OpenShift Container Platform 4.17

硬件加速器

硬件加速器

Last Updated: 2025-10-16

OpenShift Container Platform 4.17 硬件加速器

硬件加速器

Legal Notice

Abstract

本文档提供了有关安装和配置 Red Hat OpenShift AI 支持的 GPU Operator 的说明，以提供硬件加速功能，以创建人工智能和机器学习(AI/ML)应用程序。

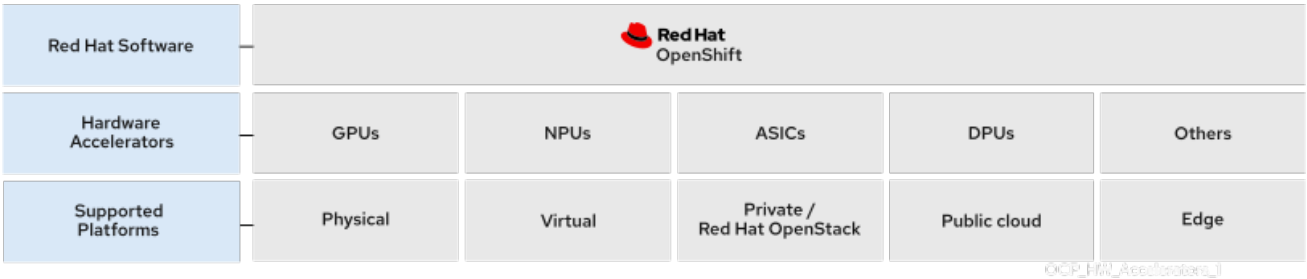
Table of Contents

第 1 章 关于硬件加速器

专用硬件加速器在新兴人工智能和机器学习(AI/ML)行业中发挥了关键作用。具体来说，硬件加速器对于培训以及支持这种新技术的大型语言和其他基础模型至关重要。数据科学家、数据工程师、ML 工程师和开发人员可以利用专门的硬件加速数据密集型转换和模型开发及服务。其中大多数生态系统都是开源的，一些贡献合作伙伴和开源基础。

Red Hat OpenShift Container Platform 支持卡和外围硬件，添加组成硬件加速器的处理单元：

- 图形处理单元(GPU)
- Neural processing units (NPUs)
- 特定于应用程序的集成电路(ASIC)
- 数据处理单元(DPU)



专用硬件加速器为 AI/ML 开发提供了丰富的优点：

一个平台适用于所有功能

面向开发人员、数据工程师、数据科学家和 DevOps 的协作环境

使用 Operator 扩展功能

Operator 允许将 AI/ML 功能引入到 OpenShift Container Platform

混合云支持

对模型开发、交付和部署的内部支持

支持 AI/ML 工作负载

模型测试、迭代、集成、提升，并作为服务提供给生产环境中

红帽提供了一个优化的平台，可在 Linux (kernel 和 userspace) 和 Kubernetes 层的 Red Hat Enterprise Linux (RHEL)和 OpenShift Container Platform 平台中启用这些专用硬件加速器。为此，红帽在单个企业级 AI 应用平台中结合了 Red Hat OpenShift AI 和 Red Hat OpenShift Container Platform 的成熟功能。

硬件 Operator 使用 Kubernetes 集群的操作框架来启用所需的加速器资源。您还可以手动部署提供的设备插件或守护进程集。此插件在集群中注册 GPU。

某些专用硬件加速器设计为在必须维护安全环境以进行开发和测试的断开连接的环境中工作。

1.1. 硬件加速器

Red Hat OpenShift Container Platform 启用以下硬件加速器：

- NVIDIA GPU
- AMD Instinct® GPU

- Intel® Gaudi®

其他资源

- Red Hat OpenShift AI 简介
- NVIDIA GPU Operator on Red Hat OpenShift Container Platform
- AMD Instinct 加速器
- Intel Gaudi AI Accelerators

第 2 章 NVIDIA GPU 架构

NVIDIA 支持在 OpenShift Container Platform 上使用图形处理单元 (GPU) 资源。OpenShift Container Platform 是一个以安全为中心的、强化的 Kubernetes 平台，由红帽开发并提供支持，用于大规模部署和管理 Kubernetes 集群。OpenShift Container Platform 包括对 Kubernetes 的增强，以便用户可以轻松地配置和使用 NVIDIA GPU 资源来加快工作负载。

