C140-0133-01Z3

Fsas Technologies

生成 AI を実現する PRIMERGY/ETERNUS の 最新テクノロジー

AI ストレージ リファレンス・ アーキテクチャー

はじめに

近年、人工知能（AI）は急速な発展を遂げ、様々な分野で革新的な技術として注目されています。特に、機械学習や深層学習は、膨大なデータから複雑なパターンを学習し、予測や分類、生成など、人間の能力を超えるタスクを可能にする技術として期待されています。

本書は、オンプレミスによる AI 運用環境におけるストレージの重要性と、ETERNUS AX/AC series を活用したリファレンス・アーキテクチャーを紹介します。

本書を通して、ETERNUS AX/AC series が AI 運用におけるストレージの課題解決に貢献できることをご理解いただければ幸いです。

なお、本書でご紹介しているソフトウェアは、弊社からサポートを提供していないものもあります。具体的なサポートについては、弊社営業担当にご相談ください。

初版
2024 年 9 月

対象者

このドキュメントは、以下の読者を対象としています。

- AI モデルとソフトウェアの開発ソリューションを設計するエンタープライズアーキテクト
- 深層学習（DL）および機械学習（ML）の開発目標を達成するための効率的な方法を探しているデータサイエンティストおよびデータエンジニア
- IT の意思決定者、および AI イニシアチブによる最速の市場投入に関心のあるビジネスリーダー

登録商標

本製品に関連する他社商標については、以下のサイトを参照してください。
https://www.fujitsu.com/jp/products/computing/storage/trademark/
https://www.fujitsu.com/jp/products/computing/servers/brand.html

本書では、本文中の ™、® などの記号は省略しています。

目次

1 生成 AI の概要 5
1.1 生成 AI チャットボットの利点 5
1.2 生成 AI チャットボットのユースケース 5
1.3 RAG のセキュリティー上の課題 6
2 リファレンス・アーキテクチャー 7
2.1 全体アーキテクチャー 7
2.2 チャットボットアーキテクチャー 8
2.3 ハードウェア構成 11
2.3.1 ストレージ構成 11
2.3.2 ホストサーバー 13
2.4 ソフトウェア構成 15
3 検証に向けた構築手順 16
3.1 ハードウェアセットアップ（ストレージ／スイッチ／サーバー） 17
3.1.1 ストレージ 17
3.1.2 スイッチ 17
3.1.3 サーバー 18
3.2 GPU Driver のインストール／セットアップ 18
3.3 Kubernetes のインストール／セットアップ 18
3.3.1 NVIDIA GPU Operator の設定 18
3.3.2 NVIDIA Network Operator のインストール 19
3.3.3 NetApp Astra Trident のインストール 19
3.4 StorageClass の設定 19
3.4.1 Trident Backend Storage の設定 19
3.4.2 StorageClass の設定 19
3.5 Open Source MLOps with NetApp の構築 20
3.5.1 Kubeflow のインストール 20
3.5.2 DataOps Toolkit 20
3.6 Pod の設定 20
3.7 チャットボット FastChat の設定 21
3.8 RAG LangChain 設定 21
3.9 LLMPerf のインストール 21

3.10 【参考】GPUDirect Storage と NFS over RDMA 22

4 検証手順 24

4.1 検証構成 24

4.2 検証 24

4.2.1 LLMPerf の実行 24

4.2.2 LLMPerf の実施結果の確認 24

5 評価結果 25

5.1 性能評価 25

5.2 機能評価 26

6 まとめ 29

付録 A 関連資料 30

1　生成 AI の概要

生成 AI は、テキスト、画像、音声、動画、コードなど、人間が作成したかのような新しいコンテンツを生成する人工知能です。近年、深層学習技術の発展により、驚くべき精度と創造性をもち、様々な分野で活用され始めています。
代表的な生成 AI には、テキスト生成の ChatGPT、画像生成の DALL-E 2、音声生成の Jukebox などがあります。これらの AI は、膨大なデータセットから学習し、独自のコンテンツを生み出すことができます。
生成 AI は、コンテンツ作成、デザイン、翻訳、コード開発など、様々な分野で効率性向上や創造性促進に貢献します。例えば、記事や広告の自動生成、ロゴやイラストのデザイン、プログラミングコードの自動作成などが可能になります。

本リファレンス・アーキテクチャーでは、生成 AI を用いたチャットボットに焦点を当て、その構成要素と機能について解説します。

1.1　生成 AI チャットボットの利点

生成 AI チャットボットは、従来のチャットボットに比べて、より自然で人間らしい会話体験を提供します。大規模言語モデル（LLM）を活用することで、文脈を理解し、柔軟な応答を生成し、創造的な表現も可能です。

さらに、生成 AI チャットボットは、24 時間 365 日対応、迅速な対応、人件費削減など、効率性とコスト削減にも貢献します。ユーザーの過去の会話履歴や好みを学習することで、パーソナライズされた回答を提供することも可能です。
生成 AI チャットボットは、顧客サポート、情報提供、エンターテイメント、教育など、様々な分野で活用され、ビジネスの効率化や顧客満足度の向上に貢献することが期待されています。

1.2　生成 AI チャットボットのユースケース

生成 AI チャットボットは、その高度な言語処理能力と柔軟性から、様々な分野で活用され始めています。主なユースケースは以下のとおりです。

- 顧客サポート
- 情報提供
- エンターテイメント
- 教育
- ビジネスプロセス自動化
- その他

1.3　RAG のセキュリティー上の課題

RAG（Retrieval-Augmented Generation）は、大規模言語モデル（LLM）に外部の知識ベースを連携させる技術です。従来の LLM は学習データに含まれる情報しか扱えませんでした。RAG では、データベースやウェブサイトなどの外部知識ベースから必要な情報を取得し、LLM に提供することで、より正確で最新の情報に基づいた回答を生成できます。チャットボットや検索エンジンなど、様々な分野で活用され、より高度な情報生成を実現します。

しかし、RAG には以下のようなセキュリティー上の課題もあります。

■ データ漏洩

RAG は外部データソースから情報を取得するため、秘密情報を含むデータソースにアクセスした場合、データ漏洩のリスクがあります。

■ 悪意のあるデータの注入

RAG は外部データソースから情報を取得するため、悪意のあるデータが注入される可能性があります。これにより、チャットボットが誤った情報や有害な情報を提供する可能性があります。

■ プライバシー侵害

RAG はユーザーの会話履歴や個人情報を収集することがあります。この情報は、プライバシー侵害や不正利用に繋がる可能性があります。

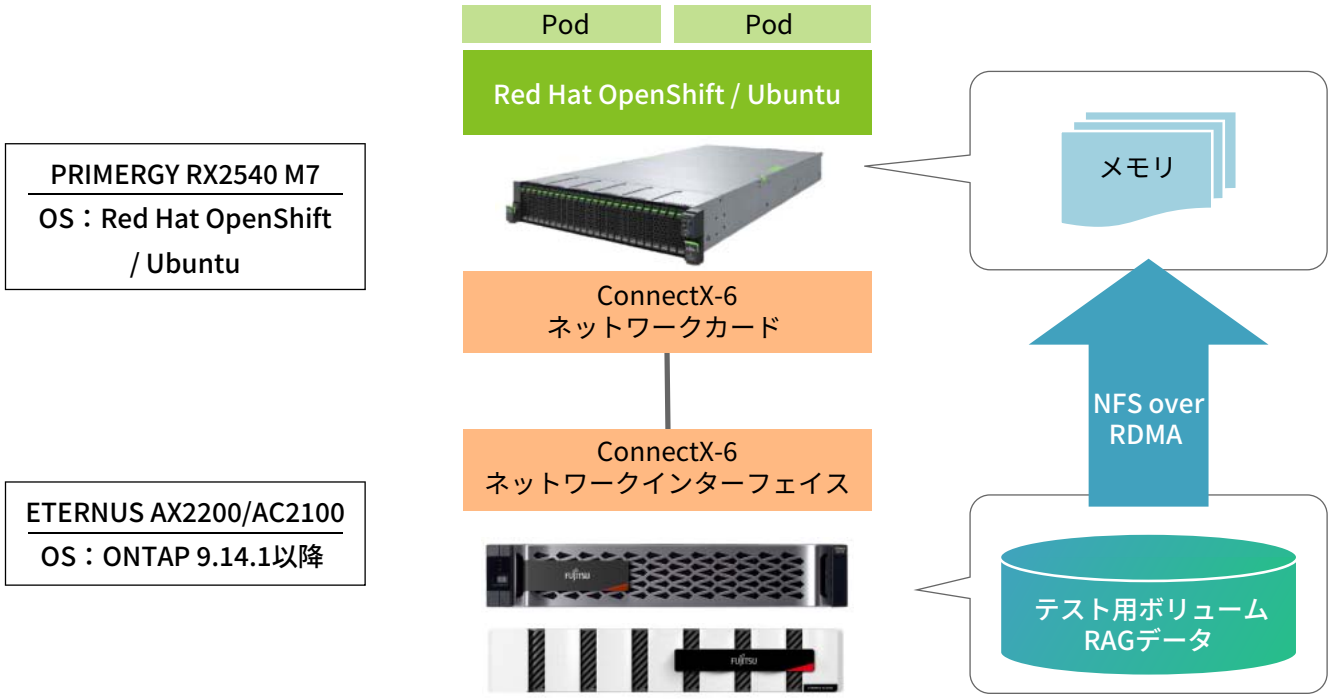
■ モデルの改ざん

RAG は外部データソースから情報を取得するため、モデル自体が改ざんされる可能性があります。これにより、チャットボットが意図しない動作をする可能性があります。

これらの課題に対処するためには、信頼できるデータソースの使用、データ暗号化、入力データの検証、モデル監査、プライバシー保護、セキュリティー対策の導入などが重要です。
RAG は、生成 AI チャットボットの機能向上に役立つ一方、セキュリティー対策を怠ると大きなリスクがあることを認識しておく必要があります。

