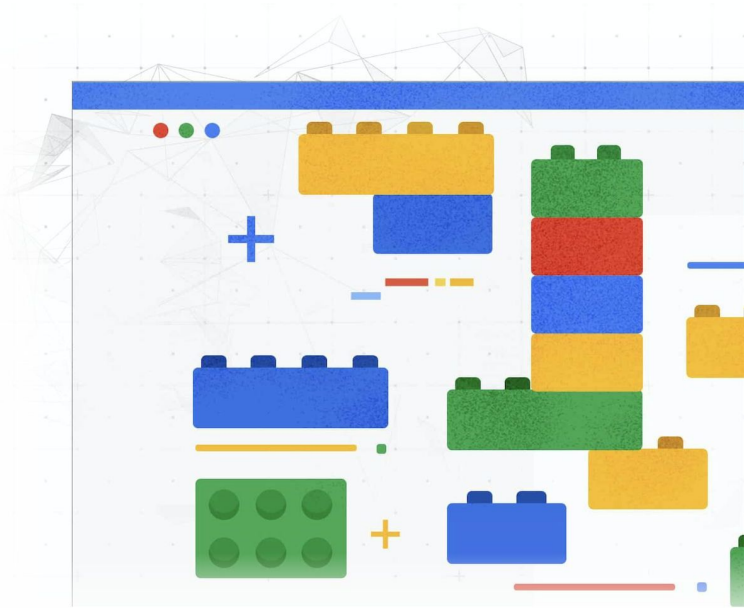
秒間 10 万リクエストを超える負荷試験環境を目指して

株式会社ビデオリサーチ
西 友規



自己紹介

Video Research Ltd.

西 友規 (Yuki Nishi)

株式会社 ビデオリサーチ
- デジタル開発部 兼 技術戦略部

経歴

- 2015 年～2019 年 SIer
  - オンプレミスからクラウドサービスまでのインフラ構築を経験
- 2019 年～現在 ビデオリサーチ
  - 開発プロセス標準化、クラウドサービス・ツールの検証・導入・管理など

クラウドサービス資格

- GCP → ACE、PCA、PDE
- AWS → SAA、SAP

LOVE
- 家族、テクノロジー

メンバー紹介

Video Research Ltd.

菊地 佑樹 (Yuki Kikuchi)

株式会社 ビデオリサーチ

- デジタル開発部 兼 技術戦略部

経歴

- 前職　2016 年～2018 年 Web 系ベンチャー
  - バックエンドエンジニア
- 2019 年～現在 ビデオリサーチ
  - 開発プロセス・クラウド運用標準化、アジャイル開発導入推進、など

クラウドサービス資格

- GCP → PCA

LOVE

- 美しい設計、美しいアーキテクチャ
- サバゲー

メンバー紹介

瀬尾 昭光 (Akimitsu Seo)

株式会社 イー・エージェンシー

- ICT パートナーソリューション部
- Solutions Architect

経歴

- 前職　アドエンジニア

Love

- アーキテクチャ全般、DevOps、DDD
- 大洗

ビデオリサーチとは

Video Research Ltd.

設　立	1962 年（昭和 37 年）9 月 20 日
資本金	2 億 2,050 万円（授権資本 3 億円）
社員数	435 名（2021 年 4 月 1 日現在）
株　主	広 告 会 社：電通、博報堂、博報堂DYメディアパートナーズ、大広 民間放送局：TBS、日本テレビ、フジテレビ、テレビ朝日、テレビ東京　他 (放送局計 23 社) 広　告　主：東芝デジタルソリューションズ　　合計　28 社
グループ会社	株式会社ビデオリサーチコムハウス、株式会社ビデオリサーチインタラクティブ、株式会社Delta Values、株式会社every sync、Muserk V.I.D., LLC

ビデオリサーチとは

Video Research Ltd.

データの収集、分析、提供、提言までを行うメジャメント企業

- 視聴率
- キャッチアップ/LIVE 配信
- Web サイト閲覧
- スマホアプリ
- アンケート調査

などなど・・・

ビデオリサーチとは

Video Research Ltd.