NVIDIA GPU Operator 利用 OpenShift Container Platform 中的 Operator 框架来管理运行 GPU 加速工作负载所需的 NVIDIA 软件组件的完整生命周期。

这些组件包括 NVIDIA 驱动程序（为了启用 CUDA）、GPU 的 Kubernetes 设备插件、NVID Container Toolkit、使用 GPU 特性发现(GFD)、基于 DCGM 的监控等的自动节点标记。

注意

NVIDIA GPU Operator 的支持仅由 NVIDIA 提供。有关从 NVIDIA 获取支持的更多信息，请参阅 NVIDIA 支持。

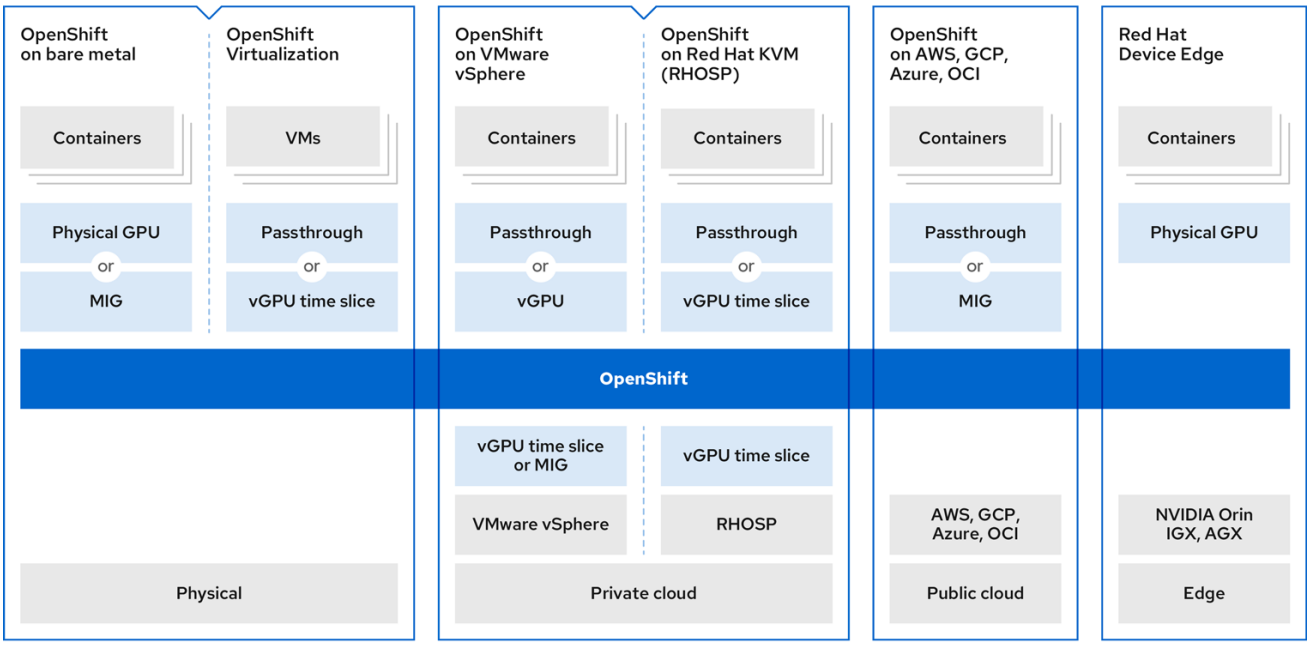
2.1. NVIDIA GPU 先决条件

- 包括至少一个 GPU worker 节点的，可正常工作的 OpenShift 集群。
- 以 **cluster-admin** 身份访问 OpenShift 集群，以执行必要的步骤。
- 已安装 OpenShift CLI (**oc**)。
- 已安装节点功能发现 (NFD) Operator 并创建了 **nodefeaturediscovery** 实例。