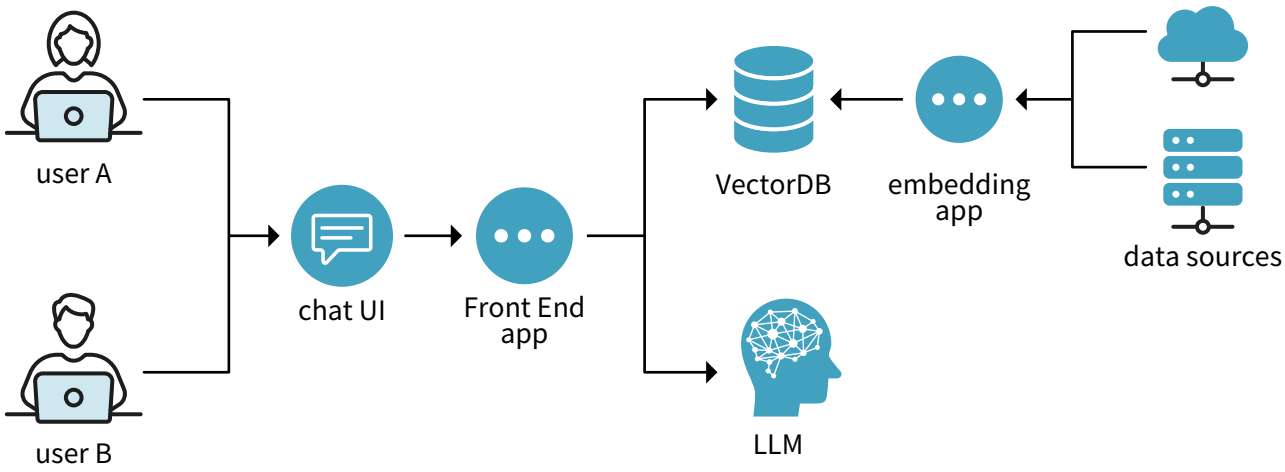
2　リファレンス・アーキテクチャー

2.1　全体アーキテクチャー



2.2　チャットボットアーキテクチャー

■ 基本アーキテクチャー



(1)　chatbot UI が user の質問を受け取り Front End (RAG) app に渡す

(2)　Front End (RAG) app が質問を含むプロンプトを embedding して VectorDB に渡す

(3)　VectorDB は data source を embedding した index からプロンプトと類似したもの応答する

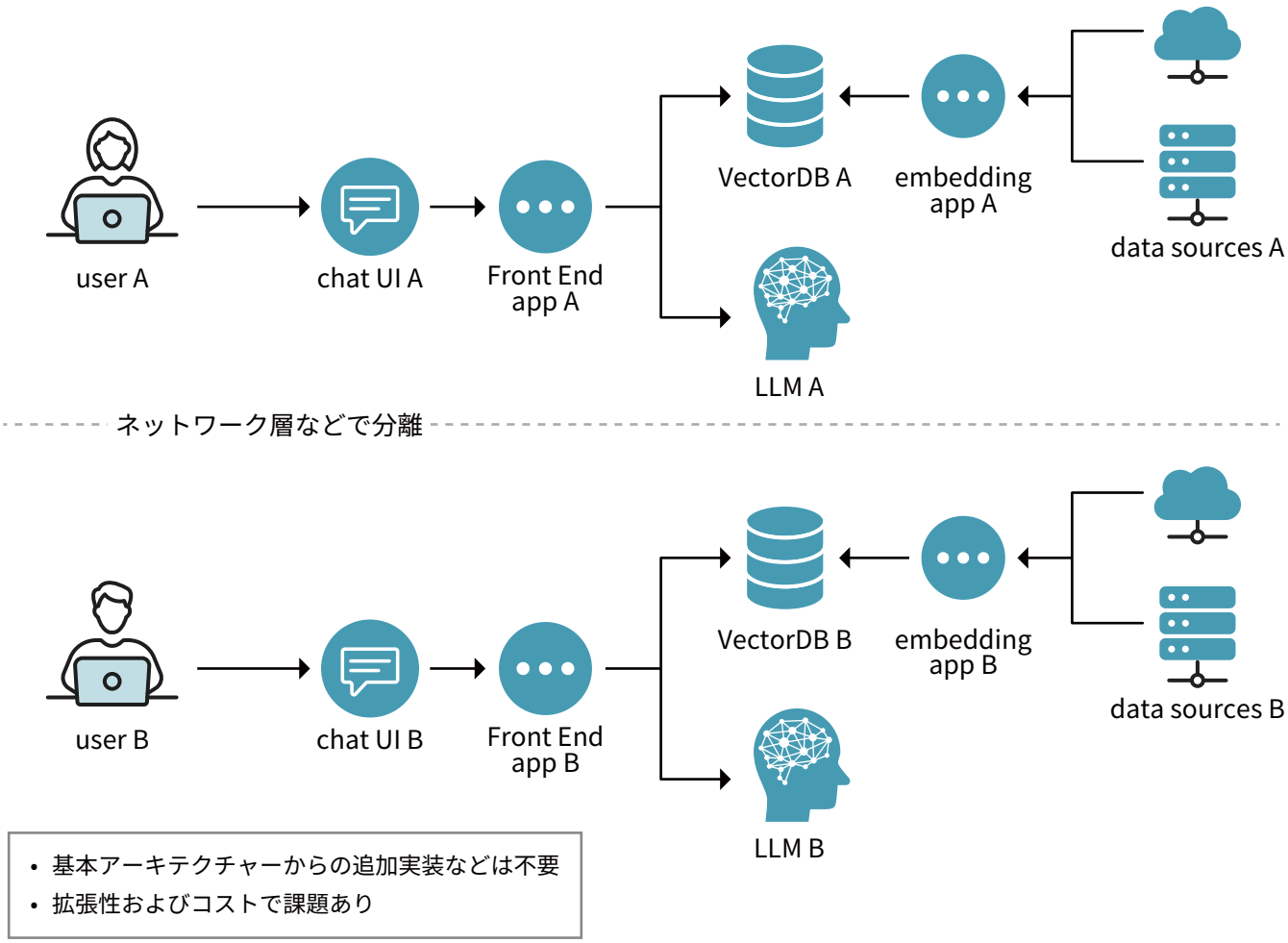
(4)　Front End (RAG) app が類似した index とプロンプトを LLM に渡す

