISTRATION_PREFER_JMX
          value: "true"
          # Configure the application to load the agent JAR file and
          # to enable JMX, so that the Cryostat agent can register
          # itself as reachable through JMX.
          # To configure authentication and SSL/TLS for the JMX
          # connections, see <1>.
        - name: JAVA_OPTS
          value: >-
            -javaagent:/deployments/app/cryostat-agent-shaded.jar
            -Dcom.sun.management.jmxremote.port=9091 (1)
      ports:
        - containerPort: 9977
          protocol: TCP
      resources: {}
  restartPolicy: Always
status: {}
```

<1>: 要为 JMX 连接配置身份验证和 SSL/TLS，并查看更多配置选项，请参阅使用 JMX 连接配置应用程序。

3. 要启用 Cryostat 来检测目标应用程序并连接到 Cryostat 代理，请配置 应用程序服务 ：

Example

```
apiVersion: v1
kind: Service
...
spec:
  ports:
    - name: "jfr-jmx"
      port: 9091
      targetPort: 9091
    - name: "cryostat-agent"
      port: 9977
      targetPort: 9977
...
```

其他资源

- 使用 Cryostat 代理配置应用程序
- 使用 JMX 连接配置应用程序

第 4 章 使用 CRYOSTAT 创建 JFR 记录

使用 Cryostat，您可以创建一个 JDK Flight Recorder (JFR)记录，该记录监控容器化应用程序中的 JVM 性能。另外，您可以为目标 JVM 应用程序生成活跃的 JFR 记录的快照，以捕获任何收集的数据（最多指向特定时间）。

4.1. 在 CRYOSTAT WEB 控制台中创建 JFR 记录

您可以创建一个 JFR 记录，以监控容器化应用程序中的 JVM 性能。创建 JFR 记录后，您可以启动 JFR 来捕获 JVM 的实时数据，如堆和非堆内存使用量。

先决条件

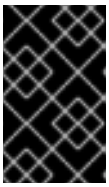
- 使用 OperatorHub 选项，在 Red Hat OpenShift 上安装 Cryostat 3.0。
- 在 Red Hat OpenShift 项目中创建一个 Cryostat 实例。
- 登录到您的 Cryostat web 控制台。
  - 您可以使用 Red Hat OpenShift Web 控制台检索 Cryostat 应用程序的 URL。

流程

1. 在 Cryostat web 控制台的 **Dashboard** 面板中，从 **Target** 列表中选择目标 JVM。

注意

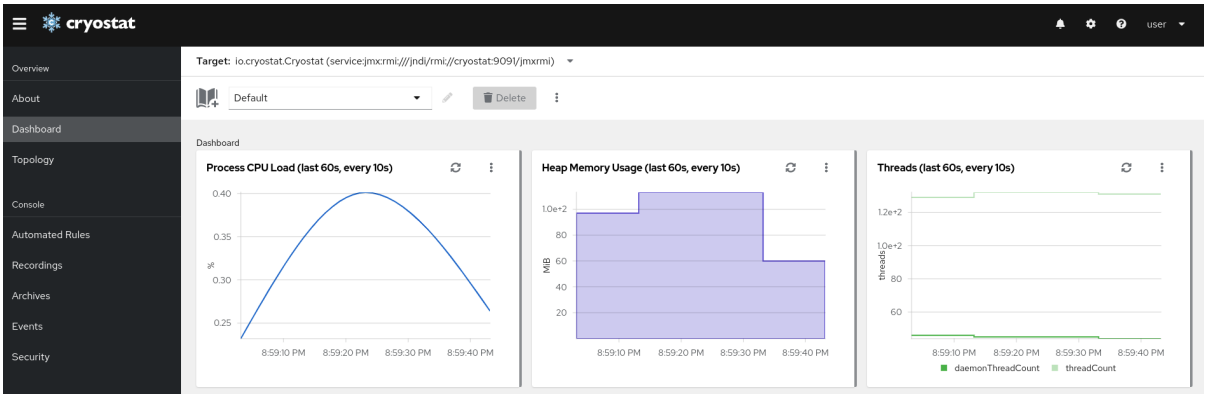
根据您配置目标应用程序的方式，您的目标 JVM 可能会使用 JMX 连接或代理 HTTP 连接。有关配置目标应用程序的更多信息，请参阅配置 Java 应用程序。



重要

如果您的目标 JVM 使用代理 HTTP 连接，请确保将目标应用程序配置为加载 Cryostat 代理时将 **cryostat.agent.api.writes-enabled** 属性设置为 **true**。否则，Cryostat 代理无法接受启动和停止 JFR 记录的请求。

图 4.1. 为 Cryostat 实例选择 Target JVM 的示例



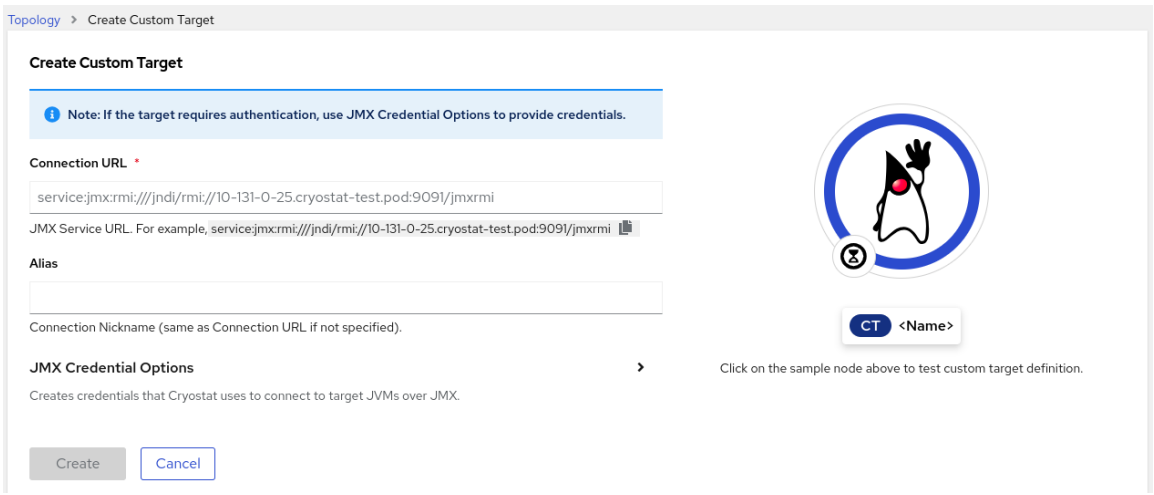