データの収集、分析、提供、提言までを行うメジャメント企業

- 視聴率
- キャッチアップ/LIVE 配信 ← 本日はこちらに関連したお話
- Web サイト閲覧
- スマホアプリ
- アンケート調査

などなど・・・

Agenda

- 背景
- 要件整理とプロダクト選定
- アーキテクチャ
- まとめ

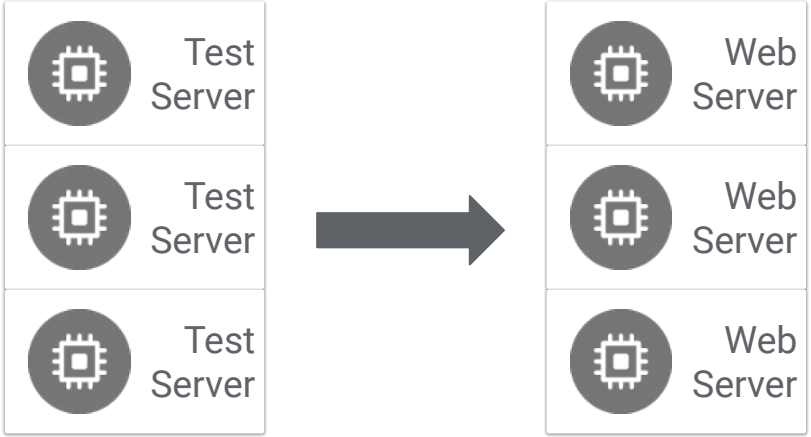
背景

01

今までの負荷試験

負荷試験を行うときにベンダーと調整し、
負荷試験環境を調達

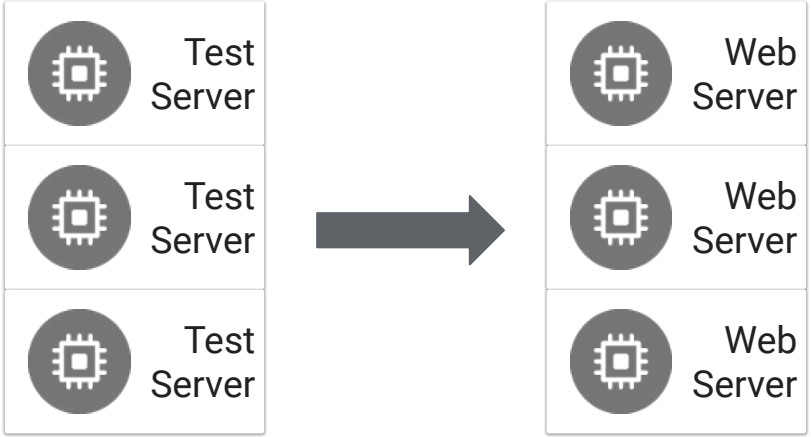
- 負荷テストは実施タイミングを調整のうえ、特定の期間しか行えない
- 負荷テスト環境調達までに時間がかかる
- 負荷テストシナリオの認識調整



今までの負荷試験

負荷試験を行うときにベンダーと調整し、
負荷試験環境を調達

- 負荷テストは実施タイミングを調整のうえ、特定の期間しか行えない
- 負荷テスト環境調達までに時間がかかる
- 負荷テストシナリオの認識調整



自分たちで管理できる負荷試験環境の構築を目指すことに！

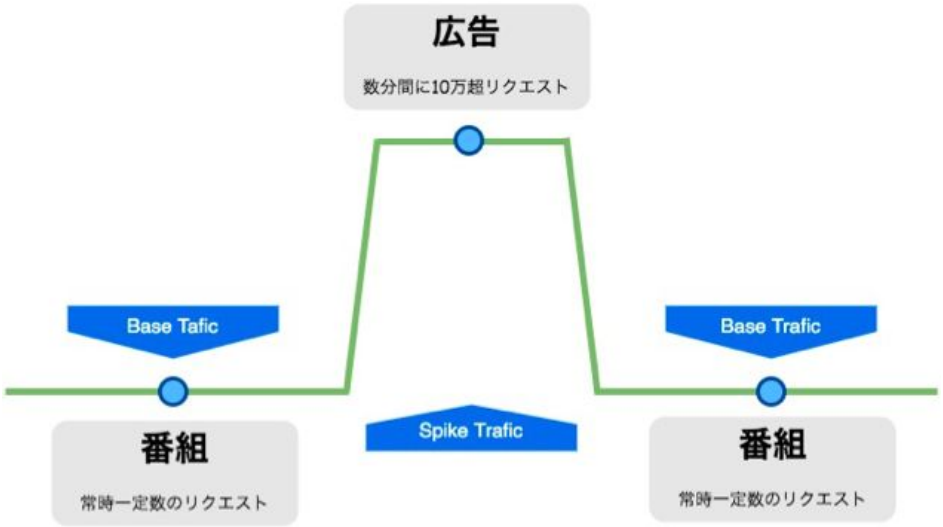
要件整理と
プロダクト選定

02

LIVE 配信のトラフィック特性

- 番組による一定数のリクエスト
- 広告による突然のスパイク

2 つの特性をシミュレートする必要がある



リクエストサンプル

https://example.com/[乱数].gif?aaa=0000&bbbb=1111

アプリケーションに求める要件

- 10 万 RPS を超えるリクエストを発生できる
- ロングランニングで実行できる
- Simple な Operation
  - 運用面やテスト実施の容易性も高めたい
  - Infrastracture のプロビジョニングや管理を軽減したい
  - 負荷テストの料金を安価におさえたい