(5)　LLM が index とプロンプトから応答を作成する

(6)　chatbot UI が user に応答を送信する

先に挙げた情報漏洩の対策として、以下のような対策が考えられます。

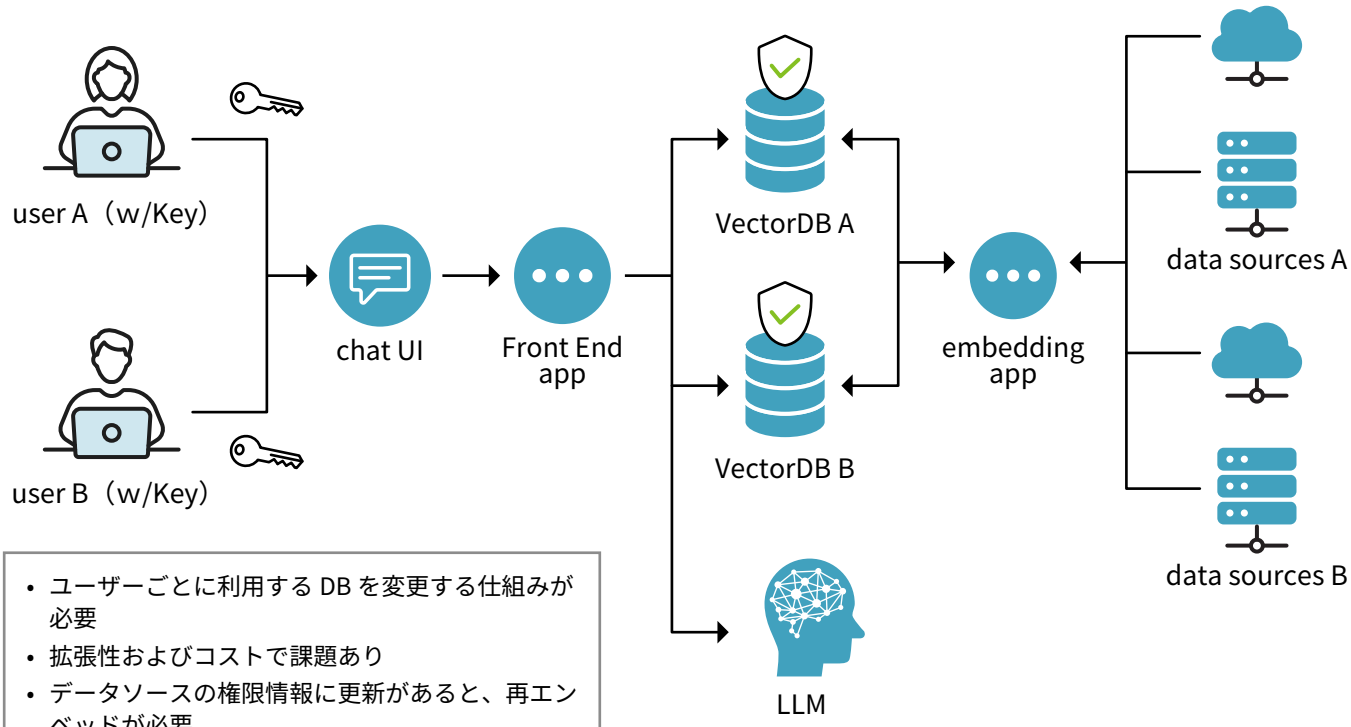
■ 完全分離アーキテクチャー



- 基本アーキテクチャーからの追加実装などは不要
- 拡張性およびコストで課題あり

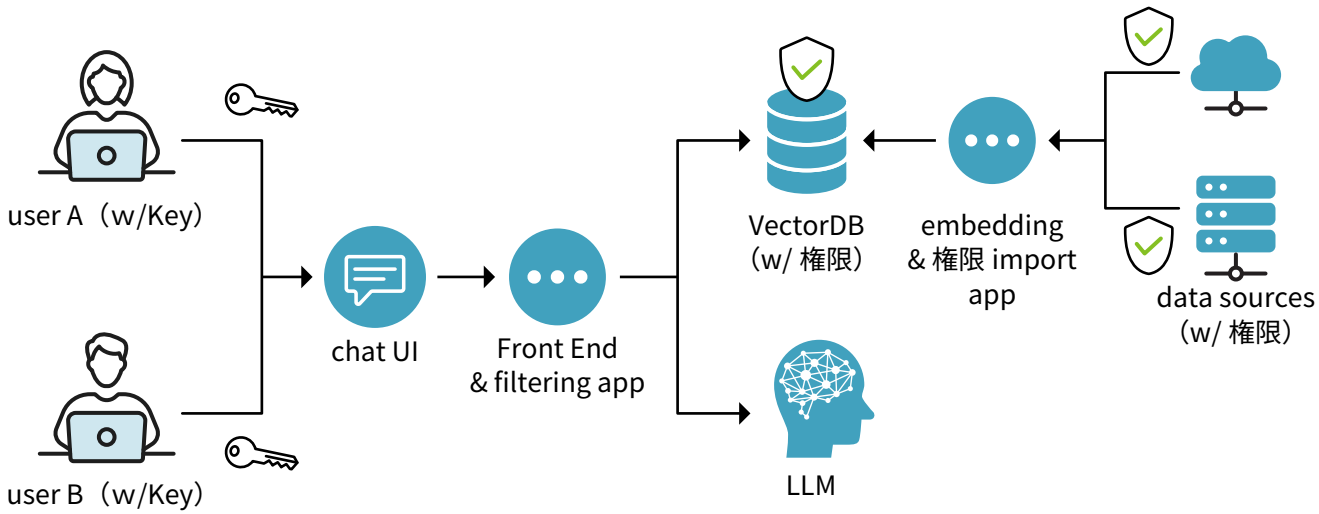
完全分離アーキテクチャーでは拡張性コストで課題があり、現在は「DB 分離アーキテクチャー」や「DB フィルタリングアーキテクチャー」などが考えられます。

DB 分離アーキテクチャー



- ユーザーごとに利用する DB を変更する仕組みが必要
- 拡張性およびコストで課題あり
- データソースの権限情報に更新があると、再エンベッドが必要

DB フィルタリングアーキテクチャー



- ユーザーごとに利用する DB のフィルタリングを行う仕組みが必要
- データソースの権限情報に更新があると、再エンベッドが必要

2.3　ハードウェア構成

2.3.1　ストレージ構成

機種	ETERNUS AX2200、ETERNUS AC2100
OS	ONTAP 9.14.1 以降（最新版を推奨）
インターフェイス	アダプターカード（25Gbit/s、Ethernet、4 ポート、AC2100、Networking 用） SFP モジュール（Optical、25Gbit/s）

■ ETERNUS AX/AC series

ETERNUS AX/AC series は、ファイル／ブロックアクセスの双方に対応する多機能なユニファイドストレージです。長年の実績をもつ、堅牢なストレージ専用 OS である「ONTAP」を搭載。業界標準といえるあらゆるプロトコルに対応しており用途を選びません。スケールアウト、優れたデータ保護や管理機能等を実現した万能オールフラッシュストレージです。

ETERNUS AX2200 （図 2.1）および ETERNUS AC2100（図 2.2）は、エントリークラスでエンド・ツー・エンドの NVMe をサポートしています。ETERNUS AX2200 は 、エントリーレベルでありながらワンランク上の性能を有したモデルです。ETERNUS AC2100 は、QLC SSD を採用し、性能と価格のバランスが取れた大容量型オールフラッシュストレージです。

図 2.1　ETERNUS AX2200

ETERNUS AX2200 の詳細はこちら

図 2.2　ETERNUS AC2100

ETERNUS AC2100 の詳細はこちら

ETERNUS AX/AC series で採用している OS の ONTAP は、可用性およびセキュリティーに優れています。また、オールフラッシュストレージで、高性能なストレージです。

● 可用性

データのミラーリング、レプリケーション、Snapshot、クローンなどの機能により、障害発生時でもデータへのアクセスを継続的に維持できます。さらに、ONTAP は自動化された障害回復機能を提供し、ダウンタイムを最小限に抑え、ビジネスの継続性を確保します。

● セキュリティー

ONTAP は、様々なセキュリティー機能を備えています。これらの機能を活用することで、RAG におけるセキュリティー問題を効果的に軽減できます。

ONTAP を活用した RAG セキュリティー対策のメリット

- **データ漏洩のリスク軽減**
  データ暗号化、アクセス制御、バックアップ機能により、データ漏洩のリスクを軽減します。
- **悪意のあるデータの注入防止**
  データ完全性チェック、自律型ランサムウェア対策機能により、悪意のあるデータの注入を防ぎます。
- **プライバシー保護**
  データ匿名化、プライバシー保護ポリシーにより、ユーザーのプライバシーを保護します。
- **モデルの改ざん検出**
  セキュリティー監査機能により、モデルの改ざんを検出します。

セキュリティー問題	ONTAP の対応策	詳細
データ漏洩	データ暗号化	FIPS レベルの暗号化に対応
	アクセス制御	ONTAP は、多層的なアクセス制御でデータの機密性を守ります。ACL、暗号化、監査ログにより、セキュリティー強化、運用効率化、およびコンプライアンス対応を実現します。
	バックアップと復旧	Snapshot、SnapMirror、および SnapRestore など、性能を犠牲にしないバックアップに対応します。
悪意のあるデータの注入	データ完全性チェック	ONTAP は、データの完全性をチェックする機能を提供し、データが改ざんされていないことを確認します。
	マルウェア対策	ONTAP は、Vscan 機能でサードパーティー製ウイルス対策ソフトウェアをサポートし、マルウェア感染によるデータ改ざんを防ぎます。
	自律型ランサムウェア対策	ONTAP の自律型ランサムウェア対策機能は、リアルタイム監視、攻撃阻止、データ復元、および AI 学習で、ランサムウェア攻撃からデータを保護します。

