

研究室インターン最終発表

京都大学大学院 情報学研究科

通信情報システムコース

安済翔真

背景

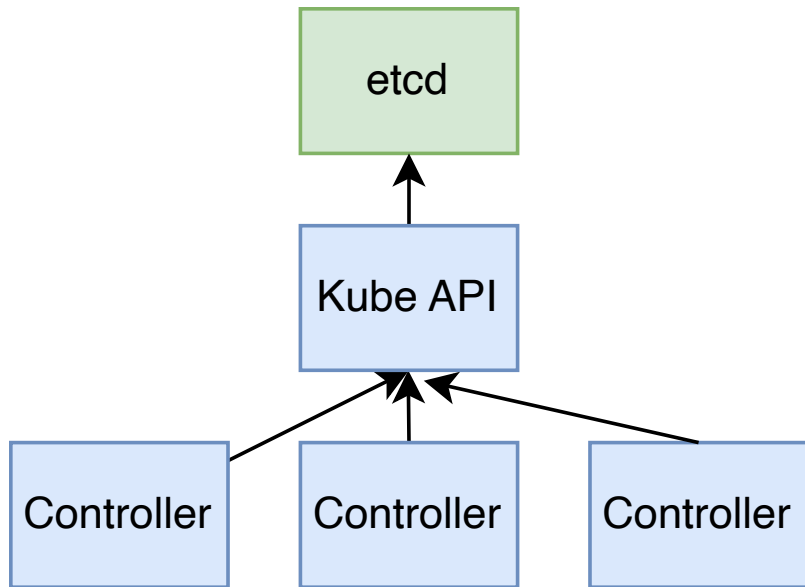
やりたいこと

Kubernetes の動作検証がしたい

- Kubernetes: コンテナをいい感じに管理するソフトウェア
- Kubernetes の設定はミスしやすい
 - いろんなプラグインの設定が関わり合って駆動する
 - ネットワーク、カーネル、etc

やりたいこと

- しかもミスに気づきにくい
 - 複数 Controller が非決定的な順序で reconcile 処理を行う
- クラスタが意図した動作をしているか検証する仕組みがほしい



どうするか？

- 静的検証
 - Controller をモデル化 [Liu et al., 2024]
- 動的検証
 - 時間オートマトン

時間オートマトンを用いた実行時モニタリング

- イベントが発生した時刻・出力値が意図通りかを時間オートマトンやその拡張を用いて検証する
- 例： $(A(B)\%(1,20))\$$ (AB 間が 1s 以上)

```
A 0.5  
B 0.8  
C 1.5  
A 2.0  
B 3.2  
A 3.5  
C 4.6
```

実際のログで試してみた (monaa)

- ログの内容を 1 文字の char に対応づける前処理

```
mappings = [  
    ['no new tags found, next scan in 1m0s', 'N'],  
    [r"^Latest image tag for (略) resolved to main-\S+$", 'U']  
]
```

```
U 1746447443.495  
N 1746447444.1  
U 1746447444.108  
U 1746447444.116  
U 1746447504.121  
N 1746447504.74  
U 1746447504.755
```

実アプリケーションの動作検証

やりたいこと

アプリの Image を更新するサービスの検証

1. Image をレジストリに push
2. 最新の Image が変わったことを検知

1 と 2 の間が 5 分以内であることを保証したい

= 1 と 2 の間が 5 分以内になっていない場合に検出したい

やりたいこと - 例

```
// 1のログ
{
  "time": "2025-07-02T02:50:46.42462649Z",
  "package_name": "auth-frontend",
  "package_tag": "stg-9c8f5e28c2c7d78da2648f5eaa62216038cbd1fd-1458"
  ....
}
```

```
// 2のログ
{
  "level": "info",
  "ts": "2025-07-03T07:06:59.990Z",
  "msg": "Latest image tag for
         ghcr.io/piny940/auth-frontend resolved to
         stg-9c8f5e28c2c7d78da2648f5eaa62216038cbd1fd-1458 ...",
  ...
}
```

monaa だと難しい理由

- package 名・タグ名ごとに時間制約を確認したい

```
create auth:1.0.1  
fetch auth:1.0.1  
create auth:1.0.2  
fetch auth:1.0.2
```

package 名・タグ名を変数に格納する必要がある

monaa は

- 変数を保持できない
- char 型以外のログを扱えない

SyMon

- 変数が保持できる

```
var {  
  # We define a string parameter to represent the current ID.  
  current_name: string;  
  current_tag: string;  
}
```

- 変数の値と一致するログだけマッチさせられる

```
fetch(name, tag | name == current_name && tag == current_tag)
```

前処理

- create: 作成、fetch: 検出
- package 名、タグ名のみ抽出
- timestamp を UNIX 時間の MOD に処理

```
create auth-backend stg-7c03f5241c93d6e77bb132d8ea9ffe9e59e7b62d-1445 171982
fetch auth-example stg-379cca639565f93fe2485c6f443b1d5b45285534-1441 172084
fetch auth-example stg-379cca639565f93fe2485c6f443b1d5b45285534-1441 172085
create auth-frontend stg-7c03f5241c93d6e77bb132d8ea9ffe9e59e7b62d-1445 172140
fetch auth-frontend stg-7c03f5241c93d6e77bb132d8ea9ffe9e59e7b62d-1445 172146
```

```
// 1のログ
{
  "time": "2025-07-02T02:50:46.42462649Z",
  "package_name": "auth-frontend",
  "package_tag": "stg-9c8f5e28c2c7d78da2648f5eaa62216038cbd1fd-1458"
  ....
}
```

```
// 2のログ
{
  "level": "info",
  "ts": "2025-07-03T07:06:59.990Z",
  "msg": "Latest image tag for
         ghcr.io/piny940/auth-frontend resolved to
         stg-9c8f5e28c2c7d78da2648f5eaa62216038cbd1fd-1458 ...",
  ...
}
```

実際に書いた SyMon ファイル (一部)

```
expr correct {
  create(name, tag | name == current_name && tag == current_tag);
  within (<400) {
    (ignore_irrelevant
      || create(name, tag | name == current_name && tag == current_tag))*;
    fetch(name, tag | name == current_name && tag == current_tag)
  };
  (ignore_irrelevant
    || fetch(name, tag | name == current_name && tag == current_tag))*
}
expr failed {
  create(name, tag | name == current_name && tag == current_tag);
  within (>300) {
    (ignore_irrelevant
      || create(name, tag | name == current_name && tag == current_tag))*;
    one_of {
      create(name, tag)
    } or {
      fetch(name, tag)
    }
  }
}
}
```

結果

過去 1 週間のログを検証

create 12 件のうち、

- 「5 分以内」という条件だと、条件を満たさないログを 5 件検出

@305024.	(time-point 9443)	x0 == auth-example	x1 == stg-x-1458 true
@305024.	(time-point 9443)	x0 == auth-frontend	x1 == stg-x-1458 true
@305051.	(time-point 9444)	x0 == auth-example	x1 == stg-x-1458 true
@305051.	(time-point 9444)	x0 == auth-frontend	x1 == stg-x-1458 true
@305053.	(time-point 9445)	x0 == auth-example	x1 == stg-x-1458 true

- 「10 分以内」だとすべて条件を満たしていた

まとめ

まとめ

- 時間オートマトンを用いてログが時間制約を満たすことを確認
- 時間制約にマッチするログを検出できた

今後は

- 検証システムの汎用性向上
- リアルタイム検出システムの構築