2. *可选：* 在 **Dashboard** 面板中，您可以创建一个目标 JVM。从 **Target** 列表中，单击 **Create Target**。此时会打开 **Create Custom Target** 窗口。
   a. 在 **Connection URL** 字段中，输入 JVM 的 Java 管理扩展(JMX)端点的 URL。

b. *可选：* 要测试您指定的 **连接 URL** 是否有效，请点击 **Click to test** sample node image。如果 **Connection URL** 存在问题，则会显示一条错误消息，其中提供了问题的描述以及要排除故障的指导。

c. *可选：* 在 **Alias** 字段中，为您的 JMX Service URL 输入别名。

d. 点 **Create**。

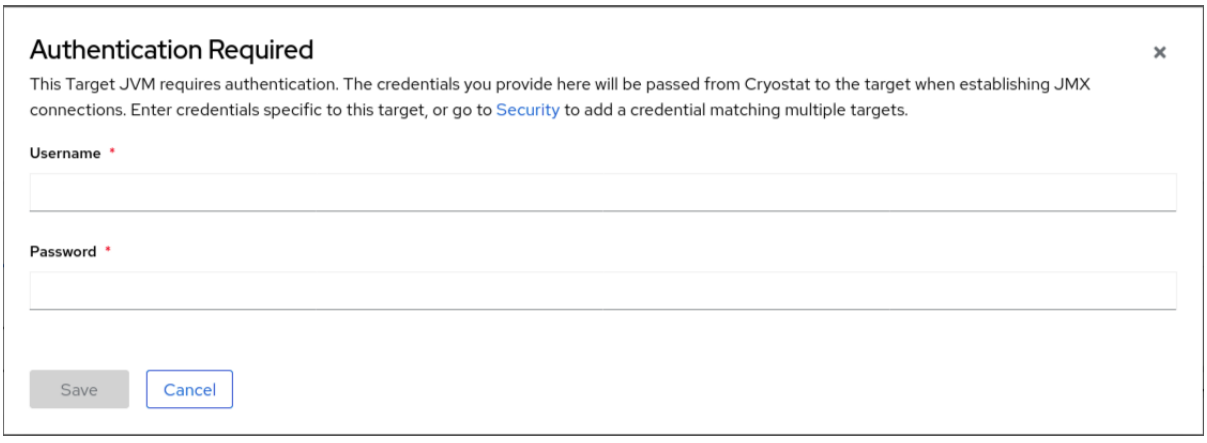
图 4.2. 创建自定义目标窗口



3. 在 Cryostat web 控制台的导航菜单中点 **Recordings**。

4. *可选：* 根据您如何配置目标 JVM，在 web 控制台中可能会打开 **Authentication Required** 对话框。在 **Authentication Required** 对话框中，输入您的 **Username** 和 **Password**。若要向目标 JVM 提供您的凭据，请单击 **Save**。

图 4.3. Cryostat 身份验证所需的窗口示例



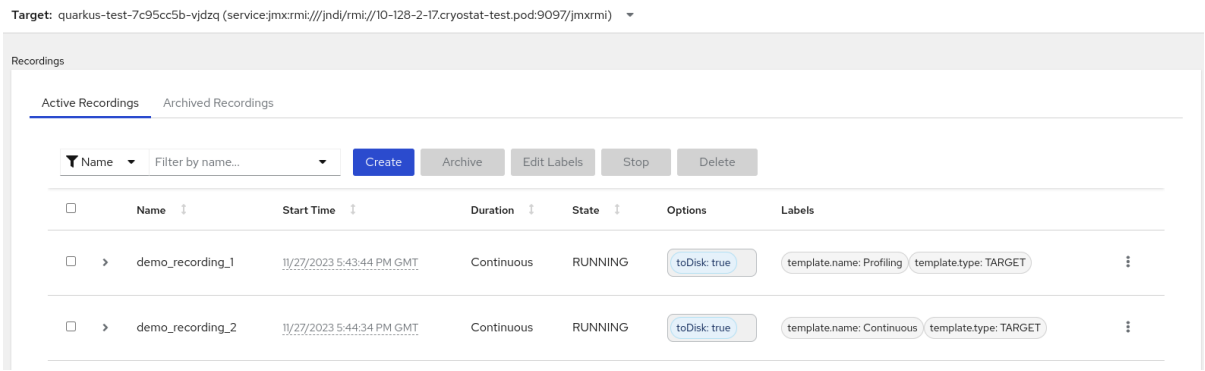
注意

如果所选目标 JMX 为 JMX 连接启用了安全套接字层(SSL)认证，则必须在提示时添加其证书。

Cryostat 在 Red Hat OpenShift 上的持久性卷声明(PVC)上加密并存储目标 JVM 应用程序的凭证。请参阅 Storing 和管理凭证 （使用 Cryostat 管理 JFR 记录）。

5. 在 **Active Recordings** 选项卡中，点 **Create**。

图 4.4. 创建活跃记录的示例



6. 在 **Custom Flight Recording** 选项卡中：

