

# Cloud Operations で Google Cloud の運用を始めよう！

INSIDE Games and Apps

Google Cloud カスタマーエンジニア  
TT 🦊 Tsukasa / たまるつかさ

# システムの全体を知るのは難しい

複数の部品が組み合わさっている

- インフラストラクチャ
- アプリケーション
- サービス

大量の情報をわかりやすく管理する必要

メトリクス、ログなど性質の違うデータを収集



# 運用の際に必要なことはなにか

## システムの状態

### メトリクス

アプリケーション  
サービス  
プラットフォーム  
マイクロサービス

### ログ

アプリケーション  
サービス  
プラットフォーム

### トレース

アプリケーション



## 可視化と分析

### ダッシュボード

メトリクスエクスプローラ

ログビューア

サービスモニタリング

ヘルスチェック

デバッガー

プロファイラー



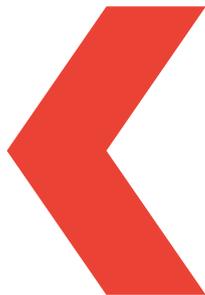
## トラブルシューティング

## 問題の管理

アラート

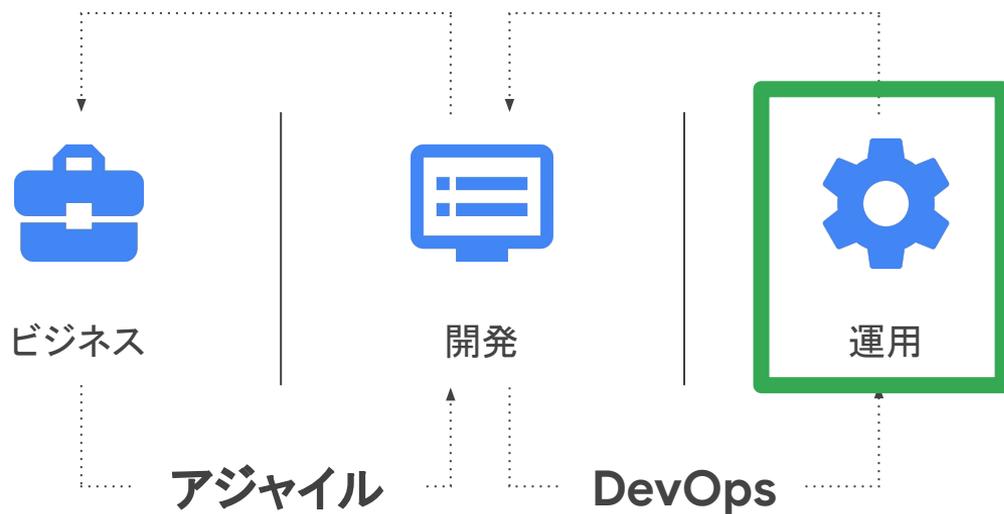
エラーレポート

SLO



# Google Cloud Operations とは

システムの **運用** で利用する様々な機能が揃った製品群



## (参考) Stackdriver からの名称変更



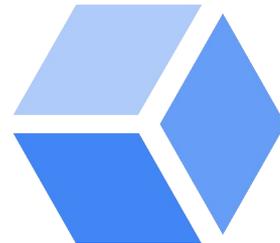
Stackdriver

2012年に創設された SaaS 型の  
クラウドインフラ監視サービス



Google Stackdriver

2014年5月に Google が買収  
Google Stackdriver にブランド  
変更され、2016年10月に  
一般提供が開始



**Cloud Operations**

2020年に Stackdriver  
の名が外れました

# Operations Management Observability at scale



## Logging

プラットフォーム、アプリケーション、サービスからのログを収集

- ログの検索・閲覧・フィルタリング
- エラーレポーティング & ダッシュボード
- ログメトリクス
- ログルーターでログをエクスポート



## Monitoring

プラットフォーム、アプリケーション、サービス、マイクロサービスからのメトリクスを監視

- ダッシュボード
- メトリクスエクスペローラ & カスタムメトリクス
- 稼働時間チェック
- サービスモニタリング
- アラート管理



Google Cloud のオブザーバビリティを支えるプラットフォームと同じ場所で動作

運用のスケールについて心配不要



Google Cloud のすべてのサービスで設定無しでビルトインで利用できる



Google の SRE 原則に基づいています



ロギング、メトリクスプロバイダーと連携

- Dynatrace、Datadog、NewRelic などのパートナーサービスをサポート



新しい Google Cloud のサービスにも最初から対応



# Cloud Logging



- Google Cloud のすべてのログを集約する SaaS
- 収集されるログには、以下のようなものがある
  - Google Cloud の操作ログ
  - データアクセスのログ
  - サービス固有のログ
    - HTTP(S) ロードバランサのログ
    - Cloud SQL の slow query ログ
    - ...
- 例) GKE の場合、以下のようなログが自動的に収集される
  - コンテナの標準出力・標準エラー出力 (アプリケーションログ)
  - リソース使用状況 (CPU、メモリ...)
  - GKE の Cluster や Node のログ

# Logs Explorer

Filter by label or text search

GCE VM Instance All logs Any log level Last hour Jump to now

Showing logs from the beginning of time to 5:21 AM (JST) Download logs View Options

2018-09-14 05:20:57.000 JST	I0913 20:20:57.761662 1417	server.go:796	GET /healthz: (25.699µs) 200	[[curl/7.60.0] 127.0.0.1:45338]	⋮
2018-09-14 05:21:03.000 JST	I0913 20:21:03.115780 1352	server.go:796	GET /healthz: (24.728µs) 200	[[Go-http-client/1.1] 127.0.0.1...]	⋮
2018-09-14 05:21:05.000 JST	I0913 20:21:05.093409 1416	server.go:796	GET /stats/summary/: (68.38669ms) 200	[[Go-http-client/1.1] ...]	⋮
2018-09-14 05:21:05.000 JST	I0913 20:21:05.147656 1416	server.go:796	GET /stats/summary/: (24.373237ms) 200	[[Go-http-client/1.1]...]	⋮
