B2C デジタルサービス向け統合 API 基盤「 RAFTEL 」のセキュリティ強化策

株式会社 NTTドコモ

加藤 雅俊

自己紹介

加藤 雅俊　/　Masatoshi KATO

所属

- ✓ NTTドコモ　サービスデザイン部

担当業務

- ✓ グランドデザイン担当：部内の開発統括、技術戦略策定 等
- ✓ アプリケーション開発担当：サービス向け API 基盤（RAFTEL）の開発・運用

◆ 2018 年から 2023 年 6 月まで　コンシューマー向けサービスの開発・運用を担当

アジェンダ

- ✓ 会社紹介
- ✓ RAFTEL 紹介
- ✓ RAFTEL リニューアル
- ✓ DevSecOps 実現に向けた取り組み
- ✓ Sysdig の活用
- ✓ 今後の取り組み

会社概要

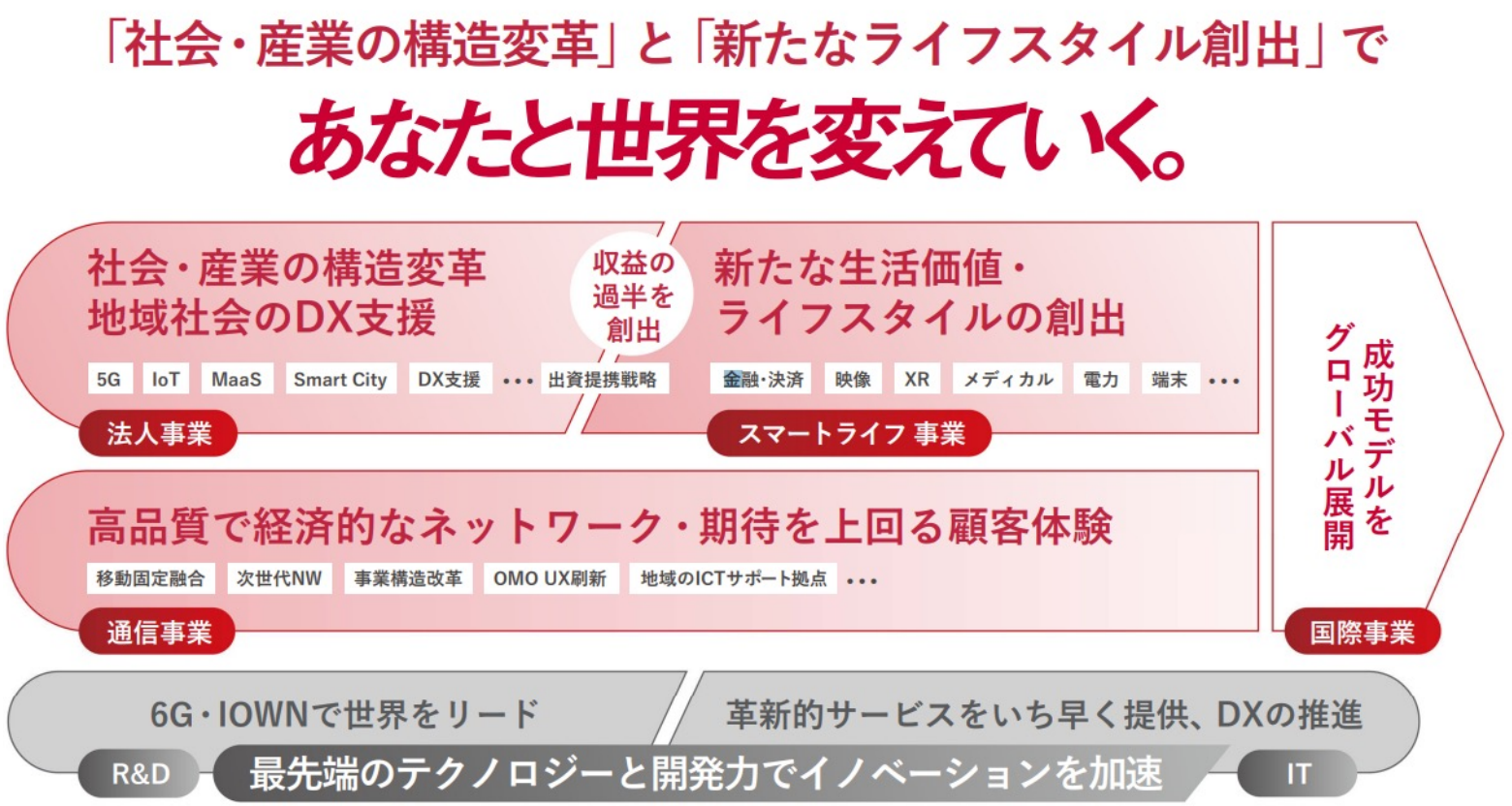
社名：　株式会社 NTT ドコモ
NTT DOCOMO, INC.