   a. 在 **Name** 字段中输入您要创建的记录的名称。如果您以无效格式输入名称，Web 控制台会显示错误消息。

   b. 如果您希望 Cryostat 自动重启现有记录，**如果记录已存在，请选择 Restart**。

注意

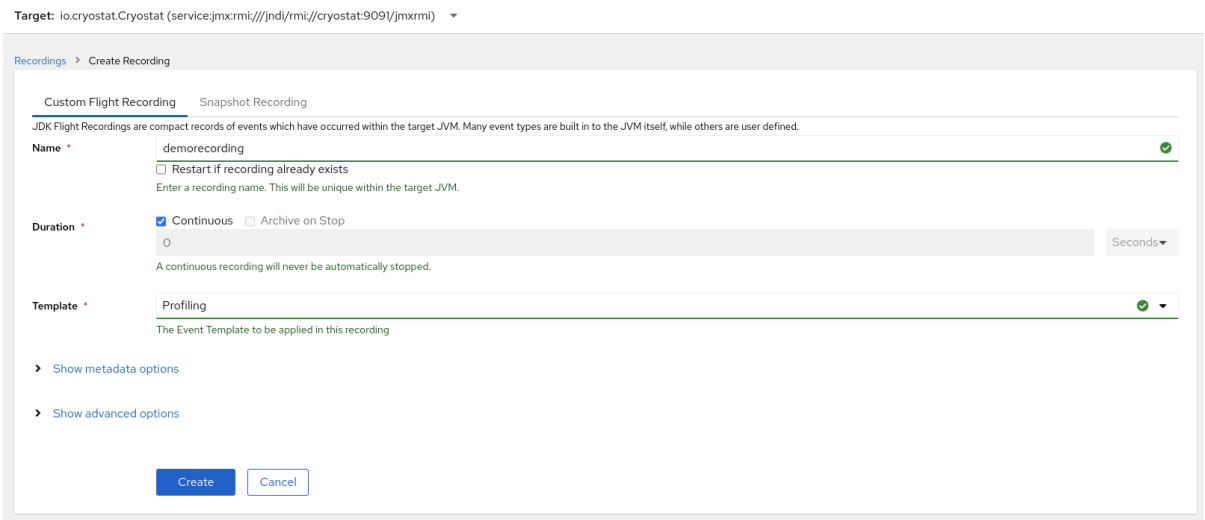
如果您输入已存在名称，但没有选择 **Restart （如果记录已存在）**，Cryostat 会在点 **Create** 按钮时拒绝创建自定义记录。

   c. 在 **Duration** 字段中，选择是否希望此记录在指定的持续时间后停止，或者在没有停止的情况下持续运行。如果您希望 Cryostat 在记录停止后自动归档您的新 JFR 记录，请点击 **Archive on Stop**。

   d. 在 **Template** 字段中，选择要用于记录的模板。

以下示例显示了持续 JVM 监控，您可以通过从 **Duration** 字段上方选择 continuous 来启用这些监控。**此设置意味着记录将继续，直到您手动停止记录。示例还显示 Template 字段中的 Profiling 模板选择。这为 JFR 记录提供了额外的 JVM 信息，以进行故障排除。**

图 4.5. 创建自定义动态记录示例



7. **要访问更多选项，请点以下可扩展超链接：**

   - **显示高级选项，您可以在其中选择额外选项来自定义 JFR 记录。**

- 显示元数据选项，您可以在其中在 JFR 记录中添加自定义标签和元数据。

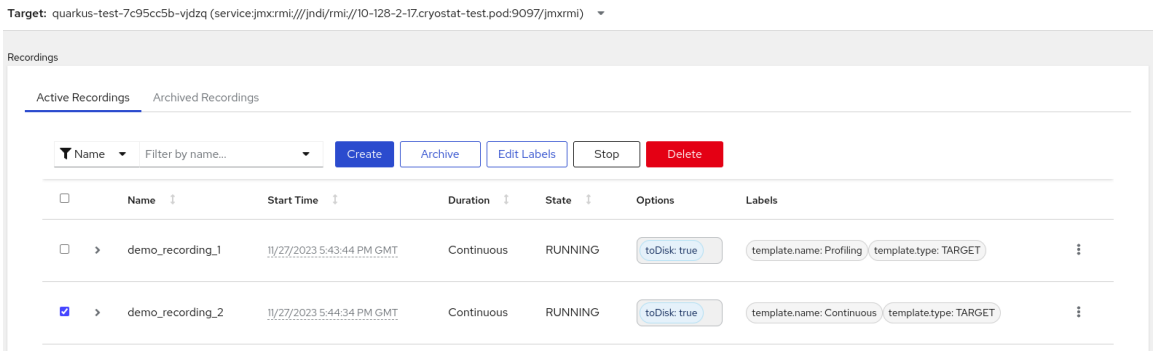
8. 要创建您的 JFR 记录，请点击 Create。Active Recordings 选项卡会打开并列出您的 JFR 记录。
   您的活跃 JFR 记录开始收集容器化应用内目标 JVM 位置的数据。如果您为 JFR 记录指定了固定持续时间，则目标 JVM 会在达到固定持续时间设置时停止记录。否则，您必须手动停止记录。

9. *可选：* 在 Active Recording 选项卡中，您还可以停止记录。

   a. 选中 JFR 记录名称旁边的复选框。在 Active Recordings 选项卡中的工具栏中，Cryostat web 控制台会激活 Stop 按钮。