2018-09-14 05:21:05.000 JST	I0913 20:21:05.452918 1416	server.go:796	GET /healthz: (21.668µs) 200	[[curl/7.60.0] 127.0.0.1:42254]	⋮
2018-09-14 05:21:05.000 JST	I0913 20:21:05.081983 1417	server.go:796	GET /stats/summary/: (62.179238ms) 200	[[Go-http-client/1.1]...]	⋮
2018-09-14 05:21:05.000 JST	I0913 20:21:05.131937 1417	server.go:796	GET /stats/summary/: (13.819337ms) 200	[[Go-http-client/1.1]...]	⋮
2018-09-14 05:21:05.000 JST	I0913 20:21:05.014560 1352	server.go:796	GET /stats/summary/: (9.497866ms) 200	[[Go-http-client/1.1] ...]	⋮
2018-09-14 05:21:05.000 JST	I0913 20:21:05.127077 1352	server.go:796	GET /stats/summary/: (9.283535ms) 200	[[Go-http-client/1.1] ...]	⋮
2018-09-14 05:21:07.000 JST	I0913 20:21:07.774442 1417	server.go:796	GET /healthz: (41.087µs) 200	[[curl/7.60.0] 127.0.0.1:45366]	⋮
2018-09-14 05:21:07.000 JST	I0913 20:21:07.180493 1352	server.go:796	GET /healthz: (42.967µs) 200	[[curl/7.60.0] 127.0.0.1:54552]	⋮
2018-09-14 05:21:09.000 JST	I0913 20:21:09.117532 1352	server.go:796	GET /healthz: (24.131µs) 200	[[Go-http-client/1.1] 127.0.0.1...]	⋮
2018-09-14 05:21:15.000 JST	I0913 20:21:15.115898 1417	server.go:796	GET /healthz: (25.98µs) 200	[[Go-http-client/1.1] 127.0.0.1...]	⋮
2018-09-14 05:21:15.000 JST	I0913 20:21:15.471084 1416	server.go:796	GET /healthz: (21.927µs) 200	[[curl/7.60.0] 127.0.0.1:42292]	⋮
2018-09-14 05:21:17.000 JST	I0913 20:21:17.787034 1417	server.go:796	GET /healthz: (61.519µs) 200	[[curl/7.60.0] 127.0.0.1:45400]	⋮
2018-09-14 05:21:17.000 JST	I0913 20:21:17.190200 1352	server.go:796	GET /healthz: (26.312µs) 200	[[curl/7.60.0] 127.0.0.1:54556]	⋮
2018-09-14 05:21:21.000 JST	I0913 20:21:21.115775 1416	server.go:796	GET /healthz: (36.871µs) 200	[[Go-http-client/1.1] 127.0.0.1...]	⋮
2018-09-14 05:21:25.000 JST	I0913 20:21:25.498912 1416	server.go:796	GET /healthz: (42.454µs) 200	[[curl/7.60.0] 127.0.0.1:42334]	⋮
2018-09-14 05:21:27.000 JST	I0913 20:21:27.800392 1417	server.go:796	GET /healthz: (27.443µs) 200	[[curl/7.60.0] 127.0.0.1:45432]	⋮
2018-09-14 05:21:27.000 JST	I0913 20:21:27.116012 1352	server.go:796	GET /healthz: (26.668µs) 200	[[Go-http-client/1.1] 127.0.0.1...]	⋮
2018-09-14 05:21:27.000 JST	I0913 20:21:27.199777 1352	server.go:796	GET /healthz: (41.098µs) 200	[[curl/7.60.0] 127.0.0.1:54562]	⋮
2018-09-14 05:21:33.000 JST	I0913 20:21:33.116218 1352	server.go:796	GET /healthz: (55.284µs) 200	[[Go-http-client/1.1] 127.0.0.1...]	⋮

