NetApp AIPod Mini for ERAG - 部署步驟

NetApp artificial intelligence solutions

NetApp
February 12, 2026

目錄

NetApp AIPod Mini for ERAG - 部署步驟

本文提供了一份全面的逐步指南，指導您如何部署 NetApp AIPod Mini for Enterprise RAG（ERAG）2.0。它涵蓋了所有核心元件的端對端安裝和配置，包括 Kubernetes 平台、NetApp Trident 用於儲存編排，以及使用 ansible playbooks 建立的 ERAG 2.0 堆疊。除了部署工作流程之外，本文檔還包含一份專門的故障排除指南，其中記錄了安裝過程中遇到的常見問題、其根本原因以及建議的解決方案，以確保流暢可靠的部署體驗。

Sathish Thyagarajan、Michael Oglesby、Arpita Mahajan NetApp

假設：

- 部署使用者擁有足夠的權限來建立命名空間和安裝 Helm 圖表。
- Xeon 伺服器執行 Ubuntu 22.04。
- 所有 Xeon 伺服器都設定了相同的使用者名稱。
- DNS 管理存取權限可用。
- ONTAP 9.16 已部署，並配置了用於 S3 存取的 SVM。
- S3 儲存桶已建立並配置完成。

先決條件

安裝 Git、Python3.11 和 pip for Python3.11

在 Ubuntu 22.04 上：

```
add-apt-repository ppa:deadsnakes/ppa
apt update
apt upgrade
apt install python3.11
python3.11 --version
apt install python3.11-pip
python3.11 -m pip --version
```

ERAG 2.0/2.0.1 部署步驟

1.從 GitHub 拉取 Enterprise RAG 2.0 版本

```
git clone https://github.com/opea-project/Enterprise-RAG.git
cd Enterprise-RAG/
git checkout tags/release-2.0.0
```

對於 ERAG 2.0.1 、請使用下列命令

```
git checkout tags/release-2.0.1
```

2.安裝必備組件

```
cd deployment/
sudo apt-get install python3.11-venv
python3 -m venv erag-venv
source erag-venv/bin/activate
pip install --upgrade pip
pip install -r requirements.txt
ansible-galaxy collection install -r requirements.yaml --upgrade
```

3.建立 inventory 檔案

```
cp -a inventory/sample inventory/<cluster-name>
vi inventory/<cluster-name>/inventory.ini
# Control plane nodes
kube-3 ansible_host=<control_node_ip_address>

# Worker nodes
kube-1 ansible_host=<worker_node1_ip_address>
kube-2 ansible_host=<worker_node2_ip_address>

# Define node groups
[kube_control_plane]
kube-1
kube-2
kube-3

[kube_node]
kube-1
kube-2

[etcd:children]
kube_control_plane

[k8s_cluster:children]
kube_control_plane
kube_node

# Vars
[k8s_cluster:vars]
ansible_become=true
ansible_user=<ssh_username>
ansible_connection=ssh
```

4.為每個節點設定免密碼 SSH

```
ssh-copy-id REMOTE_USER@MACHINE_IP
```

注意：如果使用部署節點部署 ERAG ，請確保在部署節點上也配置了無密碼 SSH 。

5.驗證連線能力

```
ansible all -i inventory/<cluster-name>/inventory.ini -m ping
```

注意：如果您的節點上未配置免密碼 sudo，則需要在此命令中新增 --ask-become-pass。使用 --ask-become

-pass 時，請務必確保每個節點上的 ssh 使用者密碼相同。

6.編輯 `config.yaml` 檔案

編輯 `inventory/<cluster-name>/config.yaml` 以反映您的環境具體情況，準備部署。

```
vi inventory/<cluster-name>/config.yaml
```

範例片段：

```
…
deploy_k8s: true
…
install_csi: "netapp-trident"
…
local_registry: false
…
trident_operator_version: "2510.0"    # Trident operator version (becomes
100.2506.0 in Helm chart)
trident_namespace: "trident"          # Kubernetes namespace for Trident
trident_storage_class: "netapp-trident" # StorageClass name for Trident
trident_backend_name: "ontap-nas"     # Backend configuration name
…
ontap_management_lif: "<ontap_mgmt_lif>"              # ONTAP management
LIF IP address
ontap_data_lif: "<ontap_nfs_data_lif>"                   # ONTAP data LIF
IP address
ontap_svm: "<ontap_svm>"                          # Storage Virtual Machine
(SVM) name
ontap_username: "<ontap_username>"                    # ONTAP username
with admin privileges
ontap_password: "<redacted>"                     # ONTAP password
ontap_aggregate: "<ontap_aggr>"                    # ONTAP aggregate name
for volume creation
…
kubeconfig: "<repository path>/deployment/inventory/<cluster-
name>/artifacts/admin.conf"
…
```