セキュリティー問題	ONTAP の対応策	詳細
プライバシー侵害	データ匿名化	ONTAP は、直接的なデータ匿名化機能は提供しませんが、暗号化、データマスク、クローン、分離、およびアクセス制御などを組み合わせることで、データ匿名化を実現できます。
	プライバシー保護ポリシー	ONTAP は、プライバシー保護ポリシーを設定し、データへのアクセスを制限することができます。これにより、ユーザーのプライバシーを保護することができます。
モデルの改ざん	セキュリティー監査	ONTAP では、クラスタで実行された管理アクティビティーについて、発行された要求、要求を発行したユーザー、ユーザーのアクセス方法、および要求が発行された時間などの情報が記録されます。

ONTAP は、RAG におけるセキュリティー問題を解決するための様々な機能を提供します。これらの機能を活用することで、データ漏洩、悪意のあるデータの注入、プライバシー侵害、モデルの改ざんなどのリスクを軽減することができます。

- 性能

エンド・ツー・エンド NVMe は、従来のストレージプロトコルと比べて、ストレージデバイスとホスト間の通信を高速化します。NVMe は、低レイテンシと高スループットを実現し、データ転送を効率化します。これにより、アプリケーションのパフォーマンスが向上し、データベースや仮想化などのワークロードで顕著な性能改善が見られます。特に、大量のデータ処理やリアルタイム処理が必要なアプリケーションに適しています。

2.3.2 ホストサーバー

サーバー	PRIMERGY RX2540 M7
OS	Red Hat OpenShift または Ubuntu
CPU	Intel Xeon Gold 6430（2 プロセッサー）
GPU	NVIDIA L4（2GPU）
メモリ容量	256GB
ネットワークカード	ポート拡張オプション（相当品：NVIDIA (Mellanox) MCX631432AN-ADAB OCPv3）（*1） 25GBASE-SR SFP28（25GBASE-SR SFP28）

*1: NFS over RDMA を有効にするため、ConnectX-5 以上のインターフェイスおよびネットワークカードが必要です。

■ PRIMERGY

図 2.3　PRIMERGY RX2540 M7

充実した機能により高性能・高信頼とお客様システムのさらなる高速化を実現しました。お客様のデジタルイノベーションを支える幅広い用途に最適な 2WAY ラック型サーバーで、インテル社の第 4 世代 Intel Xeon スケーラブル・プロセッサーに加え、最新の第 5 世代 Intel Xeon スケーラブル・プロセッサーをラインナップ、最大 64 コアの CPU を 2 個まで搭載可能です。PCI Express Gen5 にも対応し、お客様は多くのラインナップから用途に応じてオプション選択いただけます。仮想化基盤をはじめ、昨今のテレワークを実現する仮想デスクトップシステム、高性能が求められるデータベースや AI システムなど、お客様のデジタルイノベーションを支える幅広い用途に最適です。

図 2.4　GPU

本カードは、Single-slot の PCI Express Gen4 を採用しています。GPU メモリを 24GB 搭載しており、最大 300GB/s の PCIe カードメモリ帯域を提供します。

- PCI Express Gen4 を採用
- Ada Lovelace GPU と 24GB の GPU メモリを搭載
- サーバーでの 24 時間稼働を考慮した設計

2.4　ソフトウェア構成

ソフトウェアは、以下の 2 つの構成を推奨します。

- 構成①（有償サポートを受けて保証のある構成）

種別	名称	説明
コンテナプラットフォーム	Red Hat OpenShift	OS とコンテナ基盤を含む有償サポート
LLM 基盤サービス	LLM、RAG	RAG と LLM を含むソフトウェアの構築サービス

- 構成②

種別	名称	説明
OS	Ubuntu	オープンソースの無償 OS
コンテナソフトウェア	Kubernetes	オープンソースのコンテナ構築サービス OS は Ubuntu と RHEL を選択可能
LLM 基盤サービス	LLM、RAG	RAG と LLM を含むソフトウェアの構築サービス

3　検証に向けた構築手順

検証に向けて実施した構築手順は、以下のとおりです。

「3.1 ハードウェアセットアップ（ストレージ／スイッチ／サーバー）」(P.17)
▼
「3.2 GPU Driver のインストール／セットアップ」(P.18)
▼
「3.3 Kubernetes のインストール／セットアップ」(P.18)
▼
「3.4 StorageClass の設定」(P.19)
▼
「3.5 Open Source MLOps with NetApp の構築」(P.20)
▼
「3.6 Pod の設定」(P.20)
▼
「3.7 チャットボット FastChat の設定」(P.21)
▼
「3.8 RAG LangChain 設定」(P.21)
▼
「3.9 LLMPerf のインストール」(P.21)

3.1　ハードウェアセットアップ（ストレージ／スイッチ／サーバー）

3.1.1　ストレージ

備考

ストレージの設定については、以下を参照してください。

- ETERNUS AX/AC series マニュアル
  https://www.fujitsu.com/jp/products/computing/storage/manual/axhx-list.html
- ONTAP
  https://storage-system.fujitsu.com/manual/ja/axhx/introduction-concepts/index.html

■ ネットワークインターフェイス

NFS over RDMA を有効にするため、ストレージがサポートしている ConnectX-5 以上の RoCE 対応のアダプターカードが必要です。
データ用ネットワークポートの MTU は 9000 が奨励値です。値が異なる場合の設定方法は、ONTAP のマニュアルを参照してください。

■ NFS サーバーの作成

NFS サーバーを作成します。

注意

通常の NAS 接続では、同一ノードのポートをまとめてリンクアグリゲーショングループを作成して仮想インターフェイス（a0a など）とし、これを LIF に割り当ててアクセスポートとして使用することが可能ですが、仮想インターフェイスに対しては RoCE（RDMA）を有効化できません。
このため、RoCE（RDMA）を有効化したい場合は、物理ポート（e2a など）を LIF に割り当てる必要があります。

備考

Kubernetes クラスタとの連携を行う場合は、SVM の管理者、管理ネットワークインターフェイスを設定します。

セキュリティー向上のため、自律型ランサムウェア対策の設定をしてください。

3.1.2　スイッチ

スイッチを使用する場合は、NFS over RDMA の性能を引き出すために MTU の値を 9000 に設定します（Jumbo Frame 有効化）。検証環境では、Cisco Catalyst C9500-24Y4C の全ポートの MTU を 9000 に設定しています。

3.1.3　サーバー

■ ネットワークインターフェイス

NFS over RDMA を有効にするため、サーバーがサポートしている ConnectX-5 以上の RoCE 対応のネットワークインターフェイスカードが必要です。
サーバーに対応するネットワークインターフェイスカードは、PRIMERGY の『システム構成図』を参照してください。

- ネットワークインターフェイスのボンディングと MTU の設定
  複数のネットワークインターフェイスをボンディングすることで、サーバー側のデータ経路の冗長化および負荷分散を行うことができます。
  また、NFS over RDMA の性能を引き出すために MTU の値を 9000 に設定します（Jumbo Frame 有効化）。

3.2　GPU Driver のインストール／セットアップ

リファレンス構成である GPUDirect RDMA と GPUDirect Storage では、NVIDIA Open GPU Kernel module driver のインストールが必要です。
https://www.nvidia.com/en-us/drivers/unix/

3.3　Kubernetes のインストール／セットアップ

Kubernetes はコンテナオーケストレーションプラットフォームとして、AI ワークロードのデプロイ、スケーリング、および管理を効率化します。コンテナ化、自動スケーリング、およびリソース管理機能により、AI ワークロードを柔軟かつ効率的に運用できます。

本検証では、シングルノードでも動作し、構築が比較的容易な microk8s を使用します。
以下の URL を参照してインストールしてください。
https://microk8s.io/

3.3.1　NVIDIA GPU Operator の設定

Kubernetes 上で GPU を使用できるようにするために、NVIDIA GPU Operator をインストールします。
NVIDIA GPU Operator は microk8s の add-on として存在しており、nvidia add-on を enable することでインストールできます。ただし、バージョンが古いため最新のバージョンを指定します。nvidia add-on の enable の際に、--gpu-operator-version で明示的に指定します。
また、GPUDirect RDMA および GPUDirect Storage の要件であるオープン 版の GPU ドライバを使うように指定するため、--gpu-operator-set driver.useOpenKernelModules=true をパラメーターとして指定します。

3.3.2 NVIDIA Network Operator のインストール

Kubernetes 上で RDMA に使用するネットワークのリソースを管理するために、NVIDIA Network Operator をインストールします。以下の URL から『 Getting Started with Kubernetes 』を参照してインストールします。
https://docs.nvidia.com/networking/software/cloud-orchestration/index.html
Node Feature Discovery はすでに GPU Operator 側で動作しているため、Network Operator のインストール時には disable します（--set nfd.enabled=false）。