No newer entries found matching current filter. Load newer logs

- ログを Web 画面で確認できる
- ログの種類、時間、キーワード等でフィルタリングも可能

# ログの種類と保持期間

<https://cloud.google.com/logging/docs/storage?hl=ja#custom-retention>

ログタイプ	任意	デフォルトの保持期間
管理アクティビティ監査ログ	不可	400 日
データアクセス監査ログ	可	30 日
システム イベント 監査ログ	不可	400 日
アクセスの透明性ログ	不可	400 日
その他のすべてのログ	可	30 日

- 出力任意が可のものについて、保持期間を最大 3650 日まで伸ばすことが可能
- すべてのログについて別のサービス (GCS や BigQuery など) にエクスポートすることが可能 (永続化の対応手段)

# Cloud Monitoring



- システム全体のモニタリングツール
  - Google Cloud と AWS をモニタリング可能
- 主な機能
  - 様々な指標を自動で収集、表示
    - CPU、メモリ、I/O、ネットワーク等
  - ダッシュボード
  - サービスのモニタリング
  - 死活監視(稼働時間チェック)
  - アラート通知

# Metrics Explorer

モニタリングする指標  
(メトリクス)を効率的に探すための  
手段 Metrics Explorer を提供

頻繁に利用される Metrics を推奨  
指標として提示 (based on learning  
from Google Cloud wide usage)

**Google Cloud Platform** Monitoring

**Metrics explorer**

**Metric** | **VIEW OPTIONS**

Line | 1H 6H 1D 1W 1M 6W CUSTOM Save Chart

**Build Your Query** | Query Editor

Find resource type and metric

Resource type: **GCE VM Instance**

Select a metric

Popular Metrics

- CPU usage: compute.googleapis.com/instance/cpu\_usage\_time
- CPU utilization: compute.googleapis.com/instance/cpu\_utilization
- Disk read bytes: compute.googleapis.com/instance/disk/read\_bytes...
- Disk write bytes: compute.googleapis.com/instance/disk/write\_bytes...

**Metric: compute.googleapis.com/instance/cpu\_usage\_time**

Description: Delta vCPU usage for all vCPUs, in vCPU-seconds. To compute the per-vCPU utilization fraction, divide this value by (end-start)/N, where end and start define this value's time interval and N is...