営業収益：6兆590億円 （2022年度）
営業利益：1兆939億円 （2022年度）
従業員数：47,151名（2023年3月31日現在）
※グループ会社含む

グループ会社
NTT コミュニケーションズ、NTT コムウェア、
ドコモ CS、ドコモ・サポート　等

通信事業	携帯電話サービス（5G サービス、LTE（Xi）サービス、FOMA サービス）、光ブロードバンドサービス、衛星電話サービス、国際サービス、各サービスの端末機器販売など
スマートライフ事業	動画配信・音楽配信・電子書籍サービス等のd マーケット を通じたサービス、金融・決済サービス、ショッピングサービス、 生活関連サービスなど
その他の事業	補償サービス、法人 IoT、システム開発・販売・保守受託 など

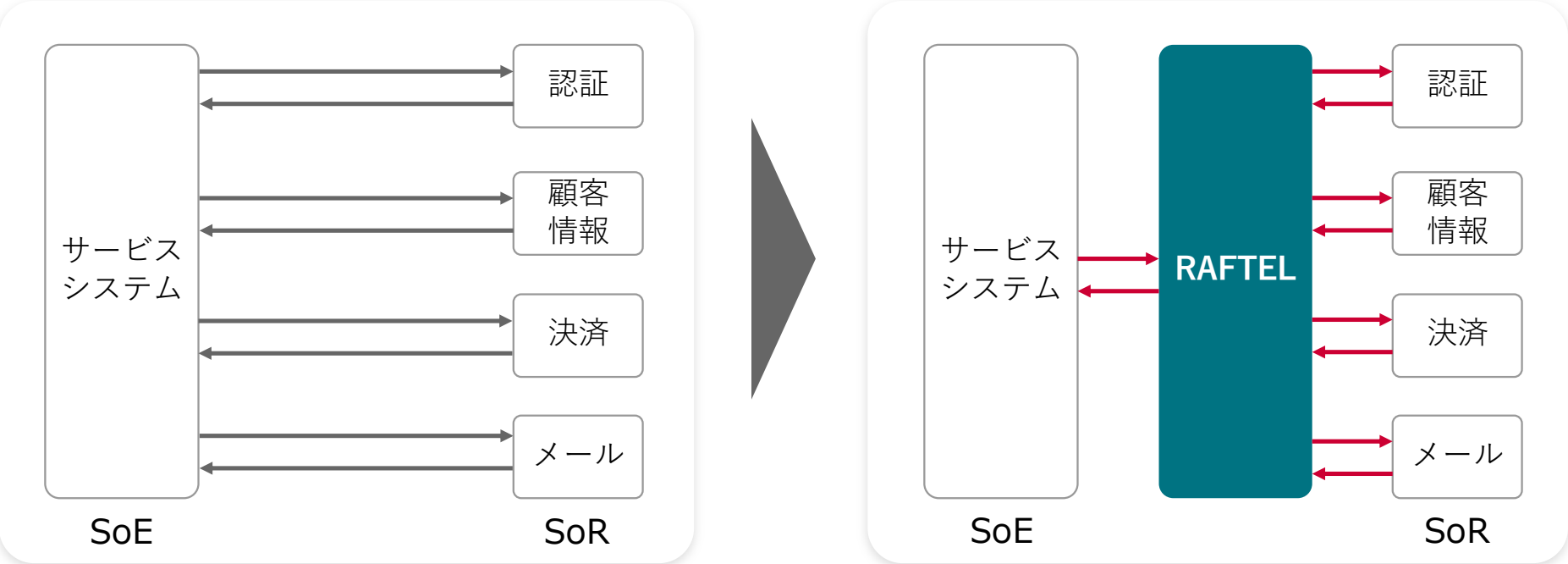
NTT ドコモ中期戦略



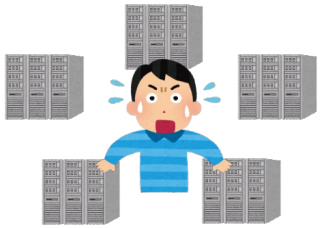