3.3.3 NetApp Astra Trident のインストール

Trident は、Kubernetes 環境で永続ストレージを管理するストレージオーケストレーションツールです。自動化されたストレージプロビジョニング、ストレージクラスのサポート、およびデータ保護機能を提供し、AI ワークロードに必要なストレージを効率的に管理します。
以下の URL を参照してインストールしてください。
https://docs.netapp.com/us-en/trident/

3.4 StorageClass の設定

3.4.1 Trident Backend Storage の設定

「3.3.3 NetApp Astra Trident のインストール」(P.19) でインストールした NetApp Astra Trident のセットアップを行い、実際に使用するストレージを設定します。
https://docs.netapp.com/us-en/trident/trident-use/backend-kubectl.html

備 考

tridentctl コマンドにより作成した TridentBackendConfig は、kubectl による変更や削除ができません。
一般的な Kubernetes の操作では、削除できないリソースとして残る場合があるため、kubectl コマンドを使用して設定した方が安全です。

3.4.2 StorageClass の設定

「3.4.1 Trident Backend Storage の設定」(P.19) で設定した Backend Storage を使用して、Pod が Persistent Volume として使えるように StorageClass を設定します。
https://docs.netapp.com/us-en/trident/trident-use/create-stor-class.html

3.5 Open Source MLOps with NetApp の構築

NetApp の AI ソリューションである Open Source MLOps with NetApp を構築します。
https://docs.netapp.com/us-en/netapp-solutions/ai/aicp_introduction.html

Open Source MLOps with NetApp は、オープンソースの MLOps tool を組み合わせることで、AI ワークフローの扱いを容易にするものです。
MLOps tool にはいくつかありますが、ここでは Kubeflow を使用します。
https://www.kubeflow.org/docs/started/installing-kubeflow/

3.5.1 Kubeflow のインストール

ここでは microk8s と相性の良い Canonical の Charmed Kubeflow（*1）を使用してインストールします。
https://charmed-kubeflow.io/docs/get-started-with-charmed-kubeflow

*1: Charmed Kubeflow と元の Kubeflow の違いは以下のとおりです。
https://charmed-kubeflow.io/docs/charmed-vs--upstream-kubeflow

3.5.2 DataOps Toolkit

DataOps Toolkit のインストールについては、以下の URL を参照してください。
https://github.com/NetApp/netapp-dataops-toolkit

3.6 Pod の設定

以下の URL を参照してください
https://kubernetes.io/docs/concepts/workloads/pods/

3.7　チャットボット FastChat の設定

(1)　huggingface リポジトリのアカウント作成
事前に https://huggingface.co/ にアクセスし、アカウントを作成します。

(2)　ログイン／アクセストークン作成
huggingface リポジトリにログインし、アクセストークンを作成します。

> 備考
>
> FastChat は controller、engine、frontend から構成されます。それぞれを Pod としてデプロイすることを推奨しますが、死活を監視しているので controller、engine、frontend の順番に起動してください。
> 停止は起動と逆の順序で実行してください。engine の再起動および再設定の際も frontend を停止することを推奨します。

(3)　Frondend API server の起動
Frontend は API サーバーとチャット GUI を選択できます。
Host と port は engine を指定します。起動ログに表示されるものを指定してください。

3.8　RAG LangChain 設定

LlamIndex など、ほかの RAG のフレームワークもありますが、ここでは LangChan の構築手順を示します。
以下の URL を参照してください
https://python.langchain.com/v0.2/docs/tutorials/rag/

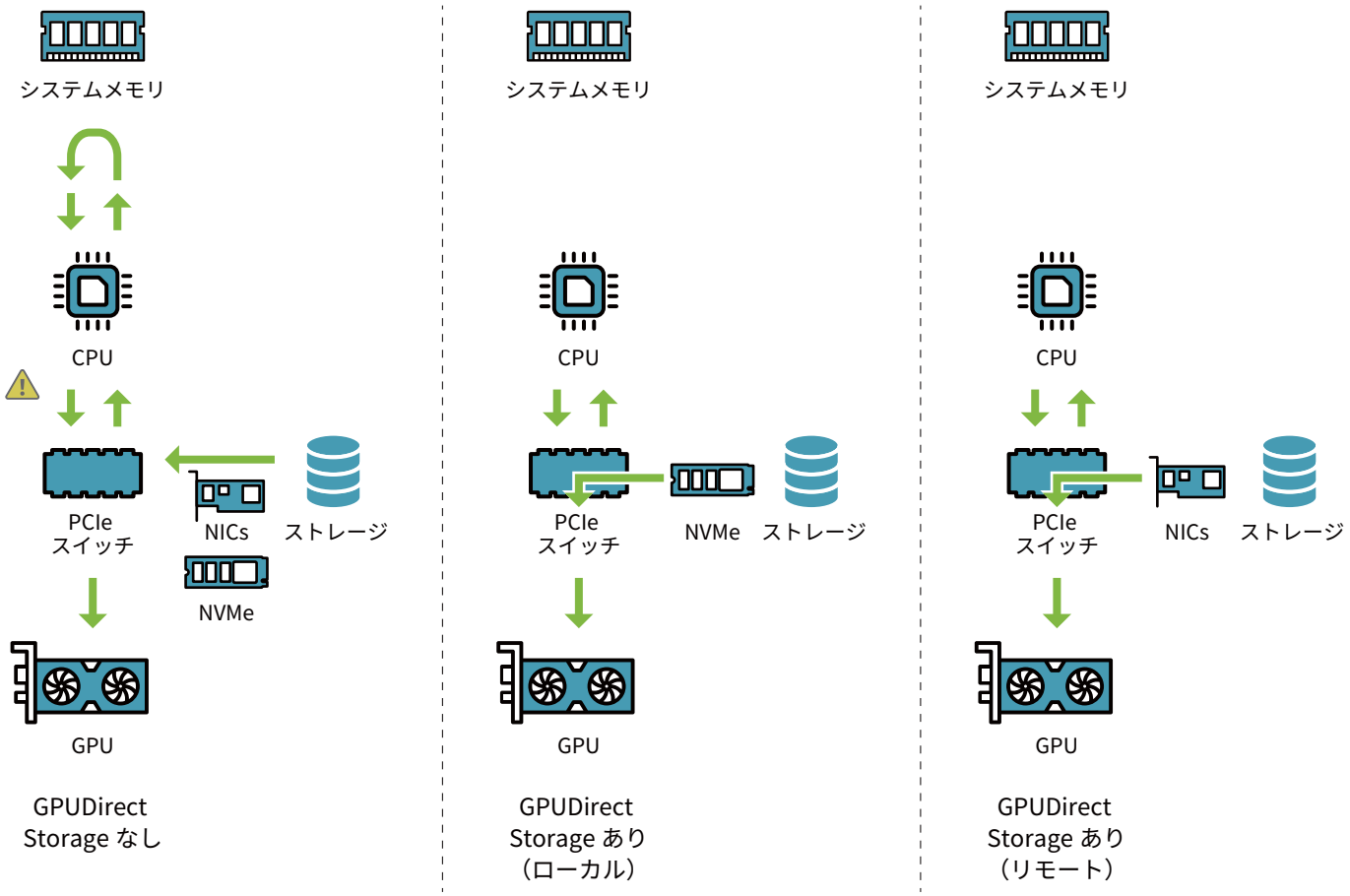
3.9　LLMPerf のインストール

LLMPerf のインストールについては、以下の URL を参照してください。
https://github.com/ray-project/llmperf

3.10 【参考】GPUDirect Storage と NFS over RDMA

GPUDirect Storage（GDS）は、NVMe や NVMe over Fabrics（NVMe-oF）などのローカル、またはリモートストレージと GPU メモリの間に直接データパスを作成します。これにより、CPU を介さずに高速なデータ転送を実現し、処理速度を大幅に向上させます。