7.部署 K8s 叢集（使用 Trident ）

使用 configure 和 install 標籤執行 ansible-playbook playbooks/infrastructure.yaml 來部署叢集和 Trident CSI。

```
ansible-playbook playbooks/infrastructure.yaml --tags configure,install -i
inventory/<cluster-name>/inventory.ini -e @inventory/<cluster-
name>/config.yaml
```

注意：- 如果您的節點上未設定免密碼 sudo，則需要將 `--ask-become-pass`新增至此命令。使用 `--ask-become-pass`時，請務必確保每個節點上的 ssh 使用者密碼相同。- 有關詳細資訊，請參閱 "NetApp Trident CSI 整合，適用於企業 RAG"。有關更多詳細資訊，請參閱 "Trident安裝文檔"。

8.變更 iwatch 開啟描述元的數量

詳情請參閱 "iwatch 開放式描述符"。

9.安裝 kubectl

如果尚未安裝，請參閱 "安裝 Kubectl"。從 `<repository path>/deployment/inventory/<cluster-name>/artifacts/admin.conf`擷取 kubeconfig 檔案。

10.在 Kubernetes 叢集中安裝 MetalLB

在 Kubernetes 叢集上使用 Helm 安裝 MetalLB 。

```
helm repo add metallb https://metallb.github.io/metallb
helm -n metallb-system install metallb metallb/metallb --create-namespace
```

詳情請參閱 "MetalLB 安裝"。

11.設定 MetalLB

MetalLB 已設定為第 2 層模式，並根據文件化的組態準則建立了所需的 IPAddressPool 和 L2Advertisement 資源。

```
vi metallb-ipaddrpool-l2adv.yaml
kubectl apply -f metallb-ipaddrpool-l2adv.yaml
```

範例片段：

```
vi metallb-ipaddrpool-l2adv.yaml
---
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: erag
  namespace: metallb-system
spec:
  addresses:
  - <IPAddressPool>
---
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: metallb-l2adv
  namespace: metallb-system
```

注意：- 使用 `metallb-system` 作為 MetalLB IPAddressPool 和 L2Advertisement 的命名空間。- IP 位址池可以包含與 Kubernetes 節點位於相同子網路內的任何未使用的 IP 位址。ERAG 只需要一個 IP 位址。- 詳情請參閱 "MetalLB Layer2 配置"。

12.使用 FQDN 、磁碟區存取模式、入口和 S3 詳細資料更新 config.yaml 。

修改位於 `inventory/<cluster-name>/config.yaml` 的 config.yaml 檔案，以定義部署 FQDN、設定磁碟區存取模式、設定 ingress 曝光並整合 ONTAP S3。

編輯 `config.yaml` 並套用以下配置變更：

- FQDN：指定用於存取部署的完整網域名稱。
- 磁碟區存取模式：在 gmc.pvc 部分下，設定 `accessMode: ReadWriteMany` 以支援跨多個 pod 對模型磁碟區的並行存取。
- Ingress 設定：將 ingress service_type 設定為 LoadBalancer 以允許外部存取應用程式。
- S3 儲存詳情：設定 storageType 為 s3compatible 並配置 ONTAP S3 參數，包括區域、存取憑證、內部和外部端點。
- SSL 憑證驗證：僅當 ONTAP S3 配置為使用自簽名憑證時，才將 edpInternalCertVerify 和 edpExternalCertVerify 設為 false。如果憑證由受公共信任的 CA 頒發，則應保持啟用這些參數。

範例片段：

```
vi inventory/<cluster-name>/config.yaml
…
FQDN: "<FQDN>" # Provide the FQDN for the deployment
…
gmc:
  enabled: true
  pvc:
    accessMode: ReadWriteMany # AccessMode
    models:
      modelLlm:
        name: model-volume-llm
        storage: 100Gi
      modelEmbedding:
        name: model-volume-embedding
        storage: 20Gi
      modelReranker:
        name: model-volume-reranker
        storage: 10Gi
…
ingress:
  …
  service_type: LoadBalancer
  …
edp:
  …
  storageType: s3compatible
  …
  s3compatible:
    region: "us-east-1"
    accessKeyId: "<your_access_key>"
    secretAccessKey: "<your_secret_key>"
    internalUrl: "https://<IP-address>"
    externalUrl: "https://<IP-address>"
    bucketNameRegexFilter: ".*"
    edpExternalCertVerify: false
    edpInternalCertVerify: false
  …
```

注意：- 預設情況下，Intel® AI for Enterprise RAG 應用程式會從 SVM 中的所有現有儲存桶中擷取資料。如果您的 SVM 中有多個儲存桶，您可以修改 `bucketNameRegexFilter`欄位，以便僅從特定儲存桶中提取資料。- 有關詳細資訊，請參閱 "Intel® AI for Enterprise RAG 部署"文件。

