# 個人用に構築した情報収集基盤がオーバーエンジニアリングで技術的負債を抱える話

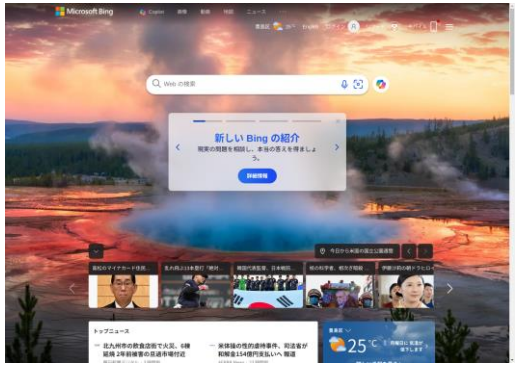
@めぐろLT #16 「みんなのオーバーエンジニアリング供養」



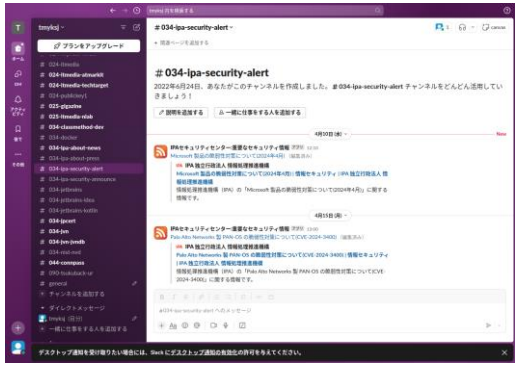
## 個人用に構築した情報収集基盤

- やりたいこと: IT 系のいろいろを情報収集したい
  - IPA
  - JPCERT/CC
  - JVN
  - iPedia
  - NVD
  - ITmedia
  - Impress
  - Docker blog
  - Spring blog
  - etc...







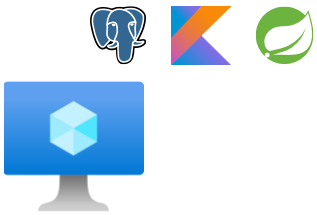



## 要件

- 収集した情報を、定期的にメールで受け取れること
- 運用コストをなるべく小さくすること



## アーキテクチャ設計



- 運用コストを小さくしたい
  → Azure Virtual Machine
  - Standard_B2ts_v2
  - vCPUs: 2, RAM: 1GiB
- データベース
  → PostgreSQL を VM 上に構成
- 言語、フレームワーク
  型安全, null-safety, ORM がほしい
  → Kotlin & Spring Framework
- バッチ処理
  → Spring Batch



## 結果

でも、世の中そんなに甘くない

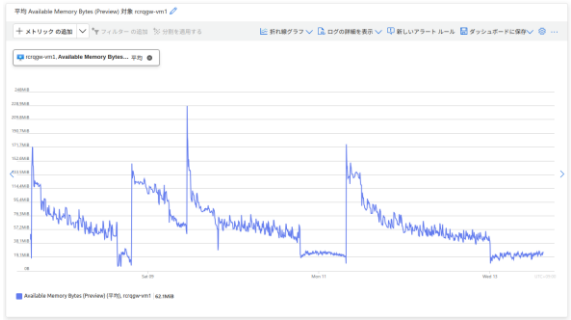


## でも、世の中そんなに甘くない

- あるとき、メールが来なくなった 🙀
- VM への ssh もできなくなった 🙀



## なぜ？

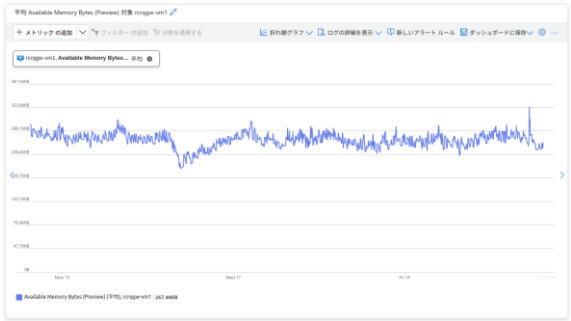




## パフォーマンス問題

- JVM, PostgreSQL が同居の時点で、それはそう
  小さいインスタンスに、全部入りのフレームワークと DBMS を同居させる選択はまずかった。
- Go などでリプレース
  - とはいえ、再実装は時間がかかりそう……
- スケールアップ
  - とはいえ、運用コストが……
- GraalVM
  - JVM アプリケーションをバイナリ化
  - 期待できそう

## GraalVM 導入結果





よかったね、で終わらない



## GraalVM

- GraalVM compiles your Java applications ahead of time into standalone binaries.
- It excludes unused classes, methods, and fields from the application binary.
  - https://www.graalvm.org/latest/docs/introduction/
- 弱点も……
  - ビルド時間が遅い
  - リフレクションを考慮できない（実行時エラーになる）



## GraalVM とビルド時間

- 小さいアプリケーションながらビルドに 5min 以上が必要 🙀
  → まぁ、仕方がない

```
BUILD SUCCESSFUL in 6m 20s
10 actionable tasks: 10 executed
12:25:16: Execution finished 'nativeCompile'.
```

## GraalVM と Reflection

- 使用しないクラス、メソッド、フィールドをバイナリから除外
  → 除外したくないクラスなどは reflect-config.json で定義
- 何か足りないときは実行時例外になるので、
  メッセージと実装を確認して reflect-config.json を更新する
- 依存するライブラリも同様 🙀
  依存関係を更新するときは、再検討する必要性あり 🙀



というのが、今のステータス



## まとめ

- 個人用に構築した情報収集基盤が、
  オーバーエンジニアリングで技術的負債を抱える話
- 全部入りのフレームワーク、データベースの選択
  → パフォーマンス問題 🙀
- GraalVM の導入
  → ビルドに時間がかかり、依存関係の更新も面倒な状態に 🙀
- **でも、でもね、 GraalVM 、実はなかなか楽しいですよ！**