コンテナ技術を用いた 教育情報システムの構築

175733E 宮平 賢
指導教員：河野 真治

Abstract

ab,ab,ab,ab, abstract

1 教育向けの情報システム

情報通信技術の普及に伴い学生が学ぶ学習環境が必要となる．その学習環境として VM やコンテナにより，手軽に開発し試せる技術が普及している．だが，手元の PC 上で VM やコンテナを立ち上げ，開発を行うことはできるが，VM やコンテナの使用には高性能 PC や有料のクラウドサービスが必要になる場合がある．これらの負担を IT 技術を学ぶ学生に負わせない新たな仕組みが必要である．

本コースでは希望する学生に学科の汎用サーバから仮想環境を貸出するサービスを行っている．貸出をする VM の基本スペックとして CPU1 コア，メモリ 1GB，ストレージ 10GB である．基本スペックでは不足する場合は要望に応じてスペックの変更を行っている．しかし，機械学習などの演習では CPU より GPU が求められる場合がある．VM 上で GPU を共有するには PCI パススルーを利用することで可能である．だが，PCI パススルーでは GPU と VM は 1 対 1 の関係となり，GPU を希望する利用者すべてに割り当てることはできない．

本研究では，学生が貸出 VM だけでなく，学科の汎用サーバのリソースを効率的に利用できる教育情報システムを提案する．教育情報システムには複数の汎用サーバと大容量ストレージサーバが存在する．複数のサーバを利用するにあたり，分散ストレージが必要となる．また，学習環境として利用されることから，複数の並列なアクセスに耐えられ，信頼性の高いファイルシステムが必要である．この用件を満たすストレージソフトウェアとして Ceph を採用した．汎用サーバのリソースを効率的に利用するために，コンテナエンジンである Podman，Singularity，ジョブスケジューラである Slurm を採用した．これらのソフトウェアを合わせ教育情報システムの構築を行った．

2 Podman

Podman は RedHat 社が開発，提供する Linux 上で OCI コンテナを開発，管理，実行するためのデーモンレスコンテナエンジンである [1]．Podman は OCI 準拠のコンテナランタイムに依存するため，前述した Docker など他のコンテナエンジンと互換性を持つ．また，Podman CLI は Docker CLI と同じ機能を提供する．Podman はコンテナとイメージストレージ，コンテナランタイムを介して Linxu カーネルと直接対話することで，デーモンレスで実行される．Podman の制御下にあるコンテナは，特権ユーザ又は非特権ユーザのいずれかによって実行することができる．

3 Singularity

Singularity[2] とは，HPC 環境向けに設計されたコンテナプラットフォームである．Singularity はマルチユーザに対応しており，コンテナ内での権限は実行ユーザの権限を引き継ぐため，ユーザに特別な権限の設定が必要ない．またデフォルトで，$HOME，/tmp，/proc，/sys，/dev がコンテナにマウントされ，サーバ上の GPU を簡単に利用できる．コンテナイメージは Singularity Image Format(以下，sif) と呼ばれる単一ファイルベースのため，アーカイブや共有が容易である．

4 Slurm

Slurm[3] は Linux クラスタ向けのフォールトトレラント設計のジョブスケジューリングシステムである．Slurm には以下の 3 つの主要機能を提供する．

- 計算を実行するユーザに対してリソースへの排他的，非排他的なアクセスを割り当てる
- 割り当てられたノード上のジョブの開始，実行，モニタリングを行う
- 待機中のジョブキューを管理することにより，リソースの競合を解決する

5 Ceph

Cephは，RedHat社が開発，提供する分散ファイルシステムである．Cephは分散オブジェクトストレージであるRADOS(Reliable Autonomic Distributred Object Storage)がベースとなっている．RAODSでは，Object Storage Daemonにデータ格納する．オブジェクトの配置には，クラスタマップを元にControlled Replication Under Scalable Hashing(以下，CRUSH)アルゴリズムによりオブジェクトの格納先を選択する．配置の計算に必要とする情報はごくわずかであるため，Cephクラスタ内のすべてのノードは保存されている位置を計算できる．そのため，データの読み書きが効率化される．また，CRUSHはデータをクラスタ内のすべてのノードに均等に分散しようとする．