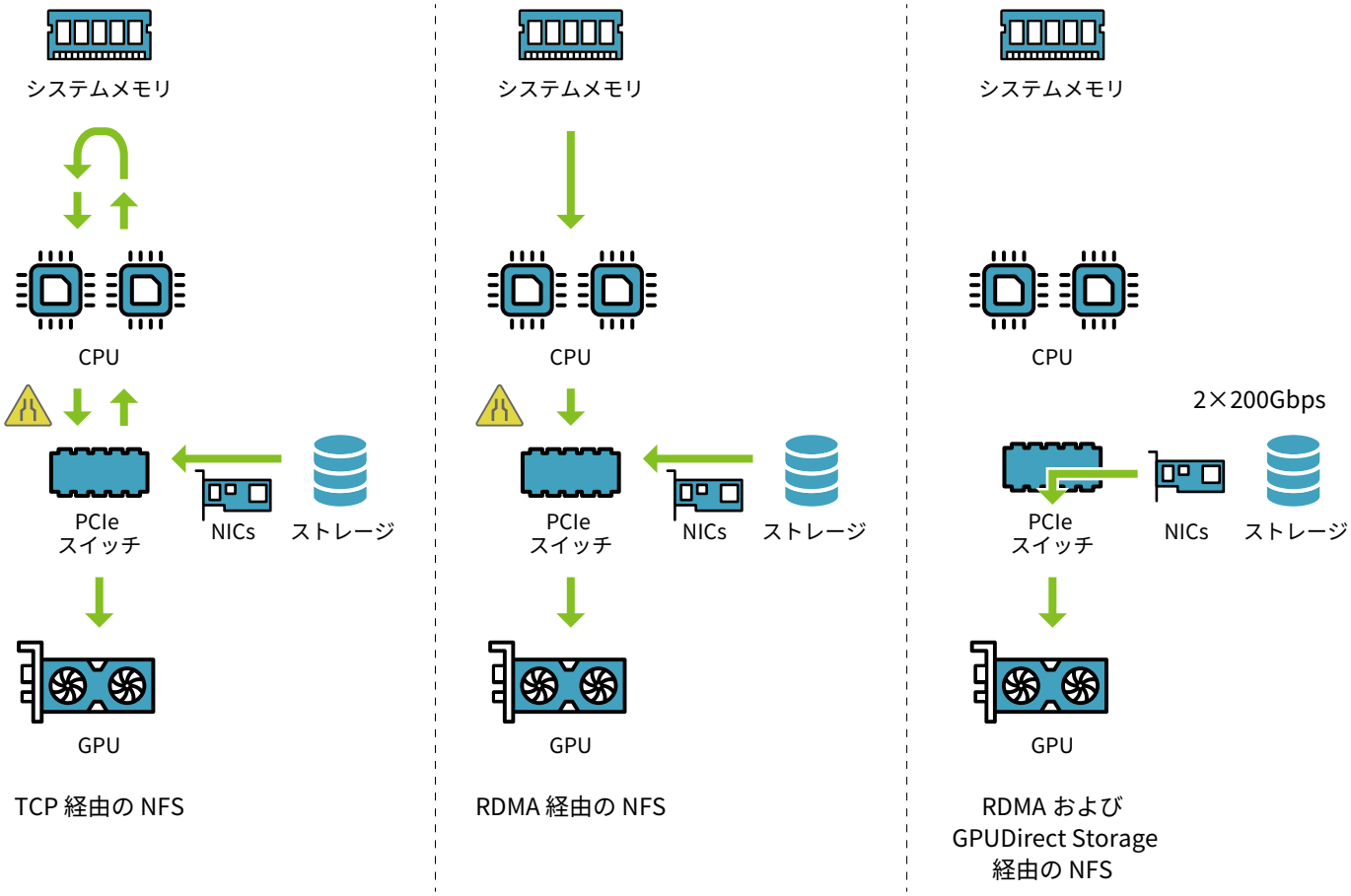
図 3.1　GPUDirect Storage



また、NFS over RDMA は TCP に代わり、RDMA（Remote Direct Memory Access）プロトコルを使用して NFS（network file system）にアクセスすることにより、システムメモリに対してリモートストレージから高速なデータ転送を実現します。PRIMERGY、ETERNUS AX/AC series でサポートされている ConnectX-5 以上の RoCE（RDMA over Converged Ethernet）対応ネットワークインターフェイスカード、ネットワークアダプターを使用することで、NFS over RDMA でのアクセスが可能となります。

GDS は、RDMA を使用してストレージシステムと GPU メモリ間でデータを直接転送することで、CPU とメインメモリを完全にバイパスし、これにより GPU 対応のワークロードをさらに高速化します。

図 3.2　NFS over RDMA



4　検証手順

本章では、ETERNUS AX/AC series を活用して、LLM の性能評価と機能評価を実施します。

4.1　検証構成

検証構成については、「2 リファレンス・アーキテクチャー」(P.7) を参照してください。

4.2　検証

4.2.1　LLMPerf の実行

LLMPerf 実行のパラメーターを示します。

- モデルは FastChat の engine と同じものを指定してください。
- Tokenizer の関係でモデル名を正しく設定しない場合は、token 数を正しくカウントできません。
- Input/output token 数とそのばらつきを設定します。
  ここでは通常のチャット QA を想定し、以下のように設定しました。

```
$ python token_benchmark_ray.py --model "Meta-Llama-3-8B-Instruct" --mean-input-tokens 150 --stddev-input-tokens 15 --mean-output-tokens 150 --stddev-output-tokens 15 --max-num-completed-requests 10 --timeout 60 --num-concurrent-requests 1 --results-dir "result_outputs" --llm-api "openai" --additional-sampling-params '{}'
```

  - Max num completed requests
    試行回数です。結果にばらつきが出る場合は、試行回数を増やしてください。
  - Num concurrent requests
    同時リクエスト数です。FastChat に対して同時に何人に話しかけるかを設定します。システムの要求仕様に合わせて設定してください。
  - Llm-api
    FastChat の API サーバーと同じものを指定してください。

4.2.2　LLMPerf の実施結果の確認

正常に実行されると以下のようなログが出力されます。17.4（token per sec）の出力が確認できます。

```
inter_token_latency_s

（省略）

Number Of Errored Requests: 0
Overall Output Throughput: 17.386019685924794
Number Of Completed Requests: 8
Completed Requests Per Minute: 7.444504415025781
```

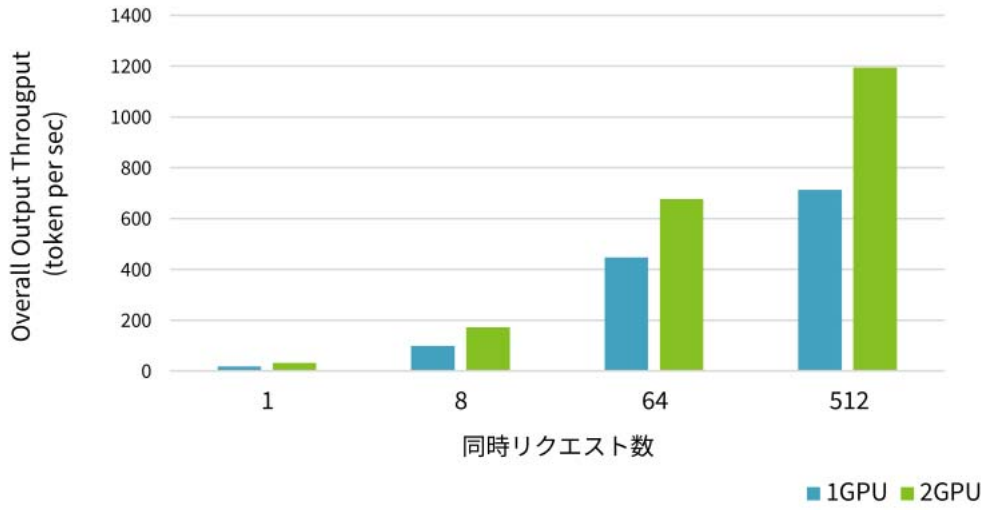
5　評価結果

5.1　性能評価

■ LLMPerf

評価構成で示したシステムで LLMPerf のベンチマークを行った結果、およびベンチマークの主なパラメーターを以下に示します。使用したモデルは、Meta 社が 2024 年 5 月に発表した Llama2 の後継モデルの Llama 3 です。8 ビリオンのパラメーターと軽量であるのに加え、日本語の性能も改善されています。入出力の token 数の設定は、一般的な QA チャットを想定しています。

```
LLM model: meta-llama/Meta-Llama-3-8B-Instruct
input token 150 std 15, output token 150 std 15
```



