Аппаратные ускорители

NPU

Alauda Build of NPU Operator

О Alauda Build of Hami

О Alauda Build of Hami

Heterogeneous AI Computing Virtualization Middleware (HAMi) — это универсальный чарт для управления гетерогенными AI-вычислительными устройствами в кластере k8s.

О плагине устройства NVIDIA GPU сборки Alauda

О плагине устройства NVIDIA GPU сборки Alauda

Плагин устройства NVIDIA для Kubernetes, позволяющий автоматически управлять GPU в кластере.

NPU

Alauda Build of NPU Operator

Введение

Установка

Предварительные требования

Процедура

FAQ

Alauda Build of NPU Operator

Введение

Kubernetes предоставляет доступ к специальным аппаратным ресурсам (таким как Ascend NPU) через Device Plugins. Однако для настройки и управления узлом с такими аппаратными ресурсами требуется несколько программных компонентов (например, драйверы, контейнерные рантаймы или другие библиотеки). Установка этих компонентов сложна, трудоемка и подвержена ошибкам. Оператор NPU использует Operator Framework в Kubernetes для автоматического управления всеми программными компонентами, необходимыми для настройки устройств Ascend. Эти компоненты включают драйвер и прошивку Ascend (которые поддерживают весь процесс работы кластеров), а также плагин устройства MindCluster (который поддерживает операции кластера, такие как планирование заданий, мониторинг O&M и восстановление после сбоев). Установив соответствующие компоненты, вы можете управлять ресурсами NPU, оптимизировать планирование рабочих нагрузок и контейнеризировать задачи обучения и инференса, чтобы AI-задачи могли развертываться и запускаться на устройствах NPU в виде контейнеров.

Для получения дополнительной информации обратитесь к NPU Operator↗.

Установка

Содержание

Предварительные требования

Онлайн-установка

1. **Версия ACP: v4.0 или выше**
2. **Доступ администратора к кластеру ACP**

3. **Убедитесь, что инструмент bash существует на узле NPU. В противном случае скрипт установки драйвера и прошивки может не быть корректно разобран.**
4. **Требования к операционной системе worker-узлов**
   - Worker-узлы (или группы узлов), на которых выполняются **NPU workloads**, должны использовать одну из следующих операционных систем (архитектура Arm):
     - `openEuler 22.03 LTS`
     - `Ubuntu 22.04`
   - Worker-узлы, на которых выполняются только **CPU workloads**, могут использовать любую операционную систему, так как оператор NPU не выполняет настройку узлов без NPU workloads.
5. **Поддерживаемое NPU-оборудование**
   - Узлы должны использовать поддерживаемые NPU:
     - `Ascend 910B`
     - `Ascend 310P`
   - Подробную информацию о совместимости ОС и оборудования см. в MindCluster Documentation ↗
6. **Должен быть установлен кластерный плагин `Alauda Build of Node Feature Discovery`.**

Оффлайн-установка

1. **Для оффлайн-установки требуются все предварительные условия онлайн-установки, а также дополнительные подготовительные шаги.**
2. **Подготовьте пакет драйвера и прошивки, а также пакет MindIO SDK.** Загрузите следующие пакеты (если вам не требуется установка MindIO, пакет MindIO загружать не нужно):
   - Для пакета драйвера и прошивки найдите файл `config.json` в репозитории GitCode npu-driver-installer ↗ и загрузите пакет по соответствующей ссылке, указанной для выбранной версии, модели NPU и архитектуры ОС соответствующего узла.

- Для пакета MindIO SDK найдите файл `config.json` в репозитории GitCode npu-node-provision ↗ и загрузите пакет SDK по соответствующей ссылке, указанной для модели NPU и архитектуры ОС соответствующего узла.

3. **Сохраните ZIP-файл пакета драйвера и прошивки в путь `/tmp/driver_pkg/` на узле, где будет выполняться оффлайн-установка.**
4. **Сохраните ZIP-файл пакета MindIO в путь `/opt/openFuyao/mindio/` на узле, где будет выполняться оффлайн-установка. (Если вам не требуется установка MindIO, пропустите этот шаг.)**
5. **Проверьте, установлены ли на целевом узле следующие инструменты.**

   - Для систем, использующих Yum в качестве менеджера пакетов, должен быть установлен следующий пакет: "jq wget unzip which net-tools pciutils gcc make kernel-devel-$(uname-r) kernel-headers-(uname-r) dkms".
   - Для систем, использующих apt-get в качестве менеджера пакетов, должен быть установлен следующий пакет: "jq wget unzip debianutils net-tools pciutils gcc make dkms linux-headers-$(uname -r)".
   - Для систем, использующих DNF в качестве менеджера пакетов, должен быть установлен следующий пакет: "jq wget unzip which net-tools pciutils gcc make kernel-devel-$(uname -r) kernel-headers-(uname-r) dkms".

Процедура

Загрузка Cluster plugin

> **INFO**
>
> Вы можете скачать приложения `Alauda Build of NPU Operator` и `Alauda Build of Node Feature Discovery` из Marketplace на веб-сайте Customer Portal.

Примечание: кластерный плагин Volcano пока можно не устанавливать.