負荷テストの要件

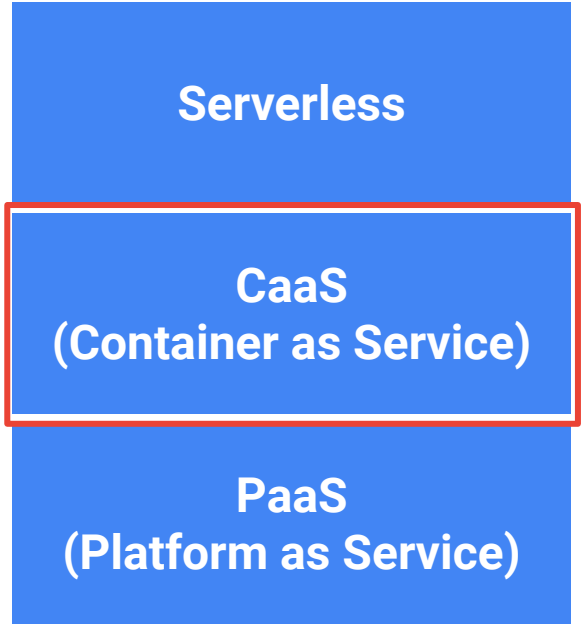
- スラムテスト(高負荷)
  広告によるトラフィックのスパイクをシミュレート

- ドリップテスト(長時間)

  番組視聴による数時間の定常的なトラフィックをシミュレート

CNCF ホワイトペーパーから選定

- クラウドネイティブモデルから要件に近いCaaS を選択



CaaS の特性

- ポータビリティ
  - コンテナによるアプリケーションの抽象化
  - Manifest によるデプロイ
- Infrastructure の抽象化
- 幅広いユースケースに対応

プロダクト選定

CaaS のフルマネージドなプロダクトで選定

- AWS Fargate
- CloudRun
- GKE Autopilot

AWS Fargate

Fargate

- リソース割当
  - 特定の組み合わせの中からユーザーがリソースを 手動選択
  - 多くのリソースを割り当て可能
  - ストレージ容量が物足りない
  - Fargate RunTask API の並列処理数が懸念
- ワークロード
  - 長時間のワークロードが実行可能
- 構成管理
  - Terraform で構築
- on Locust
  - 乱数によるリクエストのため Masterのメモリが枯渇し、
    1 万 RPS 付近で Master がヘルスチェックに応答できなくなる

AWS Fargate

CloudRun

- リソース割当
  - 特定の組み合わせの中からユーザーがリソースを 手動選択
  - リソース不足 で多数のリクエストが発生させられない
- ワークロード
  - 長時間のワークロードに懸念
- 構成管理
  - gcloud で構築
- on vegeto
  - 3 万 RPS あたりでリソース不足となる

※vegeto については、後ほど説明

Autopilot

- リソース割当
  - 適切なリソースは Autopilot が自動選択
    - デフォルトは e2-medium、リソース要求に応じて e2 シリーズのいずれのかインスタンスタイプが追加
  - 非常に多くのリソースを割り当て可能
  - 1 クラスタあたりの Pod 数 12,800
  - ストレージボリュームも豊富
- ワークロード
  - 長時間のワークロードが実行可能
- 構成管理
  - gcloud で数ステップ

Autopilot を採用

要件、運用、コストを考慮した結果、Autopilot を採用

- スラムテスト(高負荷)
  - 1 番、多くのコンピュートリソースを割り当て可能
    (クラスタ上限、vCPU、メモリ、ストレージ全て)
- ドリップテスト(長時間)
  - 長時間のワークロードが実行可能
- 運用面(Day2 operations を含む)
  - kubectl の実行でテスト実施が可能
  - CNCF の豊富なエコシステムが利用可能(今後の拡張性)
  - Pod 単位での課金、Pod への SLA