2.2. NVIDIA GPU 启用

下图显示了如何为 OpenShift 启用 GPU 架构：

图 2.1. NVIDIA GPU 启用



注意

从 NVIDIA Ampere 系列开始，在 GPU 上支持 MIG。有关支持 MIG 的 GPU 列表，请参阅 NVIDIA MIG 用户指南。

2.2.1. GPU 和裸机

您可以在 NVIDIA 认证的裸机服务器上部署 OpenShift Container Platform，但有一些限制：

- control plane 节点可以是 CPU 节点。
- Worker 节点必须是 GPU 节点，只要 AI/ML 工作负载在这些 worker 节点上执行。
  另外，worker 节点可以托管一个或多个 GPU，但它们必须是相同的类型。例如，一个节点可以有两个 NVIDIA A100 GPU，但不支持在一个节点中带有一个 A100 GPU 和一个 T4 GPU。
  Kubernetes 的 NVIDIA 设备插件不支持在同一节点上混合不同的 GPU 模型。
- 在使用 OpenShift 时，请注意，需要一个、三个或更多个服务器。不支持带有两个服务器的集群。单一服务器部署称为单一节点 openShift (SNO)，使用此配置的 OpenShift 环境不具有高可用性。

您可以选择以下方法之一来访问容器化 GPU：

- GPU passthrough（GPU 透传）
- 多实例 GPU (MIG)

其他资源

- Red Hat OpenShift on Bare Metal Stack

2.2.2. GPU 和虚拟化

虽然许多开发人员和企业都在转型到容器化应用程序和无服务器基础架构，但仍然有大量对在虚拟机(VM)上运行的应用程序进行开发和维护的需求。Red Hat OpenShift Virtualization 提供此功能，使企业能够将虚拟机合并到集群中的容器化工作流中。

您可以选择以下方法之一将 worker 节点连接到 GPU：

- 用于访问和使用虚拟机(VM)中的 GPU 硬件的 GPU 透传。
- 当 GPU 计算的容量没有因为工作负载而饱和时，可以进行 GPU (vGPU) 时间分片。

其他资源

- 带有 OpenShift Virtualization 的 NVIDIA GPU Operator

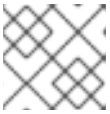
2.2.3. GPU 和 vSphere

您可以在可托管不同 GPU 类型的 NVIDIA 认证的 VMware vSphere 服务器上部署 OpenShift Container Platform。

如果虚拟机使用了 vGPU 实例，必须在 hypervisor 中安装 NVIDIA GPU 驱动程序。对于 VMware vSphere，此主机驱动程序以 VIB 文件的形式提供。

可分配给 worker 节点虚拟机的最大 vGPU 数量取决于 vSphere 的版本：

- vSphere 7.0：每个虚拟机最多 4 个 vGPU
- vSphere 8.0：每个虚拟机最大 8 个 vGPU



注意

vSphere 8.0 引入了对与一个虚拟机关联的多个完整或部分同配置集的支持。

您可以选择以下方法之一将 worker 节点附加到 GPU：

- 用于访问和使用虚拟机(VM)中的 GPU 硬件的 GPU 透传
- 当不需要所有 GPU 时，可以使用 GPU (vGPU) 时间分片

与裸机部署类似，需要一个或多个服务器。不支持带有两个服务器的集群。

其他资源

- 带有 NVIDIA vGPU 的 VMware vSphere 上的 OpenShift Container Platform

2.2.4. GPU 和 Red Hat KVM

您可以在基于 NVIDIA 认证的虚拟机 (KVM) 服务器上使用 OpenShift Container Platform。

与裸机部署类似，需要一个或多个服务器。不支持带有两个服务器的集群。

但是，与裸机部署不同，您可以在服务器中使用不同类型的 GPU。这是因为您可以将这些 GPU 分配给作为 Kubernetes 节点的不同虚拟机。唯一的限制是，一个 Kubernetes 节点在自己本身上必须具有相同的 GPU 类型。

您可以选择以下方法之一来访问容器化 GPU：

- 用于访问和使用虚拟机(VM)中的 GPU 硬件的 GPU 透传
- 当不需要所有 GPU 时，可以使用 GPU (vGPU) 时间分片

要启用 vGPU 功能，必须在主机级别安装特殊驱动程序。这个驱动程序作为 RPM 软件包提供。对于 GPU 透传分配，不需要这个主机驱动程序。

2.2.5. GPU 和 CSP

您可以将 OpenShift Container Platform 部署到主要的云服务供应商 (CSP) 之一：Amazon Web Services (AWS)、Google Cloud Platform (GCP)或 Microsoft Azure。