Unit: s Kind: Delta Value type: Double

Select a metric to start

**Metric** | **VIEW OPTIONS**

**Build Your Query** | Query Editor

Find resource type and metric

Resource type: **GKE Container**

Metric: **Memory usage**

Filter

+ Add a filter

Group By

+ Add a label

Aggregator

none

SHOW ADVANCED OPTIONS

+ ADD METRIC

Stacked Area | 1H 6H 1D 1W 1M 6W CUSTOM Save Chart

Too many lines? For a better visualization, turn on outlier mode or use mean aggregation

3 hr interval (mean)

Mar 14, 2020 3:30 AM

8 lines above

- kubelet-1473811471444491053non-... 64.4MB
- pods-1473811471444491053service... 37.5MB
- pods-1473811471444491053non-... 137MB
- docker-daemon-gke-nginx-cluster-default-pool-82cf3845-cm3serviceable 246MB
- docker-daemon-gke-nginx-cluster-default-pool-82cf3845-cm3non-... 106MB

287 lines below

pod_id	container_name	Name (from InstanceID)	Value
1be82e7-6213-11ea-a646-42010a8e012c	fluentd-gcp		133.051MB
1be82e7-6213-11ea-a646-42010a8e012c	prometheus-to-sd-exporter		15.948MB
1be82e7-6213-11ea-a646-42010a8e012c			16.105MB
1be82e7-6213-11ea-a646-42010a8e012c			140.480MB

# ダッシュボード機能

よく使うメトリクスをまとめて表示、カスタマイズ可能な  
ダッシュボードとフィルタ機能

The screenshot displays the 'All Dashboards' page in Google Cloud. At the top, there are filter buttons for 'Type', 'Environment', 'Owner', and 'Lorempsem'. Below these are 'Quick Filters' for 'Starred', 'Firing Alert', 'Created by me', and 'Automated'. A search bar is located on the right. The main content is a table of dashboards with columns for Name, Firing Alert, Owner, Type, and Last Accessed. A red circle '1' highlights the filter buttons, and a red circle '2' highlights the 'Type' column dropdown menu.

Name ↓	Firing Alert ↓	Owner ↓	Type	Last Accessed ↓
<input type="checkbox"/> crasher	No	rshalom	Service Automated	Today ☆ ⋮
<input type="checkbox"/> default	No	lkrotowski	SLO	4 hours ago ☆ ⋮
<input type="checkbox"/> loader	🚨 2 alerts firing	kconway	Alerting	4 hours ago ☆ ⋮
<input type="checkbox"/> sleeper	No	rbergman	Resource Automated	yesterday ☆ ⋮
<input type="checkbox"/> system health	No	kconway	SLO	3 days ago ☆ ⋮
<input type="checkbox"/> latency	No	rbergman	Service	1 week ago ☆ ⋮
<input type="checkbox"/> log-based metrics	⚠️ 1 warning	kconway	Alerting	1 week ago ☆ ⋮
<input type="checkbox"/> loremp-ipsem	No	jpodraza	Service Automated	2 weeks ago ☆ ⋮

# 稼働時間チェック

ユーザー視点での外部からのアクセスがどうなっているかをチェックします。

API または サービスのエンドポイントに定期的にアクセスし、レイテンシとサービスの可用性をチェックします。

期待した性能が出ていない場合はアラートを設定できます。

The image shows a composite of three screenshots from the Google Cloud Platform Monitoring console. The leftmost screenshot displays the 'Uptime details' page for a check named 'Super awesome', showing a 100% uptime percentage and a 'Passed Checks' section. The middle screenshot shows the 'Edit uptime check' configuration form, where fields like 'Title', 'Check Type', 'URL', 'Hostname', 'Path', and 'Check every' are visible. A 'Log check failures' checkbox is checked, and a tooltip explains that failures will be saved to Cloud Logging. The rightmost screenshot shows the 'Current Status' section, which lists the check's performance across various regions: North America (usa-virginia, usa-oregon, usa-iowa), Europe (eur-belgium), Asia Pacific (apac-singapore), and South America (sa-brasil-sao\_paulo), all with green status indicators. Below this, the 'Configuration' section lists details such as Check ID, Title, Check Type, Hostname, Path, Port, Check Every, Timeout, and Regions.