新ドコモグループ 2025 年度収益の過半をスマートライフ事業と法人事業で創出

RAFTEL とは

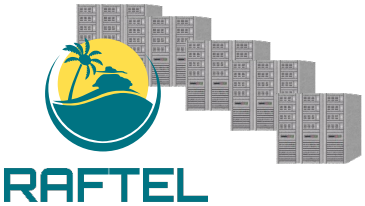
✓ SoR と SoE の間に API 基盤を作ることにより、分散して存在する社内のコア・アセットを活用しやすくすることで、主にスマートライフ領域のサービス開発を効率化



RAFTEL とは



社内には認証、決済などの様々な基幹システムが存在
コア・アセットとして活用できるが、
システムが分散し API 仕様も異なる



RAFTEL により分散した API を1つのシステムに統合、
API の仕様も統一
社内の開発者たちはアセットに容易にアクセス可能

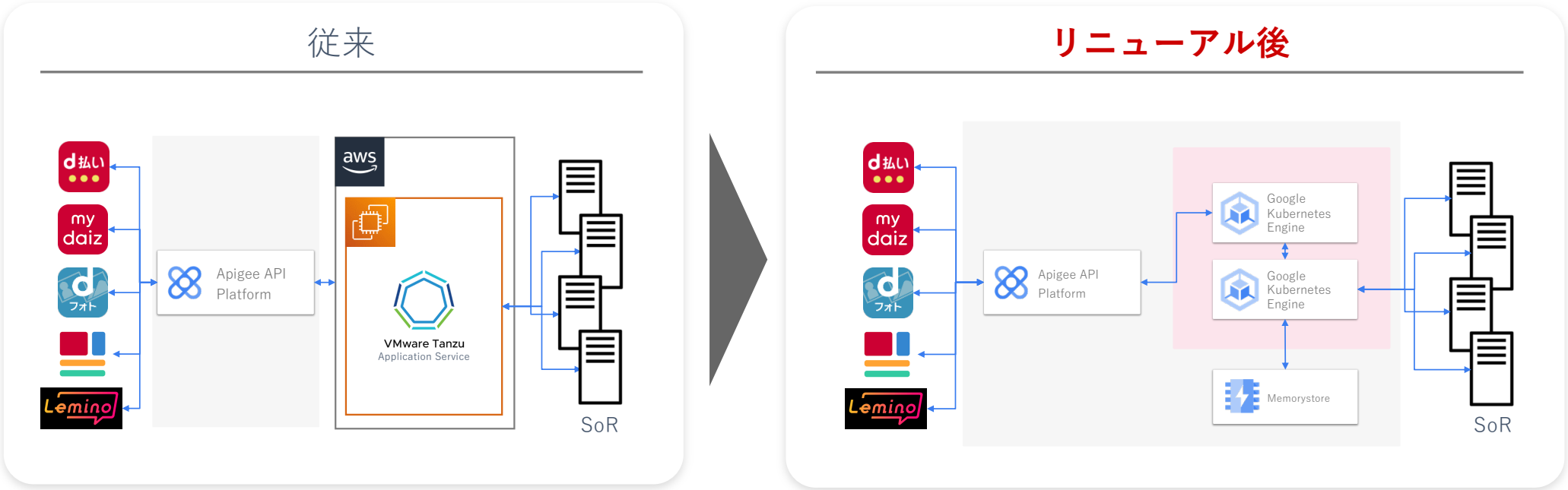
システムの統一、API 仕様の統一だけではなく、
開発者が共通的に実装する複数 API のマッシュアップを
用意し、より開発しやすい環境を提供

