石川ハイテク交流センター

2014 DataCenter とソフトウェア開発ワークショップ

プライベート Docker コンテナの運用について

2014-11-21

PhalanXware 加藤 真透

目次

- 第1部
  - プライベートDockerの利用
- 第2部
  - 多数の Docker コンテナを運用する場合のパフォーマンス改善
- まとめ

第1部

プライベートDockerの利用

Docker とはなにか

- Docker社が開発するオープンソースのコンテナー管理ソフトウェアの1つである。

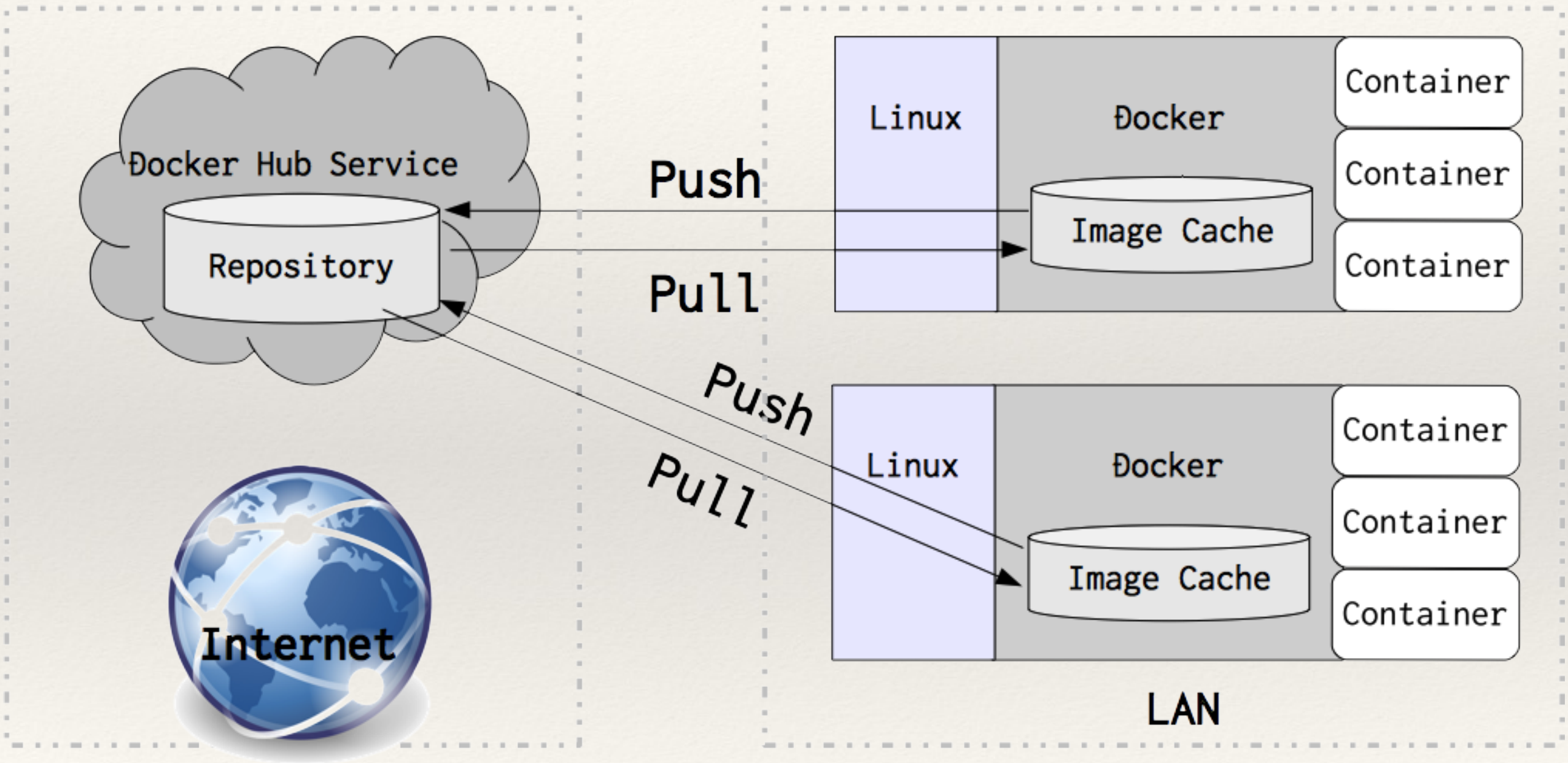
- LXC(LinuxContainers)によりLinux上に隔離された環境を作り出し、ホストの環境に影響を及ぼさずにアプリケーションの実行ができる。