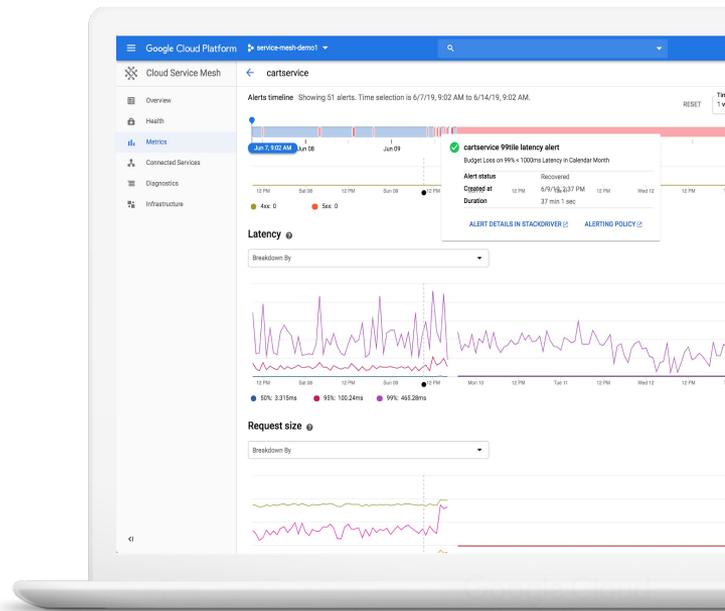
# SLO モニタリング

SLO モニタリングは、サービスレベル目標(SLO)のパフォーマンスに関するアラートポリシーを設定するためのツールを提供して、Google Cloud マイクロサービスの健全性をモニタリングするのに役立ちます。

ユーザー視点での外部からのアクセスがどうなっているかをチェックします。

API またはサービスのエンドポイントに定期的にアクセスし、レイテンシとサービスの可用性をチェックします。

期待した性能が出ていない場合はアラートを設定できます。



# アラート通知の手段

事前に定義した通報定義に従い、様々な手段で発報することが可能

## アラート発報定義の例

- Metric Threshold
- Metrics Absence
- Metric Rate of Change
- Group Aggregate Threshold
- Process Health
- Uptime Check Health



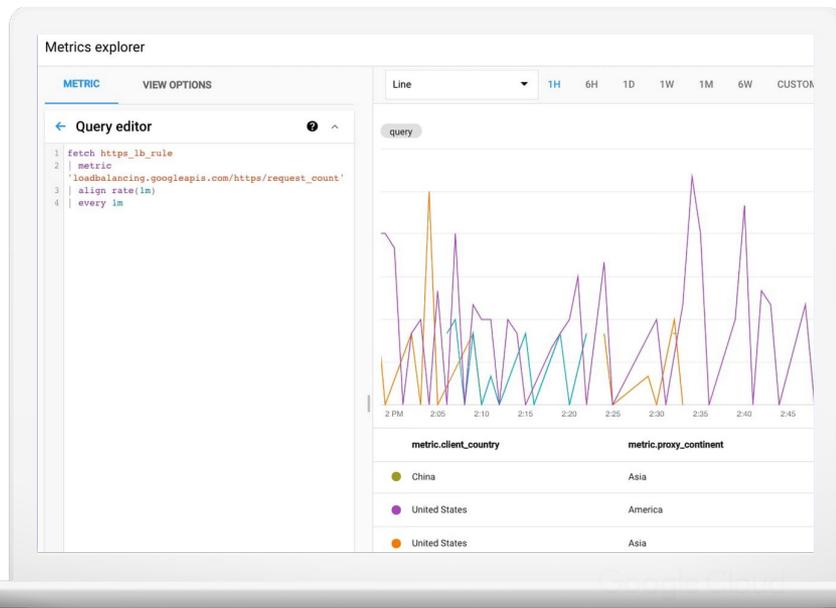
## 通知手段

- Email
- Cloud Mobile App
- PagerDuty
- SMS notifications
- Slack
- Webhooks

# Monitoring Query Language (MQL)

モニタリングのためのクエリ言語

- Google 社内の本番環境で使用されている高度なクエリを可能にする言語を、Google Cloud ユーザーも利用できるように
- Metrics Explorer から利用可能
- 強力な指標クエリ、分析、グラフ作成、アラートの機能
- あらゆるユースケースで利用可能で、モニタリングをより高度なものに