有两种操作模式：完全管理的部署和自我管理的部署。

- 在完全管理的部署中，红帽与 CSP 合作实现了一切自动化。您可以通过 CSP Web 控制台请求 OpenShift 实例，集群会自动创建并完全由红帽管理。您不必担心环境中节点故障或错误。红帽完全负责维持集群的正常运行时间。完全管理的服务在 AWS、Azure 和 GCP 上提供。对于 AWS，OpenShift 服务称为 ROSA (Red Hat OpenShift Service on AWS)。对于 Azure，该服务名为 Azure Red Hat OpenShift。对于 GCP，该服务在 GCP 上称为 OpenShift Dedicated。
- 在自我管理的部署中，您需要自行实例化和维护 OpenShift 集群。红帽提供了 OpenShift-install 工具，以支持在本例中部署 OpenShift 集群。自我管理的服务可全局提供给所有 CSP。

重要的是，此计算实例是一个 GPU 加速的计算实例，并且 GPU 类型与 NVIDIA AI Enterprise 支持的 GPU 列表匹配。例如，T4、V100 和 A100 是此列表的一部分。

您可以选择以下方法之一来访问容器化 GPU：

- 用于访问和使用虚拟机(VM)中的 GPU 硬件的 GPU 透传。
- 当不需要整个 GPU 时，可以进行 GPU (vGPU) 时间分片。

其他资源

- 云中的 Red Hat Openshift

2.2.6. GPU 和 Red Hat Device Edge

Red Hat Device Edge 提供对 MicroShift 的访问。MicroShift 提供了单节点部署的简单性和资源约束（边缘）计算所需的功能和服务。Red Hat Device Edge 满足在资源受限环境中部署的裸机、虚拟、容器化或 Kubernetes 工作负载的需求。

您可以在 Red Hat Device Edge 环境中的容器上启用 NVIDIA GPU。

您可以使用 GPU 透传来访问容器化 GPU。

其他资源

- 如何在 Red Hat Device Edge 中使用 NVIDIA GPU 加速工作负载

2.3. GPU 共享方法

红帽和 NVIDIA 已开发了 GPU 并发和共享机制，以简化企业级 OpenShift Container Platform 集群上的 GPU 加速计算。

应用程序通常会有不同的计算要求，这可能会使 GPU 使用率不足。为每个工作负载提供正确的计算资源数量对于降低部署成本和最大化 GPU 使用率至关重要。

用于提高 GPU 使用率的并发机制，范围从编程模型 API 到系统软件和硬件分区，包括虚拟化。以下列表显示了 GPU 并发机制：

- 计算统一设备架构 (CUDA) 流
- Time-slicing（时间分片）
- CUDA 多进程服务 (MPS)
- 多实例 GPU (MIG)
- 使用 vGPU 的虚拟化

在将 GPU 并发机制用于不同的 OpenShift Container Platform 场景时，请考虑以下 GPU 共享建议：

裸机

vGPU 不可用。考虑使用支持 MIG 的卡。

虚拟机

vGPU 是最佳选择。

裸机中的较旧的 NVIDIA 卡中没有 MIG

考虑使用 time-slicing。

具有多个 GPU 的虚拟机，您需要透传和 vGPU

考虑使用单独的虚拟机。

使用 OpenShift Virtualization 和多个 GPU 的裸机

对于托管的虚拟机，考虑使用透传，对呀容器，考虑使用时间分片。

其他资源

- 提高 GPU 利用率

2.3.1. CUDA 流

Compute Unified Device Architecture (CUDA) 是由 NVIDIA 开发的并行计算平台和编程模型，用于 GPU 上的常规计算。

流是 GPU 上问题顺序执行的操作序列。CUDA 命令通常在默认流中按顺序执行，任务在前面的任务完成后才会启动。

跨不同流的异步处理操作允许并行执行任务。在一个流中发布的任务之前、期间或另一个任务签发到另一个流后运行。这允许 GPU 以任何规定的顺序同时运行多个任务，从而提高性能。