   b. 点 Stop。JFR 采用 **STOPPED** 状态，因此它停止监控目标 JVM。JFR 仍然显示在Active Recording 选项卡下。

   图 4.6. 停止活跃记录的示例



重要

在以下情况下可能会丢失 JFR 记录数据：

- 目标 JVM 失败
- 目标 JVM 重启
- 目标 JVM Red Hat OpenShift Deployment 已缩减

归档您的 JFR 记录，以确保您不会丢失 JFR 记录的数据。

其他资源

- 请参阅 上传 SSL 证书（使用 Cryostat 管理 JFR 记录）。
- 请参阅 Archiving JDK Flight Recorder (JFR)记录（使用 Cryostat 管理 JFR 记录）。

4.2. 从活跃记录创建快照

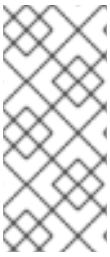
您可以对一个活跃的 JFR 记录进行快照，以便为您的目标 JVM 应用捕获任何收集的数据（最多特定的时间点）。快照类似于检查点标记，它在正在运行的 JFR 记录中具有给定时间段的起点和端点。

快照存储在目标 JVM 应用的内存中。这与一个存档不同，其中 Cryostat 将存档存储在云存储磁盘上，这是用于存储 JFR 记录数据的更持久解决方案。

如果要在活跃的 JFR 记录中试验不同的配置更改，您可以对记录进行快照。

当您为您的 JFR 记录创建快照时，Cryostat 会创建一个名为 snapshot - *<snapshot_number>* 的新目标 JVM，其中 *<snapshot_number>* 是 Cryostat 自动分配给您的快照的数字。

目标 JVM 将快照识别为活跃的记录。Cryostat 在 **STOPPED** 状态中设置任何 JFR 快照，这意味着 JFR 快照不会将新数据记录到目标 JVM。根据 JFR 配置，无论拍摄快照数量如何，活跃的 JFR 记录都可以继续监控目标 JVM。



注意

对于您为目标 JVM 应用程序持续监控而设置的 JFR 记录，请确保创建归档的记录以避免丢失 JFR 记录数据。

如果您选择进行常规快照来存储 JFR 记录数据，则目标 JVM 应用程序可能会释放一些数据存储空间，方法是将旧的记录数据替换为较新的记录数据。