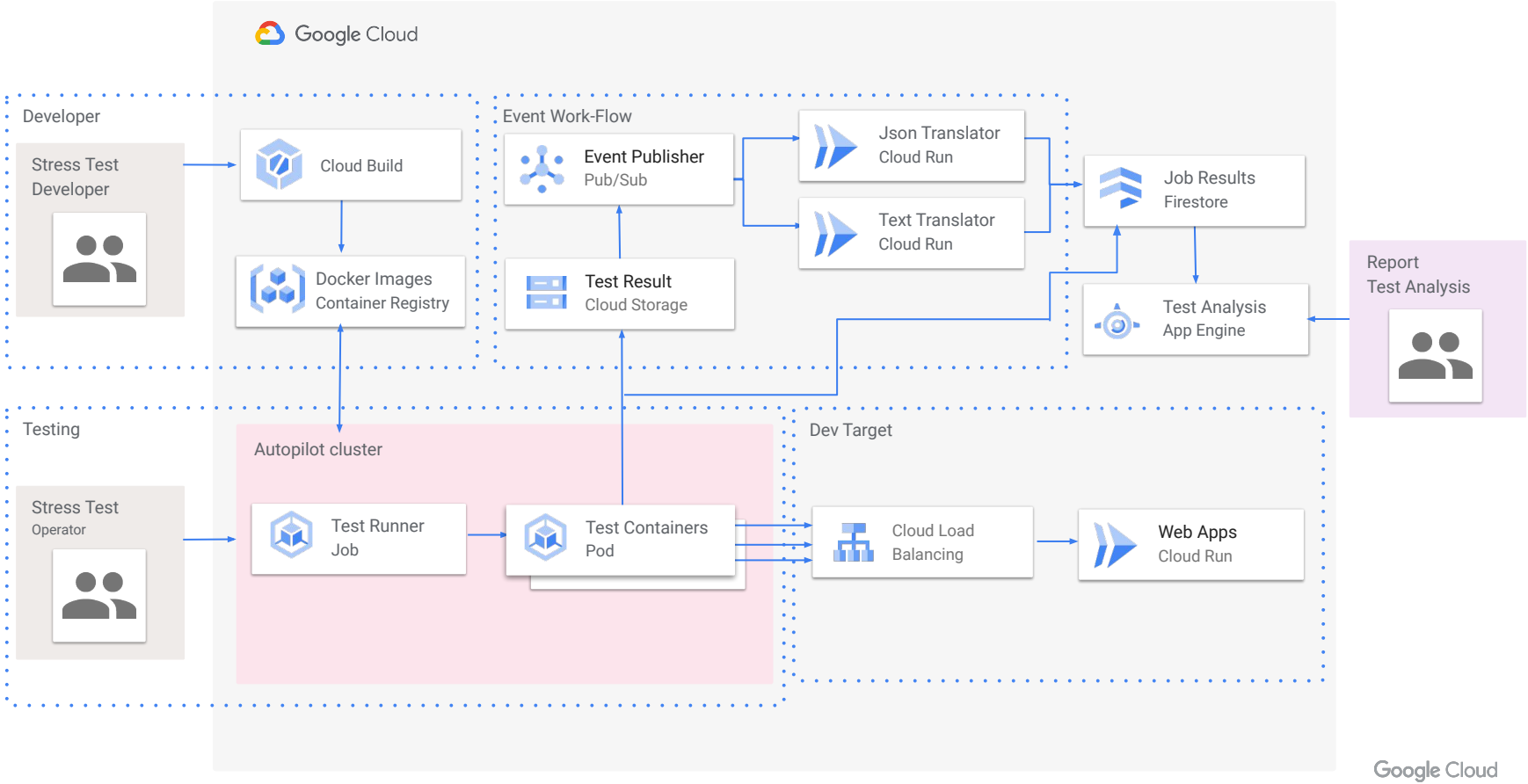
(参考) リソースの割り当て評価

要件/運用	Fargate	CloudRun	Autopilot
リソース 割り当て	・タスク数 最大 500	・コンテナ数 最大 1000 コンテナ	・Pod 数 最大 12,800 ・コンテナ数 最大 300,000
	・4vCPU ・メモリ 30GB ・ボリューム ストレージ 4GB	・4vCPU ・メモリ 8GB	・vCPU 28 個 ・メモリ 80GiB ・エフェメラル ストレージ 10GiB
料金の 構成要素	・タスク (vCPU、メモリ、 ストレージ)	・CPU ・メモリ ・リクエスト料 ・ネットワーク転送料	・Pod (vCPU、メモリ、 ストレージ) ・クラスタ