RAFTEL の稼働状況

60 以上	約 8000万	60 以上	約 5.3億
利用サービス/システム数	利用サービスの合計ユーザー数	提供 API 数	1 日当たりの処理件数

RAFTEL のリニューアル

- ✓ 2019 年に作ったシステムを見直し GKE 上で再構築
- ✓ 2024 年 3 月までに全ての API を移行予定



リニューアル経緯

✓ RAFTELの役割変化

- 利用するサービス/システムが増加、接続先となる基盤システムも増え、トラフィックも扱う情報の種類も大幅に増加
- サーバだけでなく Web ・スマートフォンアプリからも直接アクセスを受ける API を提供しトラフィックが激増
- サービス側で大規模なイベントを実施するとトラフィックが爆増

→ **もっと柔軟にキャパシティコントロールをしたい**

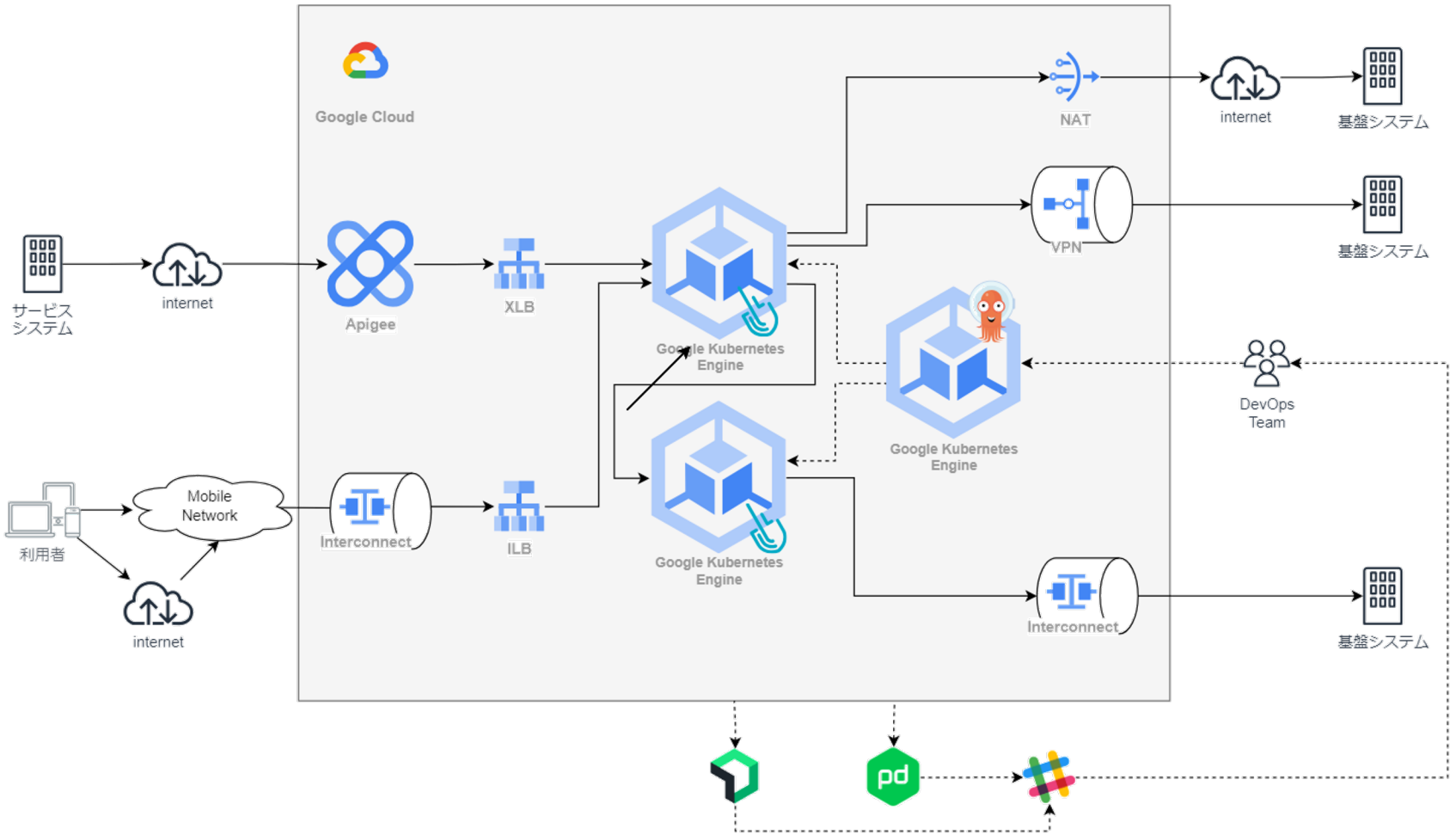
✓ 本当は RAFTEL のインフラ基盤を使った社内のサービス開発プラットフォームを作りたかったが、色々な事情でうまくいかず、現状のシステムにおけるスケールメリットを得られなかった