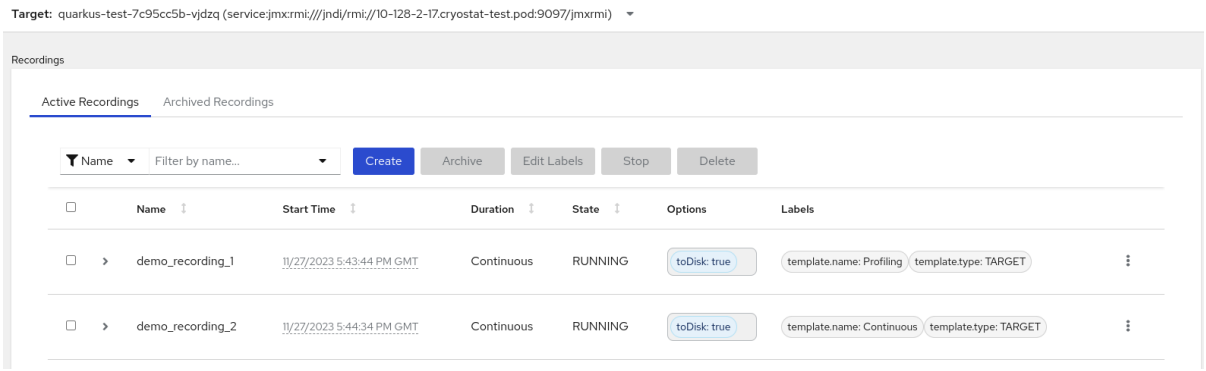
先决条件

- 为 Cryostat 实例输入您的身份验证详情。
- 创建目标 JVM 记录，并输入您验证的详细信息以访问 记录 菜单。请参阅 创建 JDK Flight Recorder (JFR)记录 （使用 Cryostat 创建 JFR 记录）。

流程

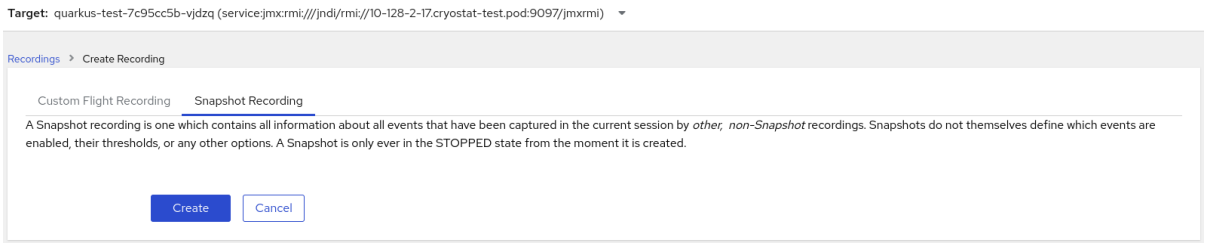
1. 在 Active Recordings 选项卡中，点 Create 按钮。在 web 控制台中打开一个新窗口。

   图 4.7. 创建活跃记录的示例



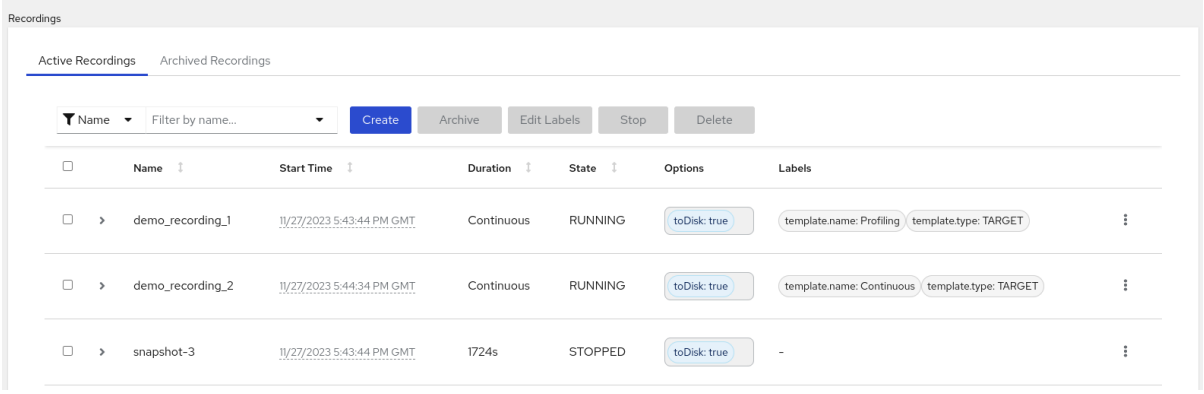
2. 点 Snapshot Recording 选项卡。

   图 4.8. 创建快照记录示例



3. 点 Create。Active Recordings 表会打开，并列出您的 JFR 快照记录。以下示例显示了名为 **snapshot-3** 的 JFR 快照记录。

图 4.9. 完成的快照记录示例



注意

您可以通过 *快照* 前缀从活跃记录列表中识别快照。

后续步骤

- 要归档您的 JFR 快照记录，请参阅 Archiving JDK Flight Recorder (JFR)记录。

4.3. JFR 记录的标签

当您在 Cryostat 3.0 上创建 JDK Flight Recorder (JFR)记录时，您可以通过指定一系列键值对在记录中添加元数据。

另外，您可以将自定义标签附加到目标 JVM 内部的 JFR 记录，以便您可以轻松识别和更好地管理 JFR 记录。

以下列表详细介绍了一些常见记录标签用例：

- 将元数据附加到您的 JFR 记录。
- 在包含相同标签的记录上执行批处理操作。
- 在记录上运行查询时使用标签。

您可以使用 Cryostat 创建监控容器化应用程序中 JVM 性能的 JFR 记录。另外，您可以为目标 JVM 应用程序生成活跃的 JFR 记录的快照，以捕获任何收集的数据（最多指向特定时间）。

4.3.1. 在 JFR 记录中添加标签