アーキテクチャ

03

Architecture

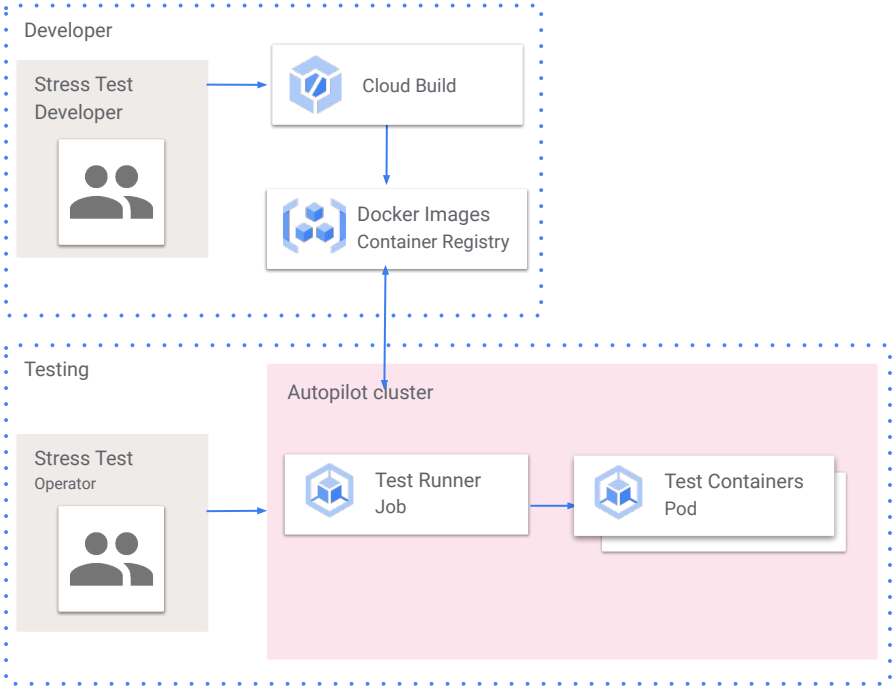


負荷テストツール

- vegeto
  - Vegeta をカスタマイズした負荷ツール
    - 乱数生成機能を追加
  - Go 製 CLI ツール
  - ライブラリとしても利用可能
  - テキストや Json など様々な形式でレポート可能
- bigbang
  - vegeto の wrapper コマンド
  - 各種 Google Cloud リソースとの連携を担う

コンテナイメージの作成

- Develover が負荷テストツールを GitHub に Push
- コンテナイメージの作成は Cloud Build により行う



負荷テストの実施

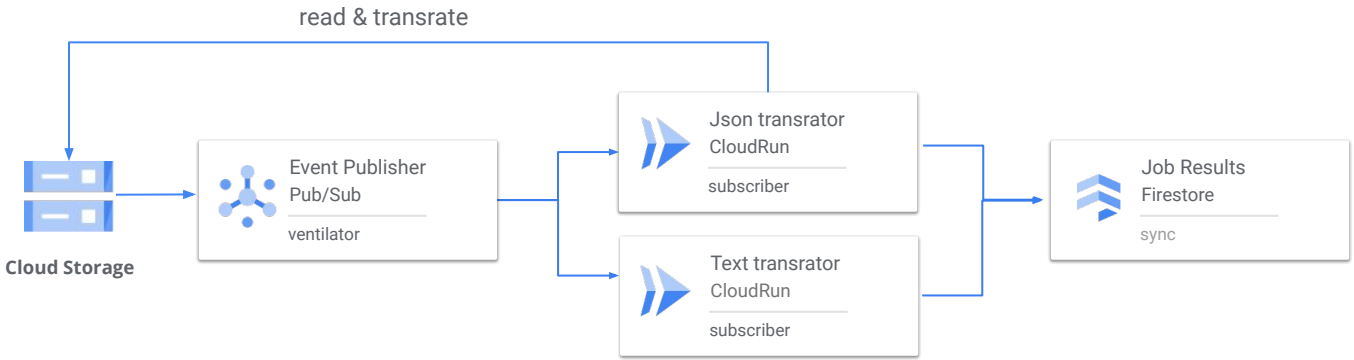
- シナリオ毎の Manifest を適応して負荷テストを実施
- parallelism により Pod を並列で実行して負荷を発生させる

```
apiVersion: batch/v1
kind: Job
metadata:
  name: stress-test-job
template:
  spec:
    containers:
      - name: bigbang
        image: gcr.io/k8s-validate/bigbang:1.1
        command:
["./bigbang","attack","--rate","10000","--duration","300s","--cpus","4",
"--job","210327", “--name”,”ad-test”]
…
```

各 Pod の
実行結果は GCS 上に
ファイルとして保存

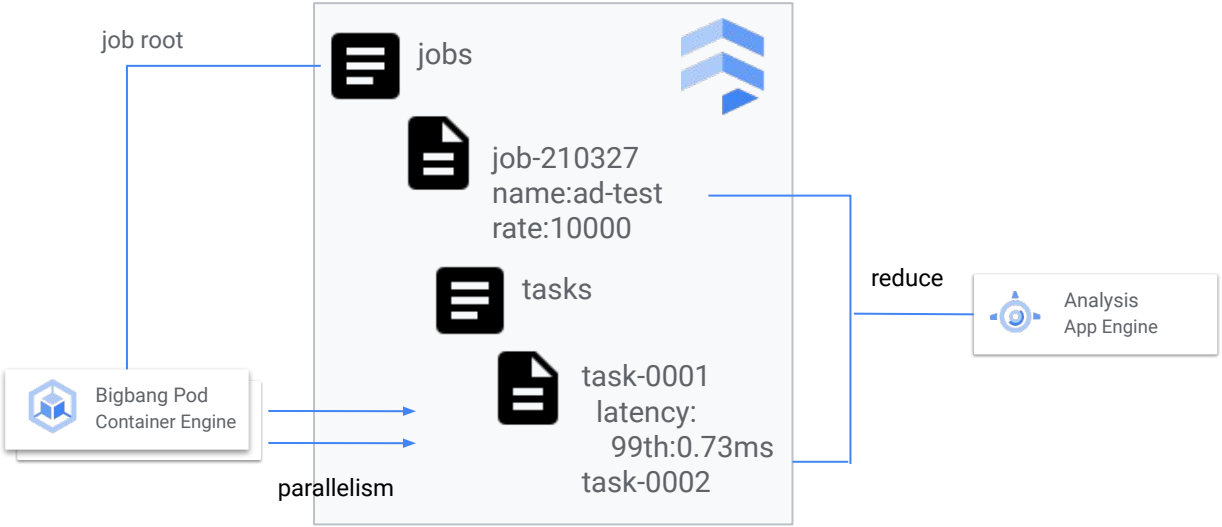
Event Work-Flow

- バイナリ形式のテスト結果ファイルを変換
  - coper pattern
  - 複数の subscriber により Json と Text 形式に変換して保存