GKE Standard を採用した理由

- ✓ RAFTEL 利用者の接続先変更を避けるべく Apigee は継続利用を前提として、パブリッククラウドを1つにまとめたい　→ Google Cloud
- ✓ コンテナ運用の標準を K8s にしたい、学習コストは必要だが K8s を理解し構築・運用できるようにしたい　→ GKE

- ✓ 大量のトラフィックを捌く場合、オートスケールの速さを求める場合は GKE Standard が推奨されている
- ✓ Pod 再起動によるトラフィックへの影響発生をコントロールするため、ノードプールのアップグレードや増設/減設は自分たちの望むタイミングで実施したい

システム構成



GKE 苦労話

✓ 構築、試験、移行など様々な問題が出た

→ Google PSO の支援を受けて乗り切る

✓ ノードプールのアップグレードは Blue/Green 方式を採用したが、アップグレード時にタイムアウトエラーやレスポンス遅延などが発生

- Graceful Shutdown を実装
- Apigee、Cloud Load Balancing、コンテナのKeep Alive タイムアウト値を調整
- アプリケーションの実装を見直し
- アプリケーションのシャットダウン → Pod 削除の順序性を守れないケースが稀にありエラー発生は完全解決には至っていない （ Cillium の仕様によるもの、今からは Dataplane V2 の使用をやめられないので今後の改善に期待）

リニューアルにおける方針

✓ RAFTELチーム目線
- コンテナオーケストレーションのデファクトである K8s を採用するのは当然の流れ
- これまではドコモ社内の情報セキュリティルールを順守したサービスサーバだけが RAFTEL の接続先だったが、アプリや Web からのアクセスを受けることになり、より高いレベルでのセキュリティ対策が必要

✓ 自組織の状況
- 2025年の崖に向け SoR 領域における技術的負債の解消に着手
- 巨大なオンプレシステムを分解しクラウドシフト（実施中）、または今後コンテナへのリファクタリングを実施予定

リニューアルにおける方針

✓ これまでの RAFTEL チームの成果
- マイクロサービス、CI/CD、O11y、インシデント対応自動化などを積極的に採用するだけでなく、チーム文化の醸成を含め DevOps を高いレベルで実現
- DevOps 実現の方法、ノウハウなどを積極的に展開

自組織内で将来的に大規模なコンテナ運用が行われる可能性を見据え、
先行して RAFTEL で K8s 環境における DevSecOps を確立し
ノウハウを蓄積、展開することを目指す