其他资源

- 异步并发执行

2.3.2. Time-slicing（时间分片）

当您运行多个 CUDA 应用程序时，在超载 GPU 上调度 GPU 时间的交集工作负载。

您可以通过为 GPU 定义一组副本来启用 Kubernetes 上的 GPU 的时间，每个副本都可以独立分发到 pod 来运行工作负载。与多实例 GPU (MIG) 不同，在副本之间不存在内存或故障隔离，但对于某些工作负载，这优于根本不进行共享。在内部，GPU 时间分片用于从同一底层 GPU 将多个工作负载分配到不同的副本。

对于时间分片，您可以应用一个集群范围的默认配置。您还可以应用特定于节点的配置。例如，您只能将时间分片配置应用到具有 Tesla T4 GPU 的节点，而不能将节点改为使用其他 GPU 模型。

您可以通过应用集群范围的默认配置来组合这两种方法，然后标记节点以授予这些节点接收特定于节点的配置。

2.3.3. CUDA 多进程服务

CUDA 多进程服务 (MPS) 允许单个 GPU 使用多个 CUDA 进程。进程在 GPU 上并行运行，消除了 GPU 计算资源的饱和。MPS 还支持并发执行或重叠内核操作和从不同进程复制的内存，以增强利用率。

其他资源

- CUDA MPS

2.3.4. 多实例 GPU

使用多实例 GPU (MIG)，您可以将 GPU 计算单元和内存分成多个 MIG 实例。每个实例都代表了从系统的角度来看的独立 GPU 设备，并可连接到节点上运行的任何应用程序、容器或虚拟机。使用 GPU 的软件将每个 MIG 实例视为单独的 GPU。

当您有一个不需要整个 GPU 完全功能的应用程序时，MIG 很有用。新的 NVIDIA Ampere 架构的 MIG 功能允许您将硬件资源分成多个 GPU 实例，每个实例都可作为独立的 CUDA GPU 提供给操作系统。

NVIDIA GPU Operator 版本 1.7.0 及更高版本为 A100 和 A30 Ampere 卡提供 MIG 支持。这些 GPU 实例旨在支持最多 7 个独立的 CUDA 应用程序，以便它们与专用硬件资源完全隔离。

其他资源

- NVIDIA 多实例 GPU 用户指南

2.3.5. 使用 vGPU 的虚拟化

虚拟机 (VM) 可以使用 NVIDIA vGPU 直接访问单个物理 GPU。您可以创建虚拟 GPU，供虚拟机在企业之间共享，并由其他设备访问。

此功能将 GPU 性能与 vGPU 提供的管理和安全优势相结合。vGPU 提供的其他好处包括虚拟机环境的主动管理和监控、混合 VDI 和计算工作负载的工作负载平衡以及多个虚拟机之间的资源共享。

其他资源

- 虚拟 GPU

2.4. OPENSHIFT CONTAINER PLATFORM 的 NVIDIA GPU 功能

NVIDIA Container Toolkit

NVIDIA Container Toolkit 可让您创建并运行 GPU 加速容器。工具包包括容器运行时库和工具，用于自动配置容器以使用 NVIDIA GPU。

NVIDIA AI Enterprise

NVIDIA AI Enterprise 是端到端的云原生 AI 和数据分析软件套件，由 NVIDIA 认证系统进行优化、认证和支持。
NVIDIA AI Enterprise 包括对 Red Hat OpenShift Container Platform 的支持。支持以下安装方法：

- 带有 GPU Passthrough 的裸机或 VMware vSphere 上的 OpenShift Container Platform。
- 带有 NVIDIA vGPU 的 VMware vSphere 上的 OpenShift Container Platform。

GPU 功能发现

NVIDIA GPU Feature Discovery for Kubernetes 是一个软件组件，可让您为节点上可用的 GPU 自动生成标签。GPU 功能发现使用节点功能发现(NFD)来执行此标记。
Node Feature Discovery Operator (NFD)通过使用硬件特定信息标记节点来管理 OpenShift Container Platform 集群中硬件功能和配置的发现。NFD 使用特定于节点的属性标记主机，如 PCI 卡、内核、操作系统版本等。

您可以通过搜索 "Node Feature Discovery" 在 Operator Hub 中找到 NFD Operator。

带有 OpenShift Virtualization 的 NVIDIA GPU Operator

到目前为止，GPU Operator 只置备了 worker 节点来运行 GPU 加速的容器。现在，GPU Operator 也可以用来置备 worker 节点来运行 GPU 加速的虚拟机 (VM)。