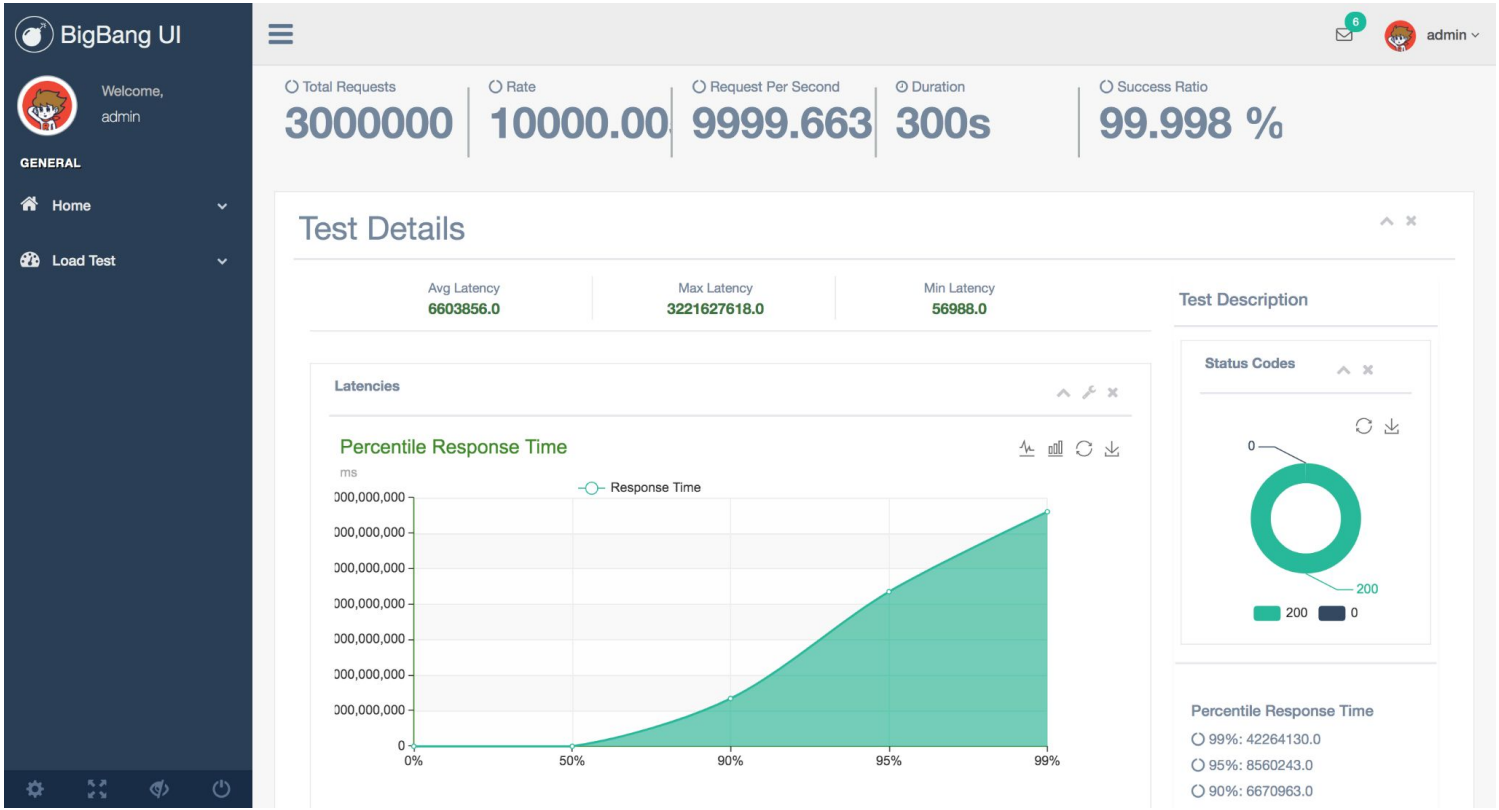


Data structure

- レポートは SubCollection として Firestore に保存
- 各タスク(Pod) の集約は UI 側のロジックに実装



Report UI (サンプル)