- OSやアプリケーションの導入済みLinux環境を丸ごとイメージ(以降 Docker イメージ)として保存し共有する機能がある。

Docker Hubとは

- Docker イメージを保存し共有するサービスが **Docker Hub** である。
- https://hub.docker.com/ にて Docker社が運営している。
- Docker Hub 上で共有されるイメージは基本的に **パブリック** で、誰でも利用することができる。
- Docker は Docker Hub との連携を前提に作られている。

Docker Hub 利用図



Docker Hub の制約

- 基本的には **パブリック** であるため、**プライベート** な利用ができない。
- 独自のアプリケーションや設定を導入したカスタムイメージであっても、公開されてしまう。
- 非公開なカスタムイメージを運用する場合には、 プライベート **な Repository** が欲しい。

プライベート Repository の利用方法

- Docker Hub の 有料プランを契約する。
- 独自の Repository を準備する。

方法1 有料プランの利用

- Docker Hubには有料プランで、 **プライベート な Repository**を提供している。
- https://registry.hub.docker.com/plans/ より抜粋
  - 1 Repository $0/Month
  - 10 Repository $12/Month
  - 50 Repository $50/Month

方法1 有料プランの利用

- メリット
  - 通常の Docker 利用方法と変わりなく利用可能。
  - イメージ保存のディスク容量を気にしなくて良い。
- デメリット
  - インターネットを経由した Docker Hub へのアクセス・認証を必要とする。
  - Docker Hub の死活にサービスが左右される。
  - 費用がかかる。

利用手順 - 契約

- https://hub.docker.com/ にてアカウントを作成
- My Repositories Pages にて、作成できるプライベート Repositoriesの数がでている(無料では1)。
- BuyMoreとあるので、そのページから Plan を Upgrade することで、数を増やすことができる。

利用手順 - push

- Repositoryのページに移動し、+ Add Repository
- NamespaceとRepository Name を決める。
- Repository Type に **Private** を選択。
- 以下のコマンドでpush (IDとpasswordを入力)

```
docker push <Namespace>/<repository name>
```

  - <Namespace>/<repository name> 決めたものを入力

利用手順 - pull/run

- 以下のコマンドでpull (IDとpasswordを入力)

```
docker pull <Namespace>/<repository name>
```