RODOSはクラスタに保存されるデータの管理を待ち受け，保存オブジェクトへのアクセス方法としてObject Gateway，RADOS Block Device(以下，RBD)，CephFSがある．Object GatewayはHTTP REST経由でクラスタに保存されるオブジェクトへ直接アクセスが可能である．RBDはブロックデバイスとしてアクセスが可能で，libvirtを組み合わせてVMのディスクとして使用できる．また，RBDドライバを搭載したOSにマップし ext4 や XFS などでフォーマットして利用できる．CephFS は POSIX 互換のファイルシステムである．複数のアクセス方法を提供することで，用途に合わせ柔軟に変更することができる．

6 教育情報システムの構築

新システムは，VMベースのシステムからコンテナベースへ移行する．新システムでもVM貸出サービスを継続するが，新しく搭載されるGPUをVMで利用するためにはPCIパススルーなどの設定が必要となる．しかし，PCIパススルーでは，VMとGPUが1対1の関係になるため，GPU希望する利用者全てに割り当てることができない．そのため，コンテナに移行することにでリソースを効率よく利用し，管理を容易にする狙いがある．また，基幹サービスのデータをSSD上に保存することで，サービスの高速化を図る．

システムは学生や教授などが利用するため，マルチユーザで利用できるコンテナエンジンが必要となる．そこで，マルチユーザに対応しているPodmanとSingularityを採用する．Podmanは独自の名前空間でコンテナ内で特権機能を利用できるため，特権が必要なサービスなどの実行に向いている．また，Singularityはコンテナ内で実行ユーザの権限を引き継ぐため，利用者が作成したプログラムの実行には向いている．だが，Podmanの rootless では実行できない機能があり，Singularityではイメージの Build がキャッシュされず低速である．そこで，Podmanを wrapper したie-podmanを作成した．

7 教育情報システムの利用

構築した教育情報システムの管理方法，利用方法について述べる．

7.1 ie-podmanの利用

ie-podmanはPodmanを wrapp し複数ユーザで利用することができるコンテナ管理ツールである．Podmanはマルチユーザに対応しているため，ie-podmanを利用せずともコンテナの作成などを行える．だが，コンテナへのIPアドレスの割り当てには，root権限が必要となるため rootlessでは実行できない．そこで，ie-podmanでは Podmanのすべての機能を wrapp するのではなく，rootlessでは実行できない機能を提供する．

新しいコンテナの作成は，Podmanの runと同じように実行できる．run時に–gpuオプションを指定することでコンテナ内にGPUを割り当てる．また，–ipオプションを指定することで，使用されていないIPアドレスが割り振られる．ie-podmanを使用して，新しいコンテナの作成はソースコード1のように行う．

ソースコード 1: コンテナの作成

```
$ ie-podman run --ip --gpu [IMAGE_NAME]
```

Singularityから sifファイルの作成は Podmanと違いイメージの Build時にレイヤーごとにキャッシュされない．そのため，Build中にエラーが発生すると一から Buildを再開する必要がある．そこで，ie-podmanで作成したイメージを sifファイルへ変換する機能を作成した．ie-podmanでイメージを作成し，ソースコード2の操作を行うことで sifファイルへ変換が行える．

ソースコード 2: イメージの sif 変換

```
$ ie-podman sif [IMAGE_NAME]
```

7.2 GPUの利用方法

新システムでは，汎用サーバに搭載されるGPUをコンテナから利用できる．Singularityから GPUを利用には–nvオプションを指定することで，コンテナから GPUを利用することが可能になる．Singularityのコンテナの実行は，ソースコード3の操作で行える．

ソースコード 3: Singularity の実行

```
$ singularity run --nv [SIF_NAME]
```

コンテナの実行には run，exec，shellのサブコマンドがあり，runでは sifファイルを作成する際に指定が可能な runscriptが実行される．また，execではイメージ内にインストールされている任意のコマンドを実行することが可能