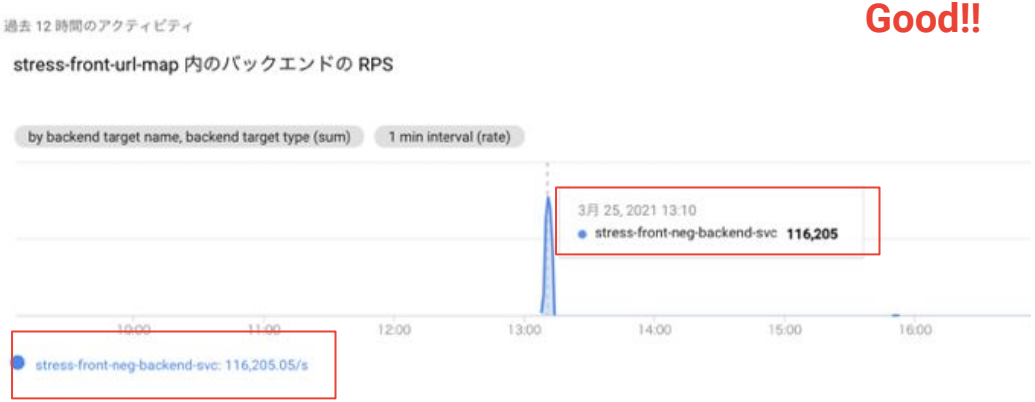
Dev target & Monitoring

- 負荷テスト基盤の開発用ターゲット
  - 負荷を受ける側も利用時のみ課金
  - 開発中の Google Cloud リソースと一緒に把握できるので非常に楽

116,205RPS

Testing

Monitoring

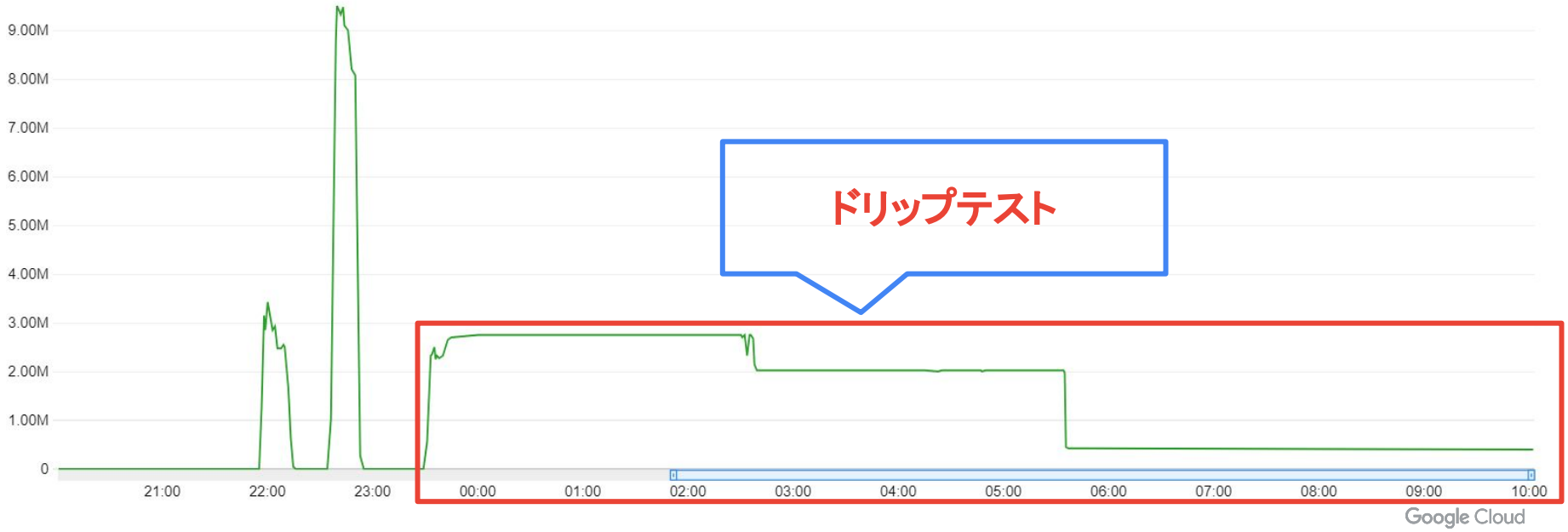


Good!!