在 Cryostat 3.0 上创建 JFR 记录时，您可以使用标签将包含键值对的元数据添加到记录中。

Cryostat 将默认记录标签应用到创建的 JFR 记录。这些默认标签捕获 Cryostat 用于创建 JFR 记录的事件模板的信息。

您可以在 JFR 记录中添加自定义标签，以便您可以运行满足您的需要的特定查询，如识别特定的 JFR 记录或在具有相同应用标签的记录上执行批处理操作。

先决条件

- 登录到您的 Cryostat web 控制台。
- 为您的 Cryostat 实例创建或选择目标 JVM。

流程

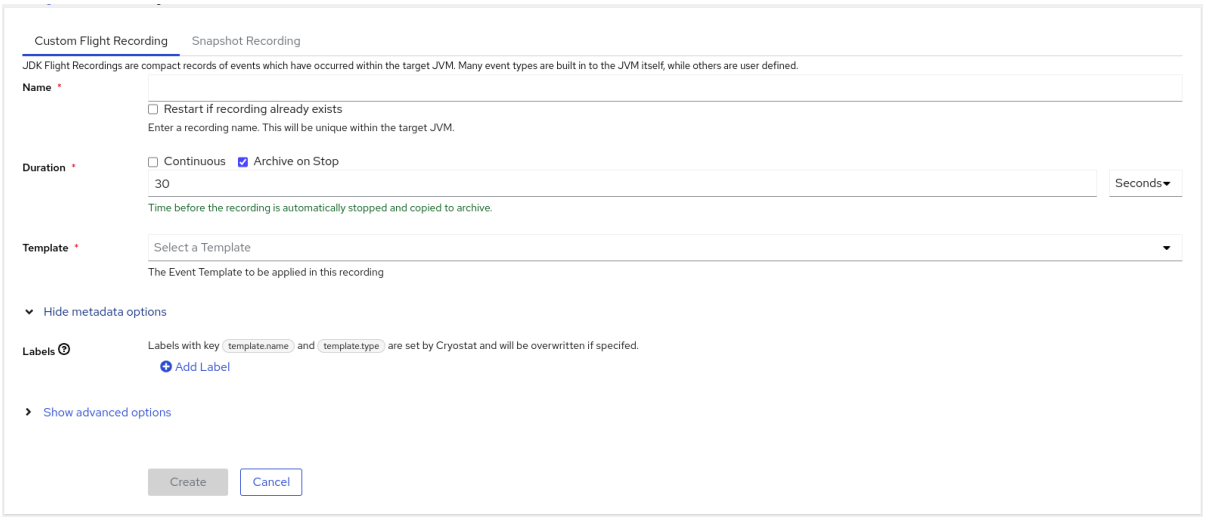
1. 在 Cryostat web 控制台中点 Recordings。
2. 在 Active Recordings 选项卡下，点 Create。
3. 在 Custom Flight Recording 选项卡中，展开 Show metadata options。

注意

在 Custom Flight Recording 选项卡中，您必须完成所有标记为星号的强制字段。

4. 点 Add label。

图 4.10. Custom Flight Recording 选项卡下显示的 Add Label 按钮



5. 在提供的 Key 和 Value 字段中输入值。例如，如果要在 Key 字段中输入问题，可以在 Key 字段中输入问题，然后在 Value 字段中输入问题类型。
6. 点 Create 创建您的 JFR 记录。然后，您的记录会在 Active Recordings 选项卡中显示，以及任何指定的记录标签和自定义标签。

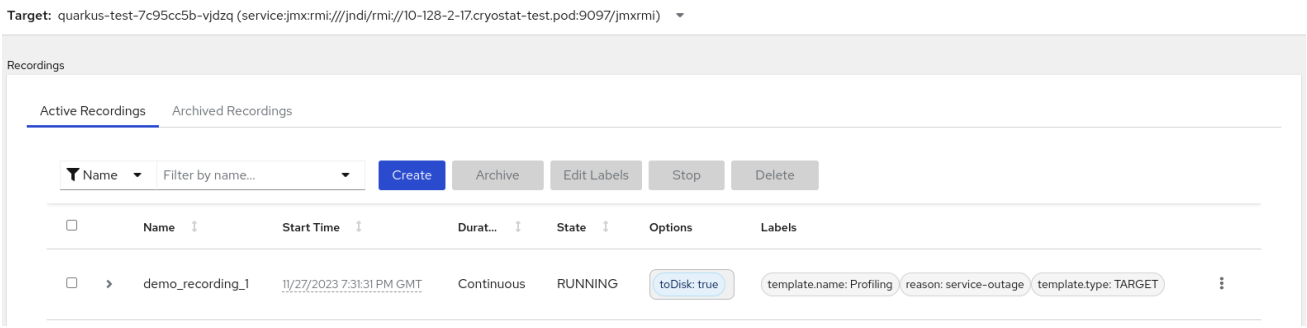
提示

您可以从 Archives 菜单访问归档的 JFR 记录。请参阅将 JFR 记录上传到 Cryostat 归档位置（使用 Cryostat 管理 JFR 记录）。

Example

以下示例显示了两个默认记录标签 **template.name: Profiling** and **template.type: TARGET**，以及一个自定义标签 **reason:service-outage**。