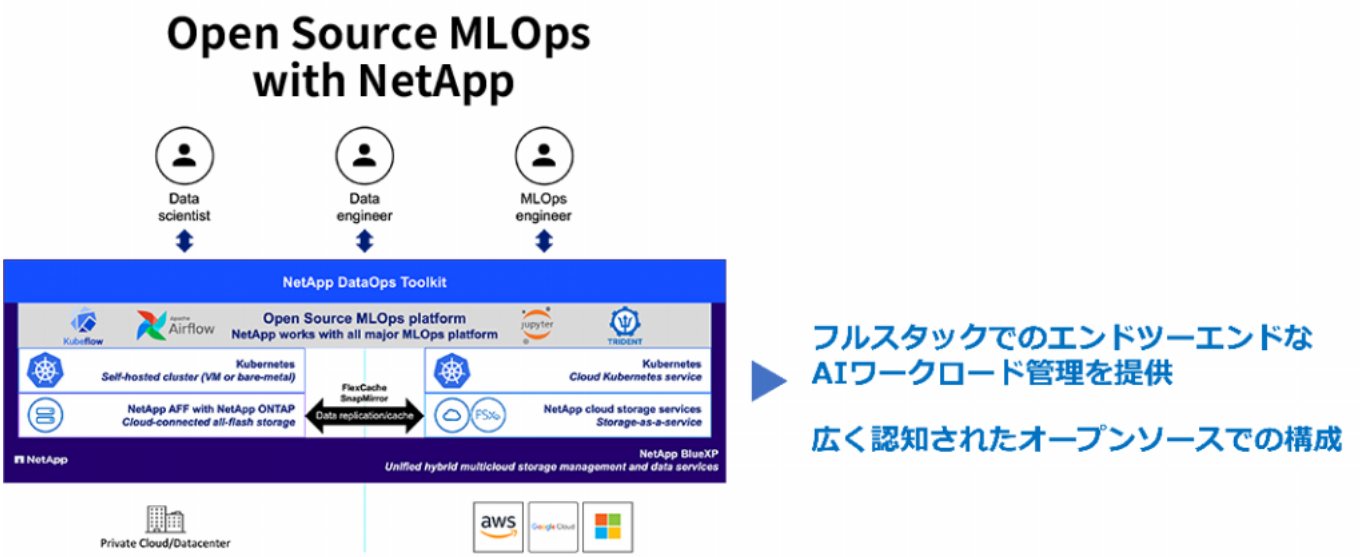
日本人の平均読書速度は 500 ～ 600 文字／分と言われています。Tokenizer の性質にも依存しますが、5 から 10 token per sec の出力が得られればストレスなく質問と応答ができていると考えられます。

上記の結果から 1GPU では同時リクエスト数が 8 程度、2GPU では 64 程度であればこの要件を満たします。
例えば、一般的な QA で、平均 1 人あたり 1 日に 1 回程度質問するシステムを構築する際に、8,000 人程度の規模で 8 時間労働する会社を想定した場合、1GPU で 64,000 人程度の会社では 2GPU を選択すると GPU の性能は十分と言えます。

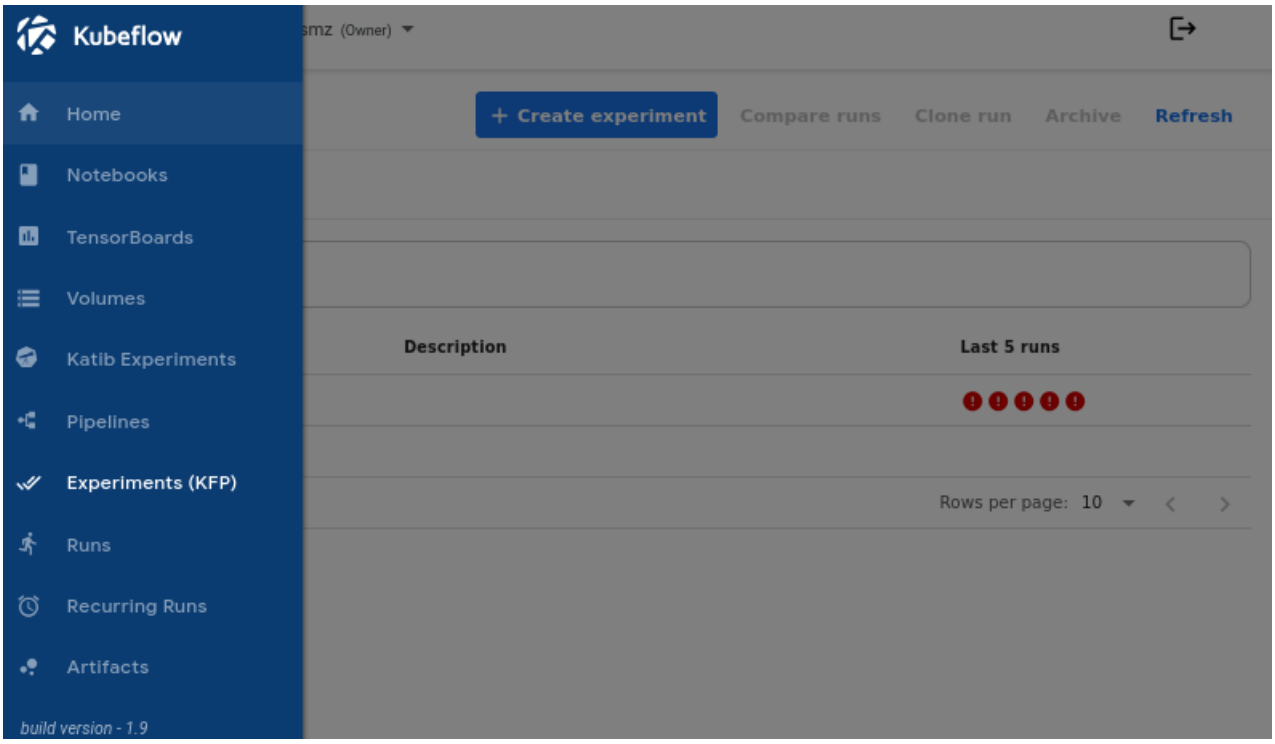
5.2　機能評価

■ Open Source MLOps with NetApp

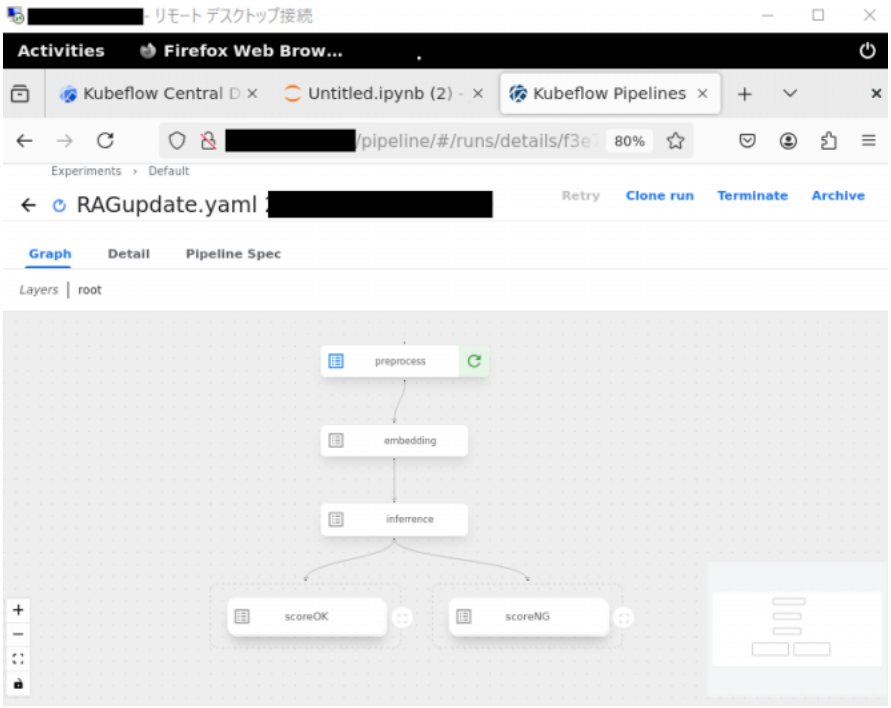
サービス運用段階では、属人的な作業はトラブルのもととなります。DevOps であればこそ作業は自動化し、素早く開発環境から運用環境に移行できます。
Open Source MLOps with NetApp は、AI の領域での DevOps（MLOps）を容易に実現できるツールを提供します。Kubernetes 上で workflow を実現する kubeflow を中心に、AI 開発者が動的にボリュームを要求可能な trident などを含むオープンソースでフルスタックの MLOps 環境を構築します。



Kubeflow では、MLOps で必要となる様々な作業を自動化し、実行結果を Dashboard から一覧で表示できます。



例えば、RAG の vectorDB の更新は、頻繁に不定期に発生するかもしれません。Data source の更新を機会に、Snapshot の作成、Preprocess、Embedding、推論によるスコア判定、およびクリアしたときのみ vectorDB の更新までを自動化しておけば、Data source の更新で精度が落ちても素早く気づき、Snapshot で Restore できます。この Snapshot の作成や Restore などのボリューム管理を trident で動的に行えます。



NetApp 社の以下の Web サイトで AI に関連する Pipeline のテンプレートが公開されています。これを利用すれば、スクラッチから Pipeline を作成する手間を省けます。

- Pipeline サンプル

  https://github.com/NetApp/netapp-dataops-toolkit/tree/main/netapp_dataops_k8s/Examples/Kubeflow