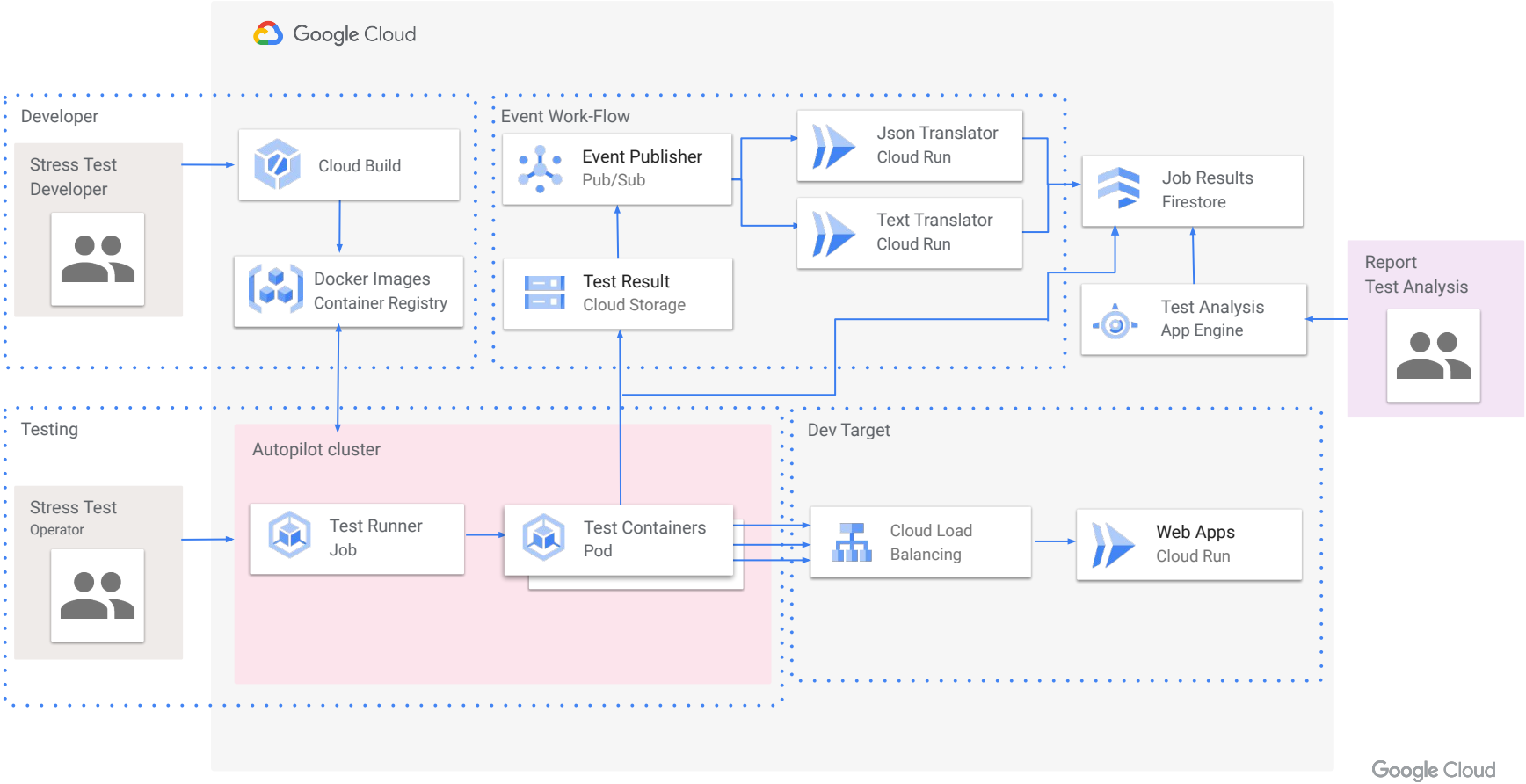
Attack!! Production!!!

9,512,978/min
≒16 万 RPS

Good!!



(再掲) Architecture



Points

- Node のカーネルチューニングはできない
- Pod のリソース制限(request=limit)
- 大規模な JOB 実行ではリソースの上限緩和が必要
  - CPUs、CPUs (all regions)、IP address など
- ttlSecondsAfterFinished(Job の自動削除)
  - 現状、Regular、Stable channel では効かない
- Cloud Logging の料金に注意
  - Autopilot のログも他の Google Cloud サービスと同じくCloud Logging に出力するため、Pod の料金よりも高くなることがある。

まとめ

04

まとめ

- GKE Autopilot を活用することにより、要件を満たした負荷手試験環境を用意し、必要なときにすぐに負荷テストができることに！
  - アプリケーション要件
    - 10 万 RPS を超えるリクエスト
    - テスト実施の容易性
    - 環境プロビジョニングおよび管理の容易性
    - 料金の最適化
  - 負荷テスト要件
    - スラムテスト(スパイク負荷シミュレート)
    - ドリップテスト(長時間の継続的な負荷シミュレート)

今後の予定と要望

- 今後の予定
  - 負荷試験ツール以外の、多くのリソースが必要なバッチ処理を動かしてみる
  - Report UI の改良
    - Report だけでなく負荷試験の実行もできるように

- Autopilot への要望
  - リクエストの受け側としても使いたい
  - GPU インスタンス
  - 各種 Google Cloud リソースへのアクセスを担う
  - ttlSecondsAfterFinished(Job の自動削除)
    - Rapid channel で 1.21.1-gke.1800 にアップグレードすれば使えるようなので、Regular、Stable でも早く使いたい

リクルーティング

求む Z 戦士よ！！

https://www.green-japan.com/company/6885

各社リクルートサイト
https://hrmos.co/pages/videor/jobs
https://www.r-agent.com/kensaku/companydetail/03646/

Thank you