RAFTEL

DevSecOpsの実現で課題となる
- K8s運用効率化
- セキュリティ対策

→ sysdig

Sysdigの採用により解決を図る

Sysdig を採用した理由

✓ K8s運用効率化とセキュリティ対策を同時に実現

- Sysdig の UI は K8s を意識した構成になっており、メトリクス、ログ、イベントといった K8s 上の挙動も容易に確認可能

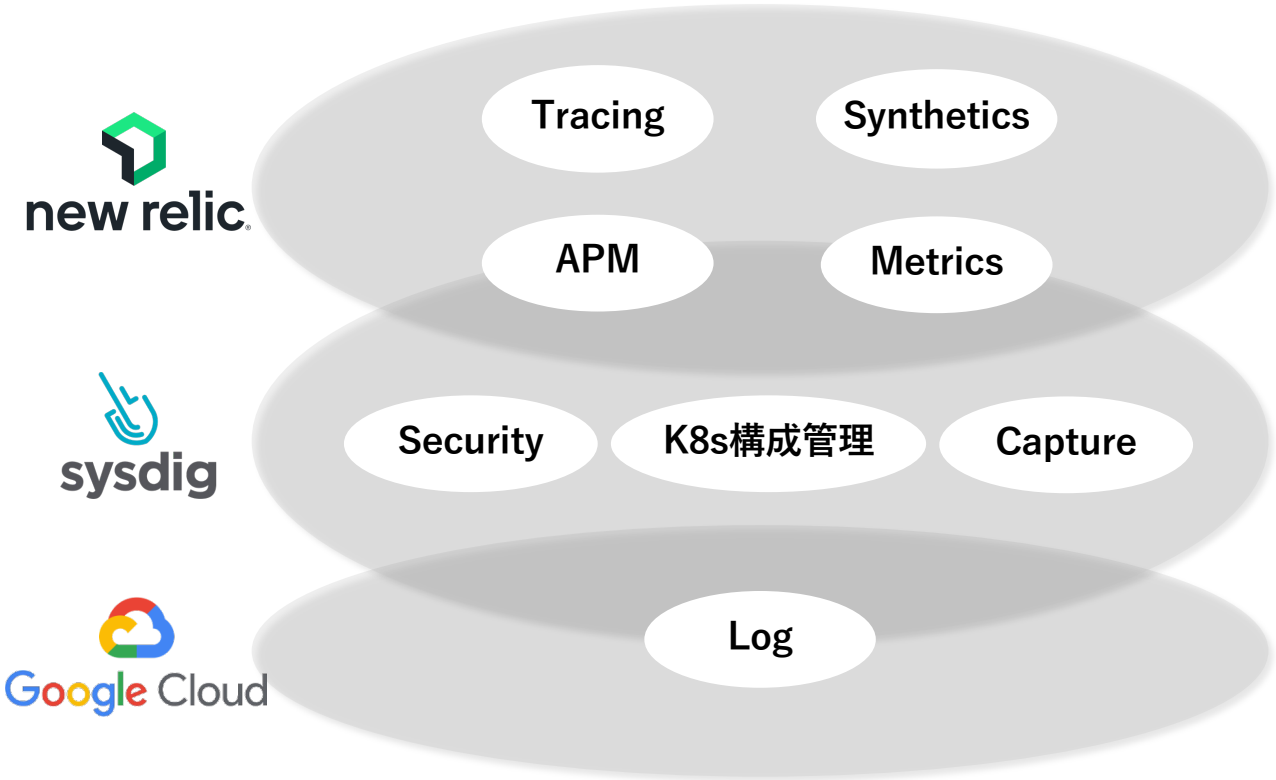
- GKE のワーカーノード上に配置する Sysdig Agent がシステムコールを hook することにより、ノード上の挙動を把握するアーキテクチャを採用していることが最大のメリット

  K8s 上の障害やセキュリティインシデントの調査時はシステムコールのキャプチャによるコンテナ内部やコンテナ間通信の可視化が可能であり、 管理・運用の難易度を下げられる