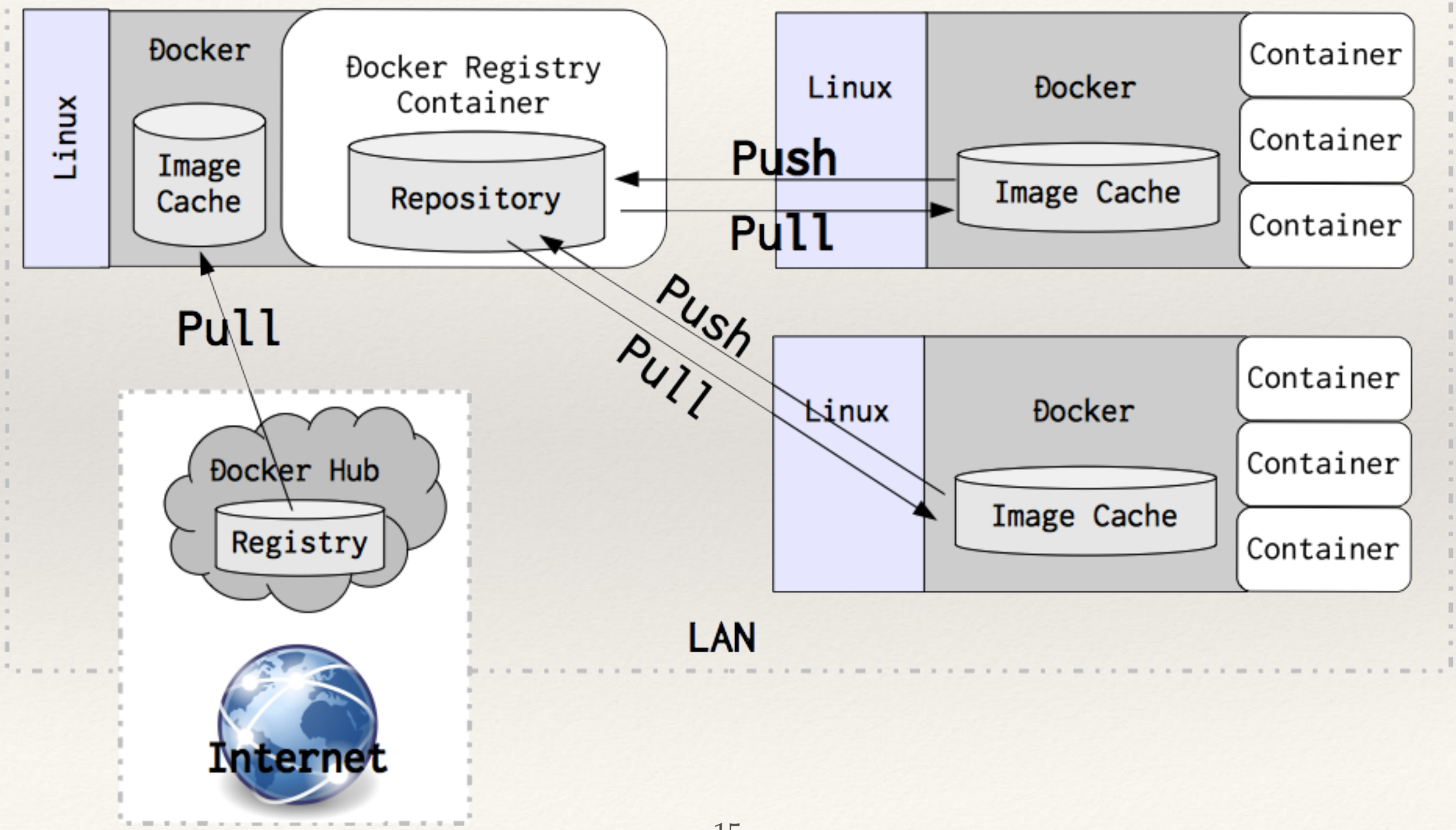
```
docker run <Namespace>/<repository name>
```

- <Namespace>/<repository name> pushしたものを入力

方法2 独自 Repository の運用

- Docker社が、プライベート Repository を運用するための **Docker Registry** というソフトウェアを配付している。
- **Docker Registry** 自身も Docker イメージとして提供されている。
- これを用いることで、**ローカルな環境上に Docker イメージを共有するインフラを作る**ことができる。

方法2 独自Repositoryの運用図



方法2 独自 Repository の運用

- **メリット**
  - インターネットに接続できない環境でも利用可能。
  - Docker Hub の死活に影響されない。
  - 費用がかからない。(ランニングコストは別)
- **デメリット**
  - Docker イメージの利用方法に一部制限がある。(イメージ名にホスト名が含まれる)
  - ディスク容量を使う。(Docker Registry を駆動させる環境を準備する必要がある)

利用手順 -セットアップ

- Docker Registry セットアップ手順
  - Docker Registry 自体も Docker イメージとして提供されている。

- 以下のコマンドで動作させることができる。

```
docker run -d -p 5000:5000 registry
```

-d : デーモンモード

-p : ポートの転送設定

registry : Docker Registry の Docker イメージ名

利用手順 - push

- ホスト名とポートを指定し、そのRepositoryへイメージをpushできる。

```
docker push <registry host>[:<port>]/<image name>
```

  - <registry host> : Docker Registry を駆動させたホスト名
  - <port> : ポート番号
  - <image name>: 保存するイメージ名

- 例) `docker push www1.local:5000/webapp_service`

利用手順 - pull/run

- 同じく、ホスト名とポートを指定し、pull/runすることができる。

```
docker pull <registry host>[:<port>]/<image name>
```

```
docker run <registry host>[:<port>]/<image name>
```

  - <registry host> : Docker Registry を駆動させたホスト名
  - <port> : ポート番号
  - <image name>: pull/runするイメージ名