图 4.11. 带有定义的记录标签和自定义标签的活跃记录示例



4.3.2. 为您的 JFR 记录编辑标签

在 Cryostat web 控制台中，您可以进入 Recordings 菜单，然后编辑 JFR 记录的标签及其元数据。您还可以编辑上传到存档的 JFR 记录的标签和元数据。

先决条件

- 登录到您的 Cryostat web 控制台。
- 创建 JFR 记录并将标签附加到此记录。

流程

1. 在 Cryostat web 控制台中，点 Recording 菜单。
2. 在 Active Recordings 选项卡中，找到您的 JFR 记录，然后选中它旁边的复选框。
3. 点击 Edit Labels。在 Cryostat web 控制台中打开Edit Recording Label 窗格，可用于为 JFR 记录添加、编辑或删除标签。

   提示

   您可以选择每个记录旁边的复选框来选择多个 JFR 记录。如果要批量编辑包含相同标签的记录，或向多个记录添加新的相同的标签，点 Edit Labels 按钮。

4. *可选*：您可以从Edit Recording Labels 窗格中执行以下操作：
   a. 单击 Add 以创建标签。
   b. 点标签旁边的 X 来删除标签。
   c. 通过修改字段中的任何内容来编辑标签。编辑内容后，字段中会显示一个绿色勾号来指示编辑。
5. 点击 Save。
6. *可选*：您可以通过完成以下步骤来归档您的 JFR 记录及其标签：
   a. 选中记录名称旁边的复选框。
   b. 点 归档 按钮。您可以在 Archived Recordings 选项卡下找到您的记录。
   使用标签归档记录，当您想要在以后的阶段找到记录时，您可以增强您的搜索功能。您还可以将额外标签添加到上传到 Cryostat 归档的任何记录中。

注意

Cryostat 使用记录保留任何标签，以便进行存档记录的生命周期。

验证

- **在 Active Recordings 选项卡中，检查您的更改是否显示在 Labels 部分以了解您的记录。**

其他资源

- 归档 JDK Flight Recorder (JFR)记录（使用 Cryostat 管理 JFR 记录）

更新于 2024-07-02