  Pod に脆弱性が含まれていても実際に使用しているライブラリなのかを確認することで対応すべきライブラリの数を減らすことができ、運用コスト削減が期待できる

  ファイル操作、通信内容、起動プロセスなどが記録でき、不正な動作（=振る舞い検知）を通知・抑止可能

監視・運用機能のオーバービュー

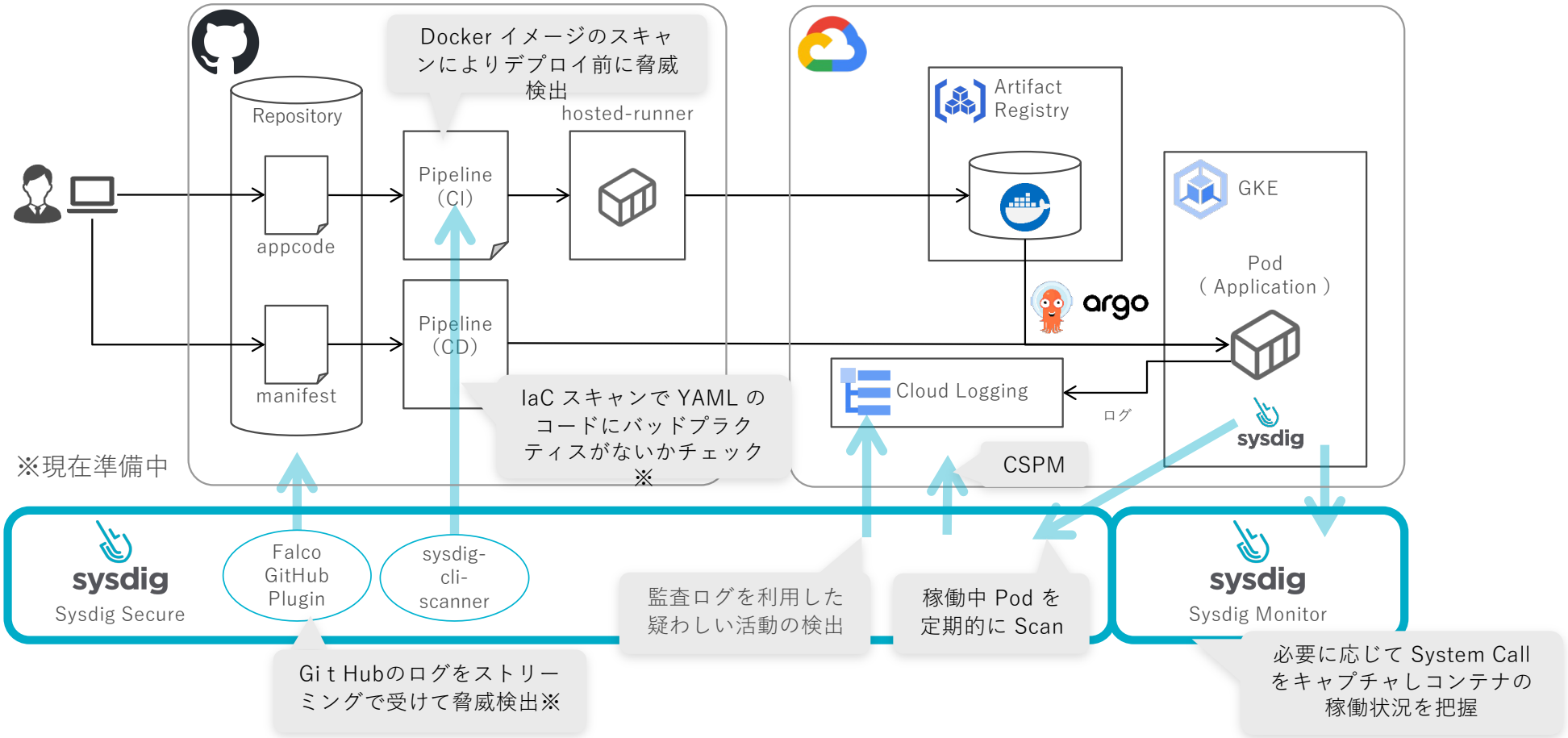


- メトリクス、APM、Synthetics で異常を検知した際に運用者へアラート発報
- 調査でトレーシングを参照

- セキュリティインシデントを検知した際に運用者へアラート発報
- 調査において K8s 構成情報、メトリクス、APM 情報を参照し、必要に応じてキャプチャを取得して詳細分析

- ログに出力された異常メッセージを検知した際に運用者へアラート発報

Sysdig 活用方法



今後の取り組み

- ✓ Sysdig を使い始めたばかりで、DevSecOps をベースとした働き方や文化の醸成はこれから
- ✓ RAFTEL はモバイルネットワーク機能の API 化や社外向け API の提供を検討中
- ✓ 先行して利用されている事例などを参考にしながら Sysdig を活用し、RAFTELの進化と運用効率化・セキュリティ対策の高度化を両立していきたい