- 例) `docker pull www1.local:5000/webapp_service`

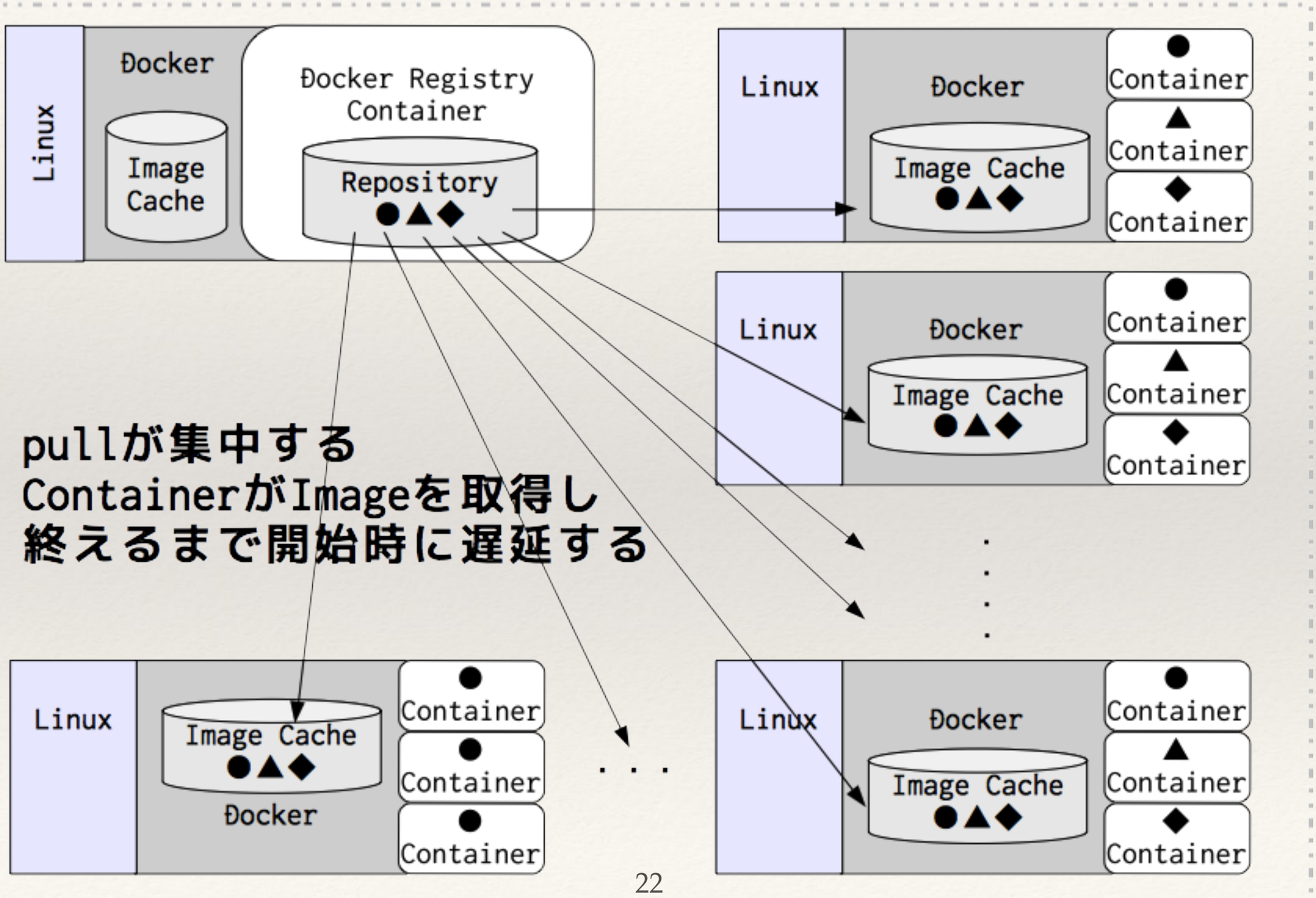
第2部

多数の Docker コンテナを運用する場合のパフォーマンス改善

docker pull ボトルネック

- 全Docker Hostが1つのRegistryよりDockerイメージをpullすることで以下の様な現象がおきる。
  - サイズの大きなイメージファイルをダウンロードするために
    - **コンテナが動作し始めるまでの遅延が増える。**
    - **大量のネットワークリソースとディスクリソースが使われる。**