# Cloud APM

## (Application Performance Management)

### Debugger

スナップショット、ブレークポイント、および動的ログステートメントを提供

根本原因分析の反復サイクルを短縮

### Profiler

アプリケーション全体で実行される CPU 集約型またはメモリ集約型の関数のパフォーマンスを継続的に分析

### Trace

分析するための分散トレース機能を提供

パフォーマンスのボトルネックを見つけるのに役立ちます

# Cloud Debugger

## 継続的な本番デバッグ

- リアルタイムのデバッグ
- デバッグセッションを共有することによるコラボレーション
- スナップショット、ブレークポイント、条件付きデバッグ
- IDE とのインテグレーション
- バージョン管理

The screenshot displays the Google Cloud Platform Stackdriver Debugger interface for a service named 'sockshop - unversioned'. The left pane shows the source code for 'index.js', which includes a REST API endpoint for adding items to a cart. The right pane shows the debugger's control panel with options to take a snapshot, set a condition, or set an expression. Below the control panel, there are instructions on how to use the debugger without stopping the application.

On the right side of the screenshot, a portion of a web application is visible, showing a product page for 'Crossed' socks. The page includes a 'SALE' badge, a price of '\$17.32', and buttons for 'Add to cart' and 'Add to wishlist'. The product details section describes it as a 'mature sock, crossed, with an air of nonchalance' and lists material and care instructions.

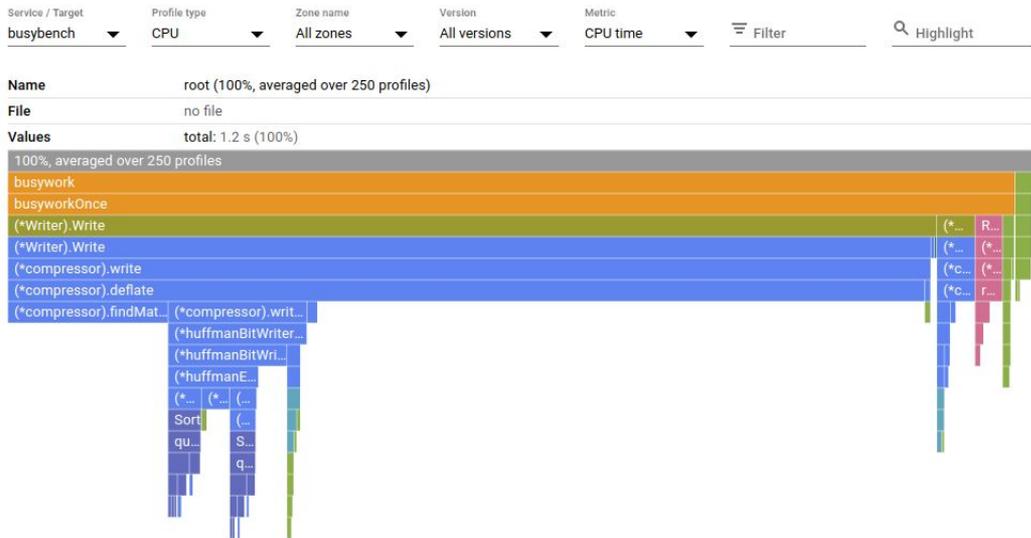
# Cloud Profiler

## 継続的な CPU とヒープ プロファイリング

- 影響の少ない **本番環境プロファイル**
- Google Cloud ワイドで利用可能 ( VMs、GAE、GCE、GKE )
- **Java、Go、Python、NodeJS** をサポート
- アプリケーションの関数コールパターンを理解
- パフォーマンスを向上させ、コストを削減

