硬件加速器

NPU

Alauda NPU Operator 构建

关于 Alauda Build of Hami

关于 **Alauda Build of Hami**

关于 Alauda Build 的 NVIDIA GPU 设备插件

关于 **Alauda Build 的 NVIDIA GPU 设备插件**

NPU

Alauda NPU Operator 构建

介绍

安装

前提条件

操作步骤

常见问题

Alauda NPU Operator 构建

介绍

安装

前提条件

操作步骤

常见问题

介绍

Kubernetes 通过 Device Plugins 提供对特殊硬件资源（例如 Ascend NPU）的访问。但是，要配置和管理拥有这些硬件资源的节点，还需要多个软件组件（例如驱动程序、container runtime 或其他库）。这些组件的安装过程复杂、困难且容易出错。NPU operator 使用 Kubernetes 中的 Operator Framework 自动管理配置 Ascend 设备所需的所有软件组件。这些组件包括 Ascend driver 和 firmware（用于支持集群的整个运行过程），以及 MindCluster device plug-in（用于支持任务调度、O&M 监控和故障恢复等集群运维操作）。通过安装相应组件，您可以管理 NPU 资源、优化 workload 调度，并将 training 和 inference 任务容器化，从而使 AI jobs 以容器形式部署并运行在 NPU 设备上。

有关更多详情，请参阅 NPU Operator ↗。

安装

目录

前提条件
- 联网安装
- 离线安装

操作步骤
- 下载集群插件
- 上传集群插件
- 安装 Alauda Build of NPU Operator
  - 验证
- 安装监控

常见问题
- 卸载 Alauda Build of NPU Operator 时需要注意什么？

前提条件

联网安装

1. **ACP** 版本：**v4.0** 或更高版本
2. 具有 **ACP** 集群的集群管理员访问权限

3. 确保 **NPU** 节点上存在 **bash** 工具。否则，驱动和固件安装脚本可能无法解析。
4. 工作节点操作系统要求
   - 运行 **NPU** 工作负载 的工作节点（或节点组）必须使用以下操作系统之一（Arm 架构）：
     - `openEuler 22.03 LTS`
     - `Ubuntu 22.04`
   - 仅运行 **CPU** 工作负载 的工作节点可以使用任意操作系统，因为 NPU operator 不会对未运行 NPU 工作负载的节点进行任何配置。
5. 支持的 **NPU** 硬件
   - 节点必须使用受支持的 NPU：
     - `Ascend 910B`
     - `Ascend 310P`
   - 有关详细的 OS 和硬件兼容性，请参见 MindCluster 文档↗
6. 必须安装 **`Alauda Build of Node Feature Discovery`** 集群插件。

离线安装

1. 离线安装除满足所有联网安装前提条件外，还需要额外的准备步骤。
2. 准备驱动和固件包以及 **MindIO SDK** 包。 下载以下软件包（如果不需要安装 MindIO，则无需下载 MindIO 包）：
   - 对于驱动和固件包，请在 npu-driver-installer↗ 的 GitCode 仓库中找到 `config.json` 文件，并通过其中提供的相应链接，按照要选择的版本、NPU 型号以及对应节点的 OS 架构下载软件包。
   - 对于 MindIO SDK 包，请在 npu-node-provision↗ 的 GitCode 仓库中找到 `config.json` 文件，并通过其中提供的相应链接，按照对应节点的 NPU 型号以及 OS 架构下载 SDK 包。
3. 将驱动和固件包的 **ZIP** 文件保存到执行离线安装的节点上的 **`/tmp/driver_pkg/`** 路径。
4. 将 **MindIO** 包的 **ZIP** 文件保存到执行离线安装的节点上的 **`/opt/openFuyao/mindio/`** 路径。（如果不需要安装 **MindIO**，请跳过此步骤。）

5. 检查目标节点是否包含以下工具。

- 对于使用 Yum 作为包管理器的系统，需要安装以下软件包："`jq wget unzip which net-tools pciutils gcc make kernel-devel-$(uname-r) kernel-headers-(uname-r) dkms"`。
- 对于使用 apt-get 作为包管理器的系统，需要安装以下软件包："`jq wget unzip debianutils net-tools pciutils gcc make dkms linux-headers-$(uname -r)`"。
- 对于使用 DNF 作为包管理器的系统，需要安装以下软件包："`jq wget unzip which net-tools pciutils gcc make kernel-devel-$(uname -r) kernel-headers-(uname-r) dkms"`。

操作步骤

下载集群插件

> **INFO**
>
> 你可以从 Customer Portal 网站上的 Marketplace 下载名为 `Alauda Build of NPU Operator` 和 `Alauda Build of Node Feature Discovery` 的应用。

注意：Volcano 集群插件可以暂时不安装。

上传集群插件

平台提供了 `violet` 命令行工具，用于上传从 Custom Portal Marketplace 下载的软件包。

有关详细信息，请参见 Upload Packages。

安装 Alauda Build of NPU Operator

1. 将标签 `masterselector=dls-master-node` 应用到所有主节点，将标签 `workerselector=dls-worker-node` 应用到所有工作节点。

```
kubectl label nodes {master-node-id} masterselector=dls-master-node
kubectl label nodes {worker-node-id} workerselector=dls-worker-node
```

2. 转到 `Administrator` -> `Marketplace` -> `Cluster Plugin` 页面，切换到目标集群，然后部署 `Alauda Build of NPU Operator` 集群插件。
   部署表单参数说明：

> **WARNING**
>
> 如果下表中列出的组件已经安装，请务必在部署时禁用相应按钮。

> **TIP**
>
> Ascend Operator、NodeD、ClusterD、Resilience Controller、MindIO TFT 和 MindIO ACP 默认不部署。仅在确有需要时再进行部署。