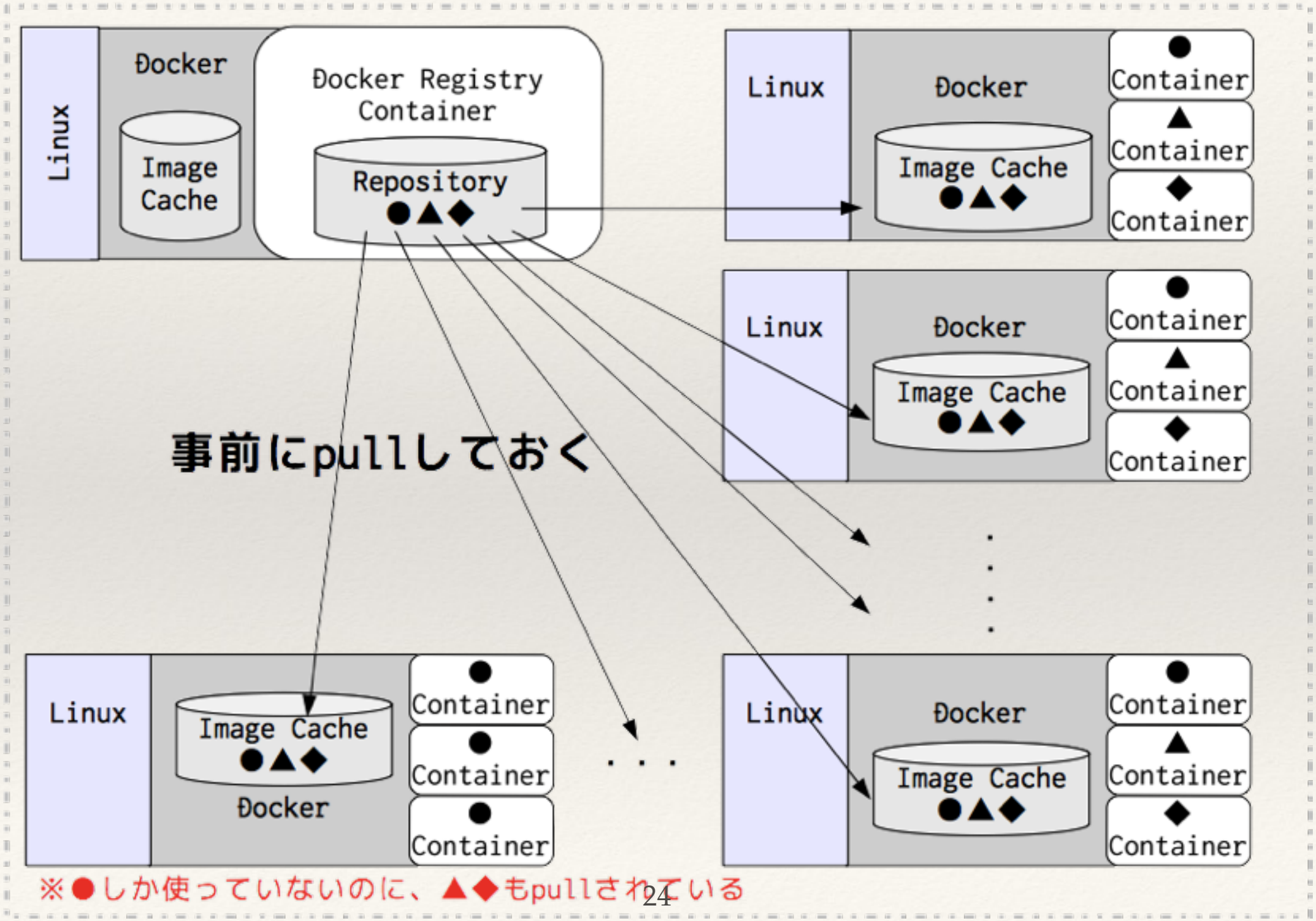
docker pull ボトルネック図



解決案

- 事前pull方式の導入。
- キャッシュシェア方式の導入。

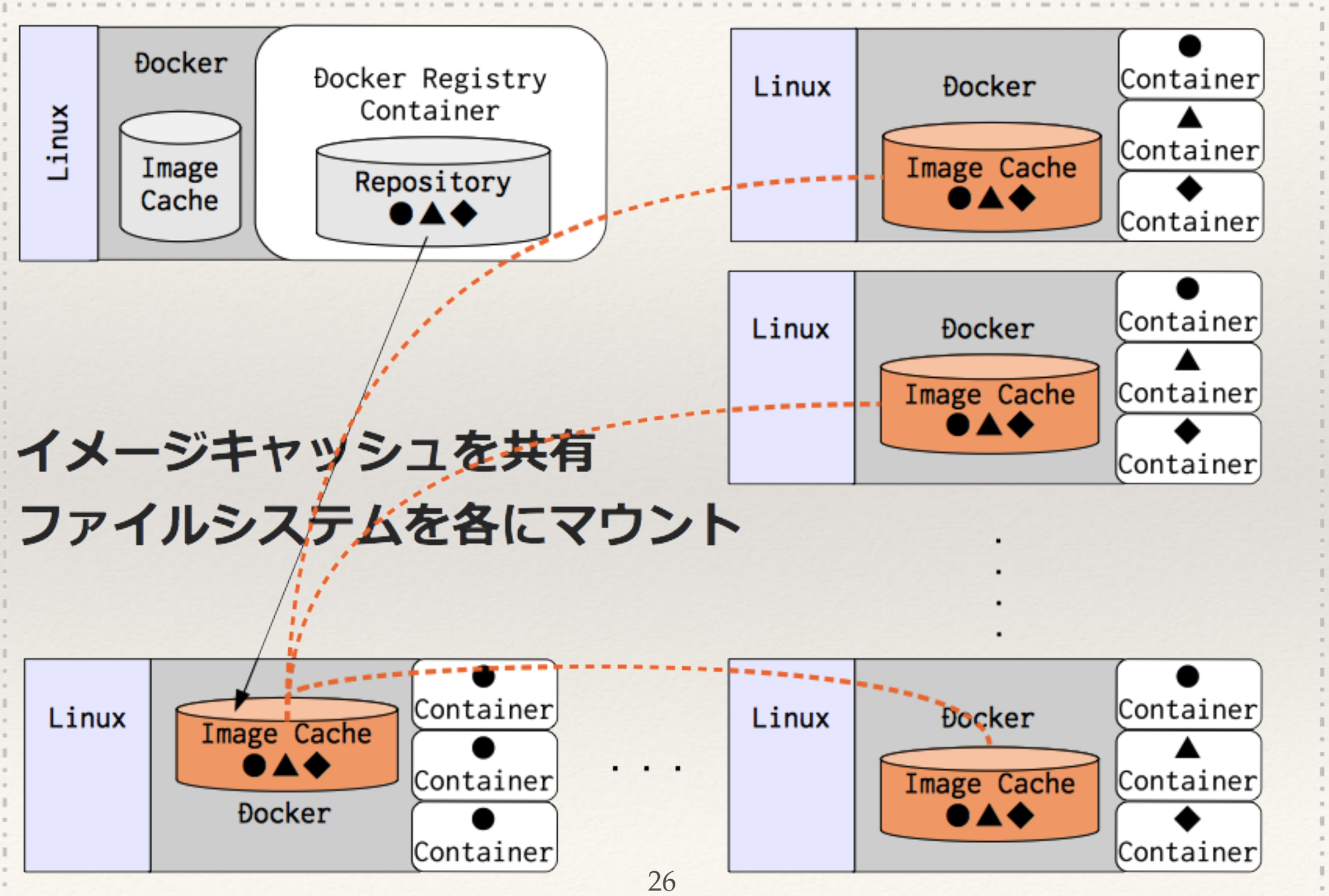
改善案1 事前pull方式図



改善案1 事前pull方式

- 必要になる前にあらかじめ、Dockerイメージをpullしておく。
  - **メリット**
    - イメージを事前にキャッシュするため、コンテナ駆動時には pull が発生しないため駆動時の遅延を抑えることができる。
  - **デメリット**
    - 使わないイメージも転送しうるため、余計なネットワークリソースとディスクリソースを利用する可能性がある。

改善案2 キャッシュシェア方式図



改善案2 キャッシュシェア方式

- メリット
  - 各DockerHostが個別にイメージを取得する必要がないため、ネットワークリソース、ディスクリソースを節約することができる。
- デメリット
  - 計画的な pull スケジューリングが必要。
    - Docker イメージのコンフリクトを抑止するため。
    - 必要な時に必ずキャッシュを存在させるため。
  - コンテナの駆動速度が遅くなる場合がある。
    - 共有ディスクのパフォーマンスがボトルネックになる可能性があるため。

まとめ

- Docker と Docker Hub について解説した。
- プライベート な Repository の運用方法を提示した。
- Docker イメージの配布方法の改善について考察した。
  - ボトルネックが存在することを示し、解決方法の以下2種を示した。
    - 改善案1. 事前pull方式。
    - 改善案2. キャッシュシェア方式。