である．Podman や Docker では exec を実行するにはコンテナを作成する必要があるが，Singularity では sif ファイルからコマンドを実行することが可能である．これらのサブコマンドを利用し，Slurm に Job を投下時に利用する Job の実行手順を記述した Batch ファイルを作成する．Batch ファイルには Job に必要とするリソースの定義，Job で実行したい処理を記述する．ソースコード 4 は 2~8 行目に Job に必要とするリソースを定義する．リソースの定義した後にプログラムを実行する処理を記述する．

ソースコード 4: Batch ファイル

```
1  #!/bin/bash
2  #SBATCH --job-name sample
3  #SBATCH --output logs/%x-%j.log
4  #SBATCH --error logs/%x-%j.err
5  #SBATCH --nodes 1
6  #SBATCH --cpus-per-task 8
7  #SBATCH --gpus tesla:1
8  #SBATCH --time 01:00
9
10 singularity exec --nv [SIF_NAME] [COMMANDS]
```

Batch ファイルを作成後，ソースコード 5 の 1 行目の操作で Job を投下することが可能である．また，2 行目の操作で Job の各種情報，3 行目で投下した Job を停止することができる．Slurm はユーザごとに Job が管理されるため，他ユーザの Job を停止するこはできない．

ソースコード 5: Job の投下

```
1 sbatch [BATCH_FILENAME]
2 squeue
3 scansel [JOB_ID]
```

8 教育情報システムの評価

8.1 ie-podman の評価

Rootless の Podman，Singularity の不便な点を補うため，Podman の wrapper である ie-podman を作成した．ie-podman により特権が必要な機能も，利用者に特別な権限を与えることなく利用できるようになる．また，ie-podman は rootfull の Podman を wrapper することにより，コンテナやイメージが SSD 上に保存される．そのため，rootless の Podman より高速化を図ることが可能になる．

そこで，ie-podman でのイメージの Build 速度の比較を行う．速度の比較を行うコンテナエンジンは，Docker，ie-podman，rootless の Podman である。図 1 はコンテナエンジンにおけるイメージの Build 速度である．

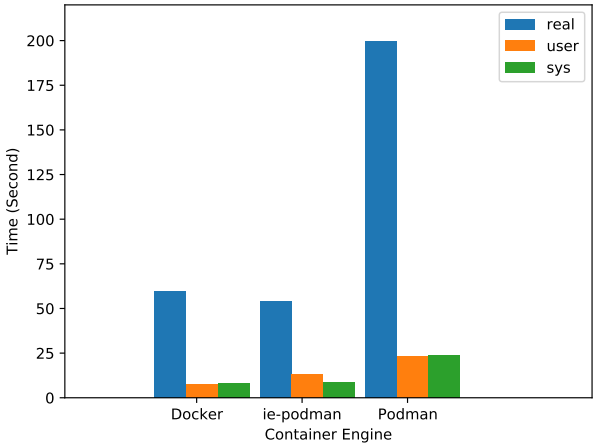

図 1: 書込み速度の比較

Docker や ie-podman の Build に掛かる時間は 1 分未満だが，rootless の Podman では 3 分程掛かっている．Rootless の Podman はコンテナイメージをユーザのホームディレクトリに保存する．また，rootless では重複排除をサポートしていない VFS ストレージに制限される．Rootless の Podman は独自の名前空間内で特権機能を利用できるようにするため，rootfull と比べ経由する関数が多くなる．そのため，rootless では rootfull と比べ syscall が多く呼ばれることにより，他と比べイメージの Build 速度が遅くなっているのではないかと考える．

8.2 ファイルシステムの評価

旧システムの VM 保存場所として利用していた GFS2，ユーザのホームディレクトリとして利用していた NFS との速度比較を行う．ベンチマークには dd コマンドを使用する．データの変換方法に fdatasync を指定することで，書き込み終了の直前に sync を 1 回要求するため，実際の動作に近い動作で測定が可能である．