13.設定排程同步設定

安裝適用於 Intel® AI for Enterprise RAG 的 OPEA 應用程式時，請啟用 `scheduledSync`，以便應用程式自動從您的 S3 儲存桶中擷取新的或更新的檔案。

什麼時候 `scheduledSync`啟用後，應用程式會自動檢查來源 S3 儲存桶中是否有新檔案或更新的檔案。在此同步過程中發現的任何新檔案或更新檔案都會自動提取並新增至 RAG 知識庫。應用程式根據預設的時間間隔檢查您的來源儲存桶。預設時間間隔為 60 秒，這表示應用程式每 60 秒檢查一次變更。您可能希望更改此間隔以滿足您的特定需求。

若要啟用 `scheduledSync`並設定同步間隔，請在 `deployment/components/edp/values.yaml`中設定以下值：

```
vi components/edp/values.yaml
…
presignedUrlCredentialsSystemFallback: "true"
…
celery:
  …
  config:
    …
    scheduledSync:
      enabled: true
      syncPeriodSeconds: "60"
…
```

14.部署 Enterprise RAG 2.0/2.0.1

安裝前，請依照 "Intel® AI 企業級 RAG 應用程式部署指南" 中概述的步驟驗證基礎架構是否已準備好。此步驟可確保底層基礎架構配置正確，並滿足 Enterprise RAG Application 成功安裝所需的所有先決條件。

使用以下命令執行安裝：

```
ansible-playbook -u $USER playbooks/application.yaml --tags
configure,install -e @inventory/<cluster-name>/config.yaml
```

注意：如果您的部署節點（執行 ansible-playbook 指令的筆記型電腦或跳板機）上未設定免密碼 sudo，則需要將 `--ask-become-pass`新增至此指令。使用 `--ask-become-pass`時，請務必確保每個節點上的 ssh 使用者密碼相同。

15.建立 DNS 項目

在您的 DNS 伺服器中為 Enterprise RAG Web 控制面板建立 DNS 項目。若要繼續、請擷取指派給 Enterprise RAG 入口 LoadBalancer 的外部 IP 位址：

```
kubectl -n ingress-nginx get svc ingress-nginx-controller
```

為步驟 12 中使用的 FQDN 建立指向此 IP 位址的 DNS 項目。

注意：- DNS 項目使用的 FQDN 必須與組態檔中的 FQDN 相符。

16.存取 Enterprise RAG UI

透過在瀏覽器中導覽至該 FQDN 來存取 Enterprise RAG UI。注意：您可以從 cat ansible-logs/default_credentials.txt 擷取預設 UI 憑證

故障排除指南

1.問題：Keycloak Helm 安裝衝突

場景：部署 ERAG 時，Keycloak 安裝可能會失敗，並出現以下錯誤：

```
FAILED - RETRYING: [localhost]: Install Keycloak Helm chart (5 retries
left).
Failure when executing Helm command. Exited 1.
    stdout:
    stderr: Error: UPGRADE FAILED: another operation
(install/upgrade/rollback) is in progress
```

操作：如果重試後仍然失敗 - 卸載 ERAG 部署，使用以下命令刪除現有的 auth 命名空間，然後重新執行部署。

```
ansible-playbook playbooks/application.yaml --tags uninstall -e
@inventory/<cluster-name>/config.yaml

helm -n auth uninstall keycloak
kubectl -n auth get pvc # confirm all PVCs are gone; if any are left,
delete them
kubectl delete ns auth
```

注意：過時的 Helm 版本狀態可能會阻止後續的安裝或升級作業。

2.問題：找不到 Trident Operator Helm Chart 版本

場景：在 ERAG 部署過程中，由於 Helm Chart 版本不匹配，Trident Operator 安裝可能會失敗。可能會出現以下錯誤：

```
TASK [netapp_trident_csi_setup : Install Trident operator via Helm]
fatal: [localhost]: FAILED! => changed=false
  command: /usr/local/bin/helm --version=100.2510.0 show chart 'netapp-
trident/trident-operator'
  msg: |-
    Failure when executing Helm command. Exited 1.
    stdout:
    stderr: Error: chart "trident-operator" matching 100.2510.0 not found
in netapp-trident index.
            (try 'helm repo update'): no chart version found for trident-
operator-100.2510.0
```

操作：如果發生此錯誤，請更新 Helm 儲存庫索引並重新執行部署 playbook。

```
helm repo update
ansible-playbook playbooks/application.yaml -e @inventory/<cluster-
name>/config.yaml
```

注意：這是 ERAG 2.0 版本中的已知問題。修復程式已提交，並將包含在未來版本中。

版權資訊

商標資訊