『マギアレコード 魔法少女まどか☆マギカ外伝』を支えるインフラ

株式会社f4samurai

森田 桂介

2018/06/01

会社概要

- 社名
  株式会社f4samurai
- 事業内容
  モバイルオンラインゲームの企画開発および運営
  ソーシャルアプリ・WEBサービスの企画開発
- 設立
  2010年1月8日
- 従業員数
  120人

サービス中のタイトル

2013.3.8

BORDER BREAK Mobile
疾風のガンフロント

2013.12.17

アンジュ・ヴィエルジュ
ガールズバトル

2015.4.22

Hortensia SAGA
蒼の騎士団

2017.8.22

マギアレコード

ワンダーグラビティ ～ピノと重力使い～

マギアレコード 魔法少女まどか☆マギカ外伝

クライアント

- 更新性を重視し、ハイブリッド構成
- WebはSPAとして実装

WebViewとCocos2d-x

Cocos2d-x

マギレコ以前のインフラ環境

- マギレコ以前のタイトルではオンプレミスのインフラを利用
- マギレコが弊社初のAWS案件

オンプレミスでの問題点

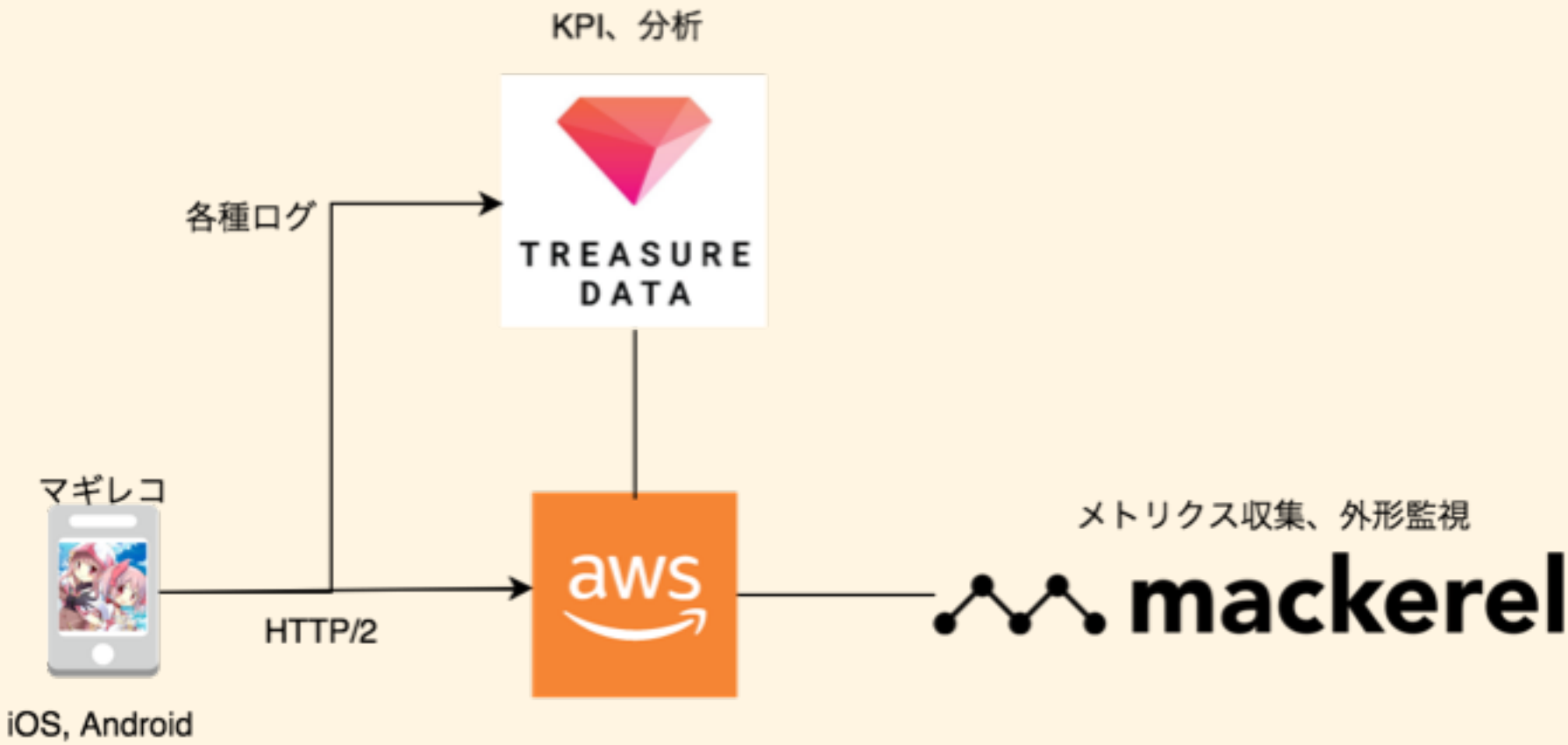
- クラウドと比べ柔軟性に乏しい
- ピーク時を想定した構成が必要
- 実際のピークはリリースしてみないと分からない