DataOps toolkit

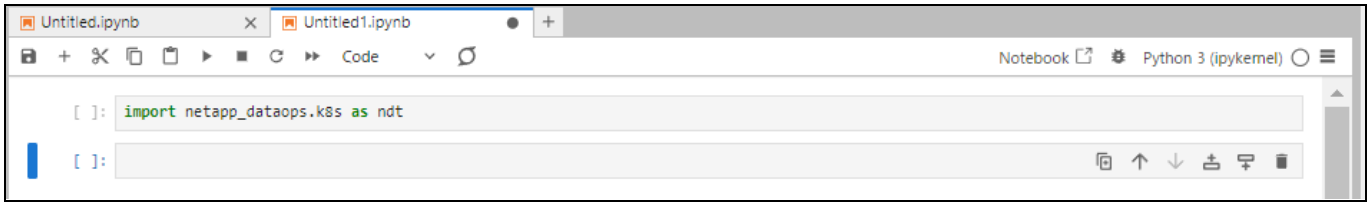
AI の迅速な開発には、AI 開発者がデータの管理を主体的に行う必要があります。しかし、多くの組織では AI 開発者とストレージの管理者は異なります。ボリュームの要求と削除、Snapshot の作成と Restore を AI 開発者がその都度 storage の管理者に依頼していては、組織が効率的に活動できません。DataOps toolkit はこのような状況を改善する手段を提供します。

DataOps toolkit は、Triton Inference server、および Jupyterlab の環境を簡単に Kubernetes cluster にデプロイするコマンド、および Jupyter からインポートして AI 開発者がボリュームの作成／削除、Snapshot の作成／削除と Restore する手段を提供します。

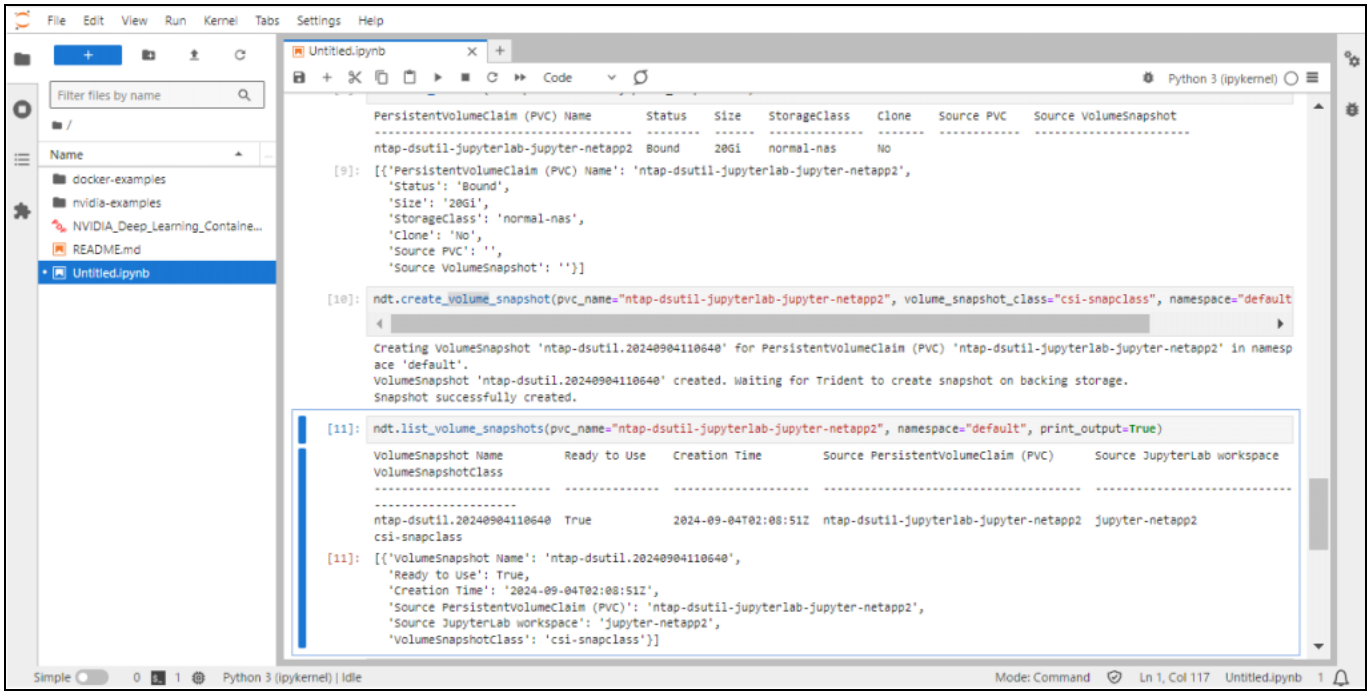
以下のコマンドで jupyterlab の環境を配備します。

```
$ netapp_dataops_k8s_cli.py create jupyterlab --storage-class=<作成した SC> --workspace-name=jupyter-netapp --size=20Gi --nvidia-gpu=2
```

次に、配備された Pod にログインし、Pod にも netapp_dataops-k8s をインストールします。
ブラウザから、起動ログに記載された WEB UI にアクセスします。インストールしたライブラリ netapp_dataops.k8s をインポートし、正常に動作しているか確認してください。



以下は、実際に kubernetes に Jupyer の環境をデプロイし、その中から Snapshot を作成する例です。
データ管理に必要なワークスペースの情報、ボリュームの情報、および日時がメタデータとして付加されています。



6　まとめ

本書では、生成 AI チャットボットの開発環境構築における、ETERNUS AX/AC series を活用したリファレンス・アーキテクチャーを紹介しました。生成 AI チャットボットの概要、アーキテクチャー、構築手順、検証結果、性能評価、および機能評価を網羅し、ETERNUS AX/AC series が AI 運用におけるストレージ課題解決に貢献することを示しています。

付録 A　関連資料

- ETERNUS AX2200
  https://www.fujitsu.com/jp/products/computing/storage/all-flash-arrays/ax/

- ETERNUS AC2100
  https://www.fujitsu.com/jp/products/computing/storage/all-flash-arrays/ac/

- ETERNUS 自律型ランサムウェア対策
  https://www.fujitsu.com/jp/products/computing/storage/product-feature/axhx-anti-ransom.html
  https://www.fujitsu.com/jp/products/computing/storage/axhx/feature/protect-ransomware.html

- ETERNUS AX/AC series マニュアル
  https://www.fujitsu.com/jp/products/computing/storage/manual/axhx-list.html
  https://storage-system.fujitsu.com/manual/ja/axhx/introduction-concepts/index.html

- ETERNUS 動画マニュアル
  https://www.fujitsu.com/jp/products/computing/storage/eternus/video-library/ax-hx/#5min

- ONTAP
  https://storage-system.fujitsu.com/manual/ja/axhx/introduction-concepts/index.html

- PRIMERGY
  https://jp.fujitsu.com/platform/server/primergy/products/lineup/rx2540m7/

- PRIMERGY マニュアル
  https://www.fujitsu.com/jp/products/computing/servers/primergy/manual/

- AI 基盤サービス
  https://www.fujitsu.com/jp/products/computing/servers/primergy/solution/ai-infra/

- GPU Driver（NVIDIA）
  https://www.nvidia.com/en-us/drivers/unix/

- Llama 3
  - 公式ウェブサイト
    https://llama.meta.com/
  - GitHub リポジトリ
    https://github.com/facebookresearch/llama

- microk8s インストール
  https://microk8s.io/

- Getting Started with Kubernetes
  https://docs.nvidia.com/networking/software/cloud-orchestration/index.html

- NetApp Astra Trident マニュアル
  https://docs.netapp.com/us-en/trident/

- NetApp Astra Trident セットアップ
  https://docs.netapp.com/us-en/trident/trident-use/backend-kubectl.html

- StorageClass 設定
  https://docs.netapp.com/us-en/trident/trident-use/create-stor-class.html

- Open Source MLOps with NetApp
  https://docs.netapp.com/us-en/netapp-solutions/ai/aicp_introduction.html

- Kubeflow インストール
  https://www.kubeflow.org/docs/started/installing-kubeflow/

- Charmed Kubeflow インストール
  https://charmed-kubeflow.io/docs/get-started-with-charmed-kubeflow

- DataOps Toolkit インストール
  https://github.com/NetApp/netapp-dataops-toolkit

- POD 設定
  https://kubernetes.io/docs/concepts/workloads/pods/

- RAG LangChain 構築
  https://python.langchain.com/v0.2/docs/tutorials/rag/

- LLMPerf インストール
  https://github.com/ray-project/llmperf

- Pipeline サンプル
  https://github.com/NetApp/netapp-dataops-toolkit/tree/main/netapp_dataops_k8s/Examples/Kubeflow

生成 AI を実現する PRIMERGY/ETERNUS の最新テクノロジー
AI ストレージ リファレンス・アーキテクチャー

C140-0133-01Z3

発行年月　2024 年 9 月
発行責任　エフサステクノロジーズ株式会社