図 2 は CephFS，Ceph RBD，GFS2，NFS におけるファイルサイズに対する書き込み速度である．

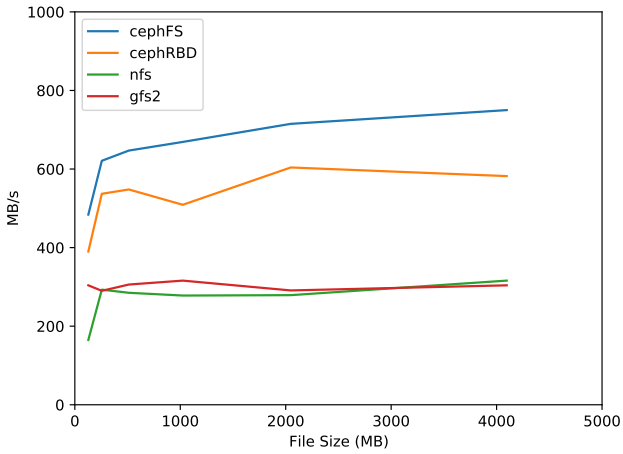

図 2: 書込み速度の比較

旧システムのホームディレクトリは，iSCSI 経由でマウントされたデバイスを NFS から提供していた．iSCSI の通信には 10Gbps の回線で接続されているが，NFS の提供は VM で行われており，1Gbps で提供されていた．そのため，10Gbps の回線で接続し，マウントしている Ceph では書き込み速度の改善が見られる．しかし，GFS2 は 10Gbps で接続されたクラスタで構成されているが，Ceph より低速である．旧システムでは，パッケージ等のアップデートがされておらず，Kernel の更新もされていなかった．Kernel は I/O に関する多くの機能を提供するため，GFS2 の書き込みより，Ceph が高速になったのではないかと考えられる．

今回の計測では，読み込み速度の測定を行えなかった．これは，旧システムで読み込み時にバッファキャッシュを削除せずに測定を行ったためである．そのため，純粋な読み込み速度を測定することができなかったことは反省点である．

9　まとめ

今年度のシステム更新で教育情報システムの構築を行い，VM ベースからコンテナベースへの移行ができた．また，学生が利用できる学習環境として VM 貸出サービスだけでなく，コンテナ環境も利用できるようにしたことにより，学生が自由にサーバのリソースを利用できるようになった．コンテナ環境として採用した Podman，Singularity の不便な点を補うために作成した ie-podman の評価を行った．新しく採用した Ceph と，旧システムのファイルシステムとして利用された GFS2，NFS との書き込み速度の比較を行い，速度向上がみられた．

10　今後の課題

旧システムの VM 貸出サービスは講義等で告知されたりしたが，実際にはあまり周知されておらず利用も少なかった．これは，システム管理チームからの利用方法について周知等が少なかったことも原因として挙げられる．本研究で構築した教育情報システムは，VM からコンテナまで利用できる．だが，利用は主に CLI から操作を行い，プログラムの実行には Slurm を利用する．VM 貸出サービスの変更や，コンテナ環境の利用方法についてまとめる必要がある．また，Slurm の Job の投下方法や必要なリソースの要求方法などをまとめ，定期的な周知を行う必要がある．

ie-podman で使用するネットワーク構成はプレフィックス長が 24 であるため，最大 254 個の IP アドレスしか割り当てできない．そのため，コンテナを削除せず停止のままでは，割り当て可能な IP アドレスが枯渇する．そこで，ie-podman が利用するネットワーク構成の変更を行う，もしくはコンテナが停止のまま数日経つ場合に ie-podman から自動削除する必要がある．

参考文献

[1] Podman，https://podman.io/，2021/1/4.

[2] Singularity，https://sylabs.io/singularity/，2021/1/8.

[3] Slurm，https://slurm.schedmd.com/overview.html，2021/1/14.

[4] Ceph，https://docs.ceph.com/en/latest/，2021/1/12.

[5] 平良 太貴 and 河野 真治，OS 授業向けマルチユーザ VM 環境の構築，研究報告システムソフトウェアとオペレーティング・システム (OS)(2014).

[6] 城戸翔太，安里悠矢，城間政司，長田智和，谷口祐治，”情報系学科における教育情報システムの構築及び運用管理に関する取り組み”，研究報告インターネットと運用技術 (IOT)(2016).