您可以根据将 GPU 工作负载配置为在这些节点上运行，将 GPU Operator 配置为将不同的软件组件部署到 worker 节点。

GPU 监控仪表板

您可以安装监控仪表板，在 OpenShift Container Platform Web 控制台的集群 **Observe** 页面中显示 GPU 用量信息。GPU 使用率信息包括可用 GPU 数、功耗（watts）、温度（Celsius）、利用率（百分比）以及其他每个 GPU 的指标。

其他资源

- NVIDIA 认证系统
- NVIDIA AI Enterprise
- NVIDIA Container Toolkit
- 启用 GPU 监控仪表板
- OpenShift Container Platform 中的 MIG 支持
- OpenShift 中的 NVIDIA GPU 时间分片
- 在断开连接的或 airgapped 环境中部署 GPU Operator
- Node Feature Discovery Operator

第 3 章 AMD GPU OPERATOR

AMD Instinct GPU 加速器与 OpenShift Container Platform 集群中的 AMD GPU Operator 相结合，可让您无缝利用计算功能进行机器学习、生成 AI 和 GPU 加速的应用程序。

本文档提供了启用、配置和测试 AMD GPU Operator 所需的信息。如需更多信息，请参阅 AMD Instinct™ 加速器。

3.1. 关于 AMD GPU OPERATOR

AMD GPU Operator 的硬件加速功能为数据科学家和开发人员提供增强的性能和成本效率，使用 Red Hat OpenShift AI 创建人工智能和机器学习(AI/ML)应用程序。加快 GPU 功能的特定区域可以最小化 CPU 处理和内存用量，提高整体应用程序速度、内存消耗和带宽限制。

3.2. 安装 AMD GPU OPERATOR

作为集群管理员，您可以使用 OpenShift CLI 和 Web 控制台安装 AMD GPU Operator。这是一个多步骤步骤，需要安装 Node Feature Discovery Operator、内核模块管理 Operator，然后是 AMD GPU Operator。使用以下步骤安装 Operator 的 AMD 社区版本。

后续步骤

1. 安装 Node Feature Discovery Operator。
2. 安装内核模块管理 Operator。
3. 安装和配置 AMD GPU Operator。

3.3. 测试 AMD GPU OPERATOR

使用以下步骤测试 ROCmInfo 安装并查看 AMD MI210 GPU 的日志。

流程

1. 创建测试 ROCmInfo 的 YAML 文件：

```
$ cat << EOF > rocminfo.yaml

apiVersion: v1
kind: Pod
metadata:
 name: rocminfo
spec:
 containers:
 - image: docker.io/rocm/pytorch:latest
   name: rocminfo
   command: ["/bin/sh","-c"]
   args: ["rocminfo"]
   resources:
    limits:
      amd.com/gpu: 1
    requests:
```

```
      amd.com/gpu: 1
 restartPolicy: Never
EOF
```

2. 创建 **rocminfo** pod:

```
$ oc create -f rocminfo.yaml
```

输出示例

```
apiVersion: v1
pod/rocminfo created
```

3. 使用一个 MI210 GPU 检查 **rocmnfo** 日志：

```
$ oc logs rocminfo | grep -A5 "Agent"
```

输出示例

```
HSA Agents
==========
*******
Agent 1
*******
  Name:                    Intel(R) Xeon(R) Gold 6330 CPU @ 2.00GHz
  Uuid:                    CPU-XX
  Marketing Name:          Intel(R) Xeon(R) Gold 6330 CPU @ 2.00GHz
  Vendor Name:             CPU
--
Agent 2
*******
  Name:                    Intel(R) Xeon(R) Gold 6330 CPU @ 2.00GHz
  Uuid:                    CPU-XX
  Marketing Name:          Intel(R) Xeon(R) Gold 6330 CPU @ 2.00GHz
  Vendor Name:             CPU
--
Agent 3
*******
  Name:                    gfx90a
  Uuid:                    GPU-024b776f768a638b
  Marketing Name:          AMD Instinct MI210
  Vendor Name:             AMD
```

4. 删除 Pod：

```
$ oc delete -f rocminfo.yaml
```

输出示例

```
pod "rocminfo" deleted
```