Загрузка Cluster plugin

Платформа предоставляет инструмент командной строки `violet` для загрузки пакетов, скачанных из Marketplace на Custom Portal.

Подробности см. в разделе Upload Packages.

Установка Alauda Build of NPU Operator

1. Примените метку `masterselector=dls-master-node` ко всем master-узлам и метку `workerselector=dls-worker-node` ко всем worker-узлам.

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

2. Перейдите на страницу `Administrator` -> `Marketplace` -> `Cluster Plugin`, переключитесь на целевой кластер, а затем разверните кластерный плагин `Alauda Build of NPU Operator`.
   Описание параметров формы развертывания:

   > **WARNING**
   >
   > Если компоненты, перечисленные в таблице ниже, уже установлены, обязательно отключите соответствующие кнопки во время развертывания.

   > **TIP**
   >
   > Ascend Operator, NodeD, ClusterD, Resilience Controller, MindIO TFT и MindIO ACP по умолчанию не развертываются. Развертывайте их только при наличии явной необходимости.

Компонент	По умолчанию	Описание
Driver	Включено	Устанавливать ли driver и firmware.

Компонент	По умолчанию	Описание
Version	24.1.RC3	Версия driver и firmware. Необходимо выбрать номер версии в каталоге репозитория npu-driver-installer↗.
Ascend Device Plugin	Включено	Устанавливать ли Ascend Device Plugin.
Ascend Docker Runtime	Включено	Устанавливать ли Ascend Docker Runtime.
NPU exporter	Включено	Устанавливать ли NPU exporter.
Ascend Operator	Отключено	Устанавливать ли Ascend Operator.
NodeD	Отключено	Устанавливать ли NodeD.
ClusterD	Отключено	Устанавливать ли ClusterD. Сначала необходимо установить кластерный плагин Volcano.
Resilience Controller	Отключено	Устанавливать ли Resilience Controller.
MindIO TFT	Отключено	Устанавливать ли MindIO TFT.
MindIO ACP	Отключено	Устанавливать ли MindIO ACP.

Проверка

1. Сначала на странице кластерного плагина `Alauda Build of NPU Operator` вы можете увидеть статус `Installed`.
2. Дождитесь, пока pod npu-driver перейдет в состояние running. Оффлайн-установка занимает около 10 минут, тогда как онлайн-установка выполняется значительно быстрее.

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

3. Перезагрузите все NPU-узлы.
4. Выполните следующую команду на узле npu.

   ```
   npu-smi info
   ```

   Убедитесь, что отображение работает корректно.
5. Выполните следующую команду на master-узле.

   ```
   kubectl get npuclusterpolicy cluster
   ```

   Убедитесь, что статус `npuclusterpolicy` равен Ready.
6. Проверьте, доступны ли allocatable-ресурсы NPU на NPU-узле в control node бизнес-кластера. Выполните следующую команду:

   ```
   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. Запустите validation workload. Создайте spec-файл:

```
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
```

Примените spec:

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

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

Затем выполните следующую команду в контейнере:

```
npu-smi info
```

Убедитесь, что отображение работает корректно.

Установка Monitor

Если компонент NPU exporter был развернут при установке Alauda Build of NPU Operator, выполните следующие шаги для создания панели мониторинга.

1. Выполните команды на **control node** кластера.

```
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. Вы можете импортировать JSON-файл дашборда Grafana, следуя инструкции Import Dashboard, которая преобразует его в панель мониторинга для отображения. JSON-файл доступен в ascend-npu-dashboard↗.

NOTE

Теги в JSON-файле дашборда Grafana не могут содержать китайские символы и должны быть удалены вручную. Например:

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

После изменения:

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

FAQ

На что следует обратить внимание при удалении Alauda Build of NPU Operator?

Даже после удаления Alauda Build of NPU Operator драйвер может по-прежнему оставаться на хост-машине. На NPU-узле выполните следующую команду, чтобы удалить драйвер:

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

About Alauda Build of Hami

Heterogeneous AI Computing Virtualization Middleware (HAMi), ранее известный как k8s-vGPU-scheduler, представляет собой универсальный чарт, предназначенный для управления гетерогенными AI-вычислительными устройствами в кластере k8s. Он обеспечивает возможность совместного использования гетерогенных AI-устройств между задачами.

Note

Поскольку выпуски Alauda Build of Hami осуществляются в ином режиме, чем у Alauda Container Platform, документация Alauda Build of Hami теперь доступна в виде отдельного набора по адресу Alauda Build of Hami ↗。

О плагине устройства NVIDIA GPU сборки Alauda

Плагин устройства NVIDIA для Kubernetes — это Daemonset, который позволяет вам автоматически:

- Отображать количество GPU на каждом узле вашего кластера
- Отслеживать состояние здоровья ваших GPU
- Запускать контейнеры с поддержкой GPU в вашем Kubernetes кластере.

Note

Поскольку выпуски Alauda Build of NVIDIA GPU Device Plugin осуществляются в ином режиме, чем у Alauda Container Platform, документация Alauda Build of NVIDIA GPU Device Plugin теперь доступна в виде отдельного набора по адресу Alauda Build of NVIDIA GPU Device Plugin ↗。