### Stackdriver Profiler

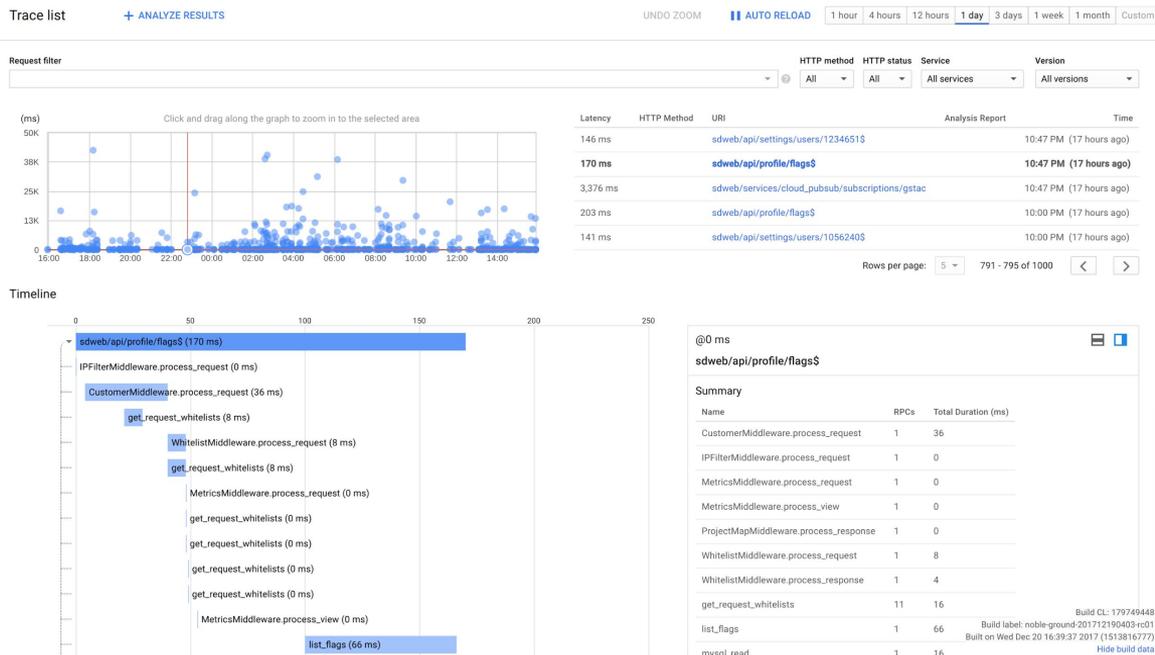
10 MINUTES 30 MINUTES 1 HOUR 4 HOURS 12 HOURS 1 DAY 3 DAYS 7 DAYS 30 DAYS



# Cloud Trace

## レイテンシデータを収集する分散トレースシステム

- App Engine と統合
- GCE と GKE をサポート
- ほぼリアルタイムの  
パフォーマンス分析
- 詳細なレイテンシレポート
- パフォーマンスのボトルネック  
を発見
- 自動課題検出とアラート



# Cloud Error Reporting

## エラーの集中管理

- App Engine と統合
- GCE と GKE をサポート
- 強力なエラーとスタックトレース検出
- 詳細なエラーレポート
- アプリの問題やバグを発見
- 自動課題検出とアラート

The screenshot displays the Stackdriver Error Reporting interface. At the top, it shows the Stackdriver logo, the text 'Stackdriver Error Reporting', and navigation options: 'All services', 'All versions', and a filter dropdown set to 'Acknowledged, Resolved, Muted, Open'. Below this is a search bar labeled 'Filter errors' and a '1 hr' refresh button.

The main section is titled 'Errors in the last 30 days' and contains a table with the following columns: 'Resolution Status', 'Occurrences', 'Error', and 'Seen in'. The table lists two error types:

Resolution Status	Occurrences	Error	Seen in
Resolved	20,690	<b>NEW</b> PermissionDenied: 403 The caller does not have permission raise_from (/usr/lib/python2.7/dist-packages/six.py)	gke_instances
Open	76	<b>NEW</b> ServiceUnavailable: 503 Getting metadata from plugin failed with error raise_from (/usr/lib/python2.7/dist-packages/six.py)	gke_instances

Below the table, there is a 'Stack trace sample' section with tabs for 'Parsed' and 'Raw'. The 'Raw' tab is selected, showing the following stack trace:

```
PermissionDenied: 403 The caller does not have permission
at raise_from (/usr/lib/python2.7/dist-packages/six.py:737)
at error_remapped_callable (/usr/local/lib/python2.7/dist-packages/google/api_core/grpc_helpers.py:56)
at __call__ (/usr/local/lib/python2.7/dist-packages/google/api_core/gapic_v1/method.py:139)
at batch_write_spans (/usr/local/lib/python2.7/dist-packages/google/cloud/trace_v2/gapic/trace_service_client.py:18
```

Thank you!