AWS導入の決め手

- Auroraの存在が一番の理由
- 高いパフォーマンス
- 調達の容易さ
- スケールアウト、イン、アップ、ダウンの容易さ

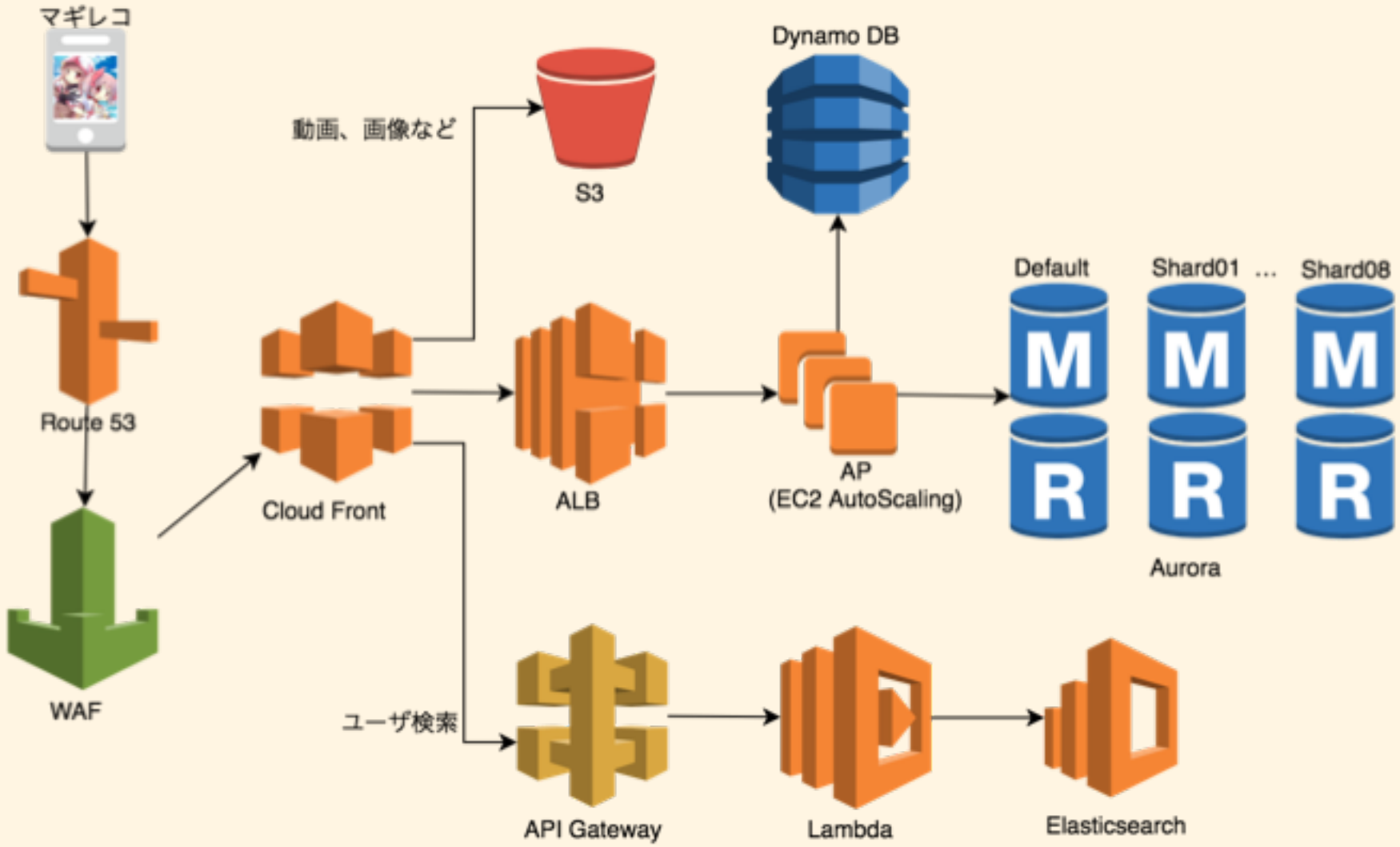
リリース時はかなり大きめのキャパシティを確保していたが、
徐々に適正なところまで調整

全体の構成概要


KPI、分析
TREASURE DATA
各種ログ
マギレコ
HTTP/2
aws
メトリクス収集、外形監視
mackerel
iOS, Android

構成詳細

AP


マギレコ
Route 53
WAF
Cloud Front
動画、画像など
S3
ALB
Dynamo DB
AP
(EC2 AutoScaling)
Default
Shard01
...
Shard08
M
R
Aurora
ユーザ検索
API Gateway
Lambda
Elasticsearch

CloudFront

- すべてのリクエストをCloudFrontで受けている
  - APIのリクエストをALBで受けていないのはパフォーマンスのため
  - 配信にエッジロケーションが使用され、地理的に近いところから配信されるこでレイテンシが改善される
  - レスポンスの圧縮をよしなにやってくれる

DynamoDB

- 書込みの多いデータを保存(アイテムの取得履歴など)