组件	默认值	说明
Driver	启用	是否安装驱动和固件。
Version	24.1.RC3	驱动和固件版本。必须从仓库目录 npu-driver-installer↗ 中选择版本号。
Ascend Device Plugin	启用	是否安装 Ascend Device Plugin。
Ascend Docker Runtime	启用	是否安装 Ascend Docker Runtime。
NPU exporter	启用	是否安装 NPU exporter。
Ascend Operator	禁用	是否安装 Ascend Operator。
NodeD	禁用	是否安装 NodeD。

组件	默认值	说明
ClusterD	禁用	是否安装 ClusterD。需要先安装 Volcano 集群插件。
Resilience Controller	禁用	是否安装 Resilience Controller。
MindIO TFT	禁用	是否安装 MindIO TFT。
MindIO ACP	禁用	是否安装 MindIO ACP。

验证

1. 首先，你可以在 `Alauda Build of NPU Operator` 集群插件页面看到“已安装”状态。
2. 等待 npu-driver pod 变为 Running 状态。离线安装大约需要 10 分钟，而联网安装要快得多。

```
kubectl -n kube-system get pod -w | grep npu-driver
```

3. 重启所有 NPU 节点。
4. 在 NPU 节点上运行以下命令。

```
npu-smi info
```

   确保显示正常。

5. 在主节点上运行以下命令。

```
kubectl get npuclusterpolicy cluster
```

   确保 `npuclusterpolicy` 的状态为 Ready。

6. 在业务集群的控制节点上，检查 NPU 节点是否存在可分配的 NPU 资源。运行以下命令：

```
kubectl get node  ${nodeName} -o=jsonpath='{.status.allocatable}'
# Example, the output contains: "huawei.com/Ascend310P":"1" (the specif
ic value depends on the number of NPU cards)
```

7. 运行验证工作负载。 创建 spec 文件：

```
key="huawei.com/Ascend310P" # For 310P
cat <<EOF > deploy-npu.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ascend-pytorch
spec:
  replicas: 1
  selector:
    matchLabels:
      service.cpaas.io/name: deployment-ascend-pytorch
  strategy:
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
    type: RollingUpdate
  template:
    metadata:
      labels:
        service.cpaas.io/name: deployment-ascend-pytorch
    spec:
      affinity: {}
      containers:
        - args:
            - |
              sleep infinity
          command:
            - /bin/bash
            - -c
          image: ascendai/pytorch:ubuntu-python3.8-cann8.0.rc1.beta1-py
torch2.1.0
          imagePullPolicy: Always
          name: ascend-pytorch
          resources:
            limits:
              cpu: 500m
              $key: "1"
              memory: 2Gi
            requests:
              cpu: 500m
              memory: 2Gi
      runtimeClassName: ascend
EOF
```

应用 spec：

```
kubectl apply -f deploy-npu.yaml
```

```
kubectl exec -it  deploy/ascend-pytorch -- bash
```

然后在容器中运行以下命令：

```
npu-smi info
```

确保显示正常。

安装监控

如果在安装 `Alauda Build of NPU Operator` 时部署了 NPU exporter 组件，请执行以下步骤创建监控面板。

1. 在集群的控制节点上执行命令。

```
cat << EOF | kubectl apply -f -
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    prometheus: kube-prometheus
    serviceMonitorSelector: prometheus
  name: npu-exporter-ai
  namespace: monitoring
spec:
  endpoints:
  - interval: 10s
    path: /metrics
    port: http
    targetPort: 8082
  namespaceSelector:
    matchNames:
    - npu-exporter
  selector:
    matchLabels:
      app: npu-exporter-svc
EOF
```

2. 你可以按照 Import Dashboard 导入 Grafana dashboard JSON 文件，将其转换为用于显示的监控仪表板。 该 JSON 文件位于 ascend-npu-dashboard ↗。

NOTE

Grafana dashboard JSON 文件中的标签不能包含中文字符，需要手动删除。 例如：

```
{
  "tags": [
    "ascend",
    "昇腾"
  ]
}
```

修改后：

```
{
  "tags": [
    "ascend"
  ]
}
```

常见问题

卸载 Alauda Build of NPU Operator 时需要注意什么？

即使卸载了 Alauda Build of NPU Operator，驱动程序仍可能存在于宿主机上。 在 NPU 节点上，执行以下命令卸载驱动程序：

```
/usr/local/Ascend/driver/script/uninstall.sh
```

关于 Alauda Build of Hami

异构 AI 计算虚拟化中间件（HAMi），前称为 k8s-vGPU-scheduler，是一个用于管理 k8s 集群中异构 AI 计算设备的“一体化”Chart。它能够实现异构 AI 设备在任务之间的共享能力。

Note

因为 Alauda Build of Hami 的发版周期与灵雀云容器平台不同，所以 Alauda Build of Hami 的文档现在作为独立的文档站点托管在 Alauda Build of Hami。

关于 Alauda Build 的 NVIDIA GPU 设备插件

NVIDIA 设备插件是 Kubernetes 的一个 Daemonset，允许您自动：

- 在集群的每个节点上暴露 GPU 数量
- 监控 GPU 的健康状态
- 在 Kubernetes 集群中运行支持 GPU 的容器

Note

因为 Alauda Build of NVIDIA GPU Device Plugin 的发版周期与灵雀云容器平台不同，所以 Alauda Build of NVIDIA GPU Device Plugin 的文档现在作为独立的文档站点托管在 Alauda Build of NVIDIA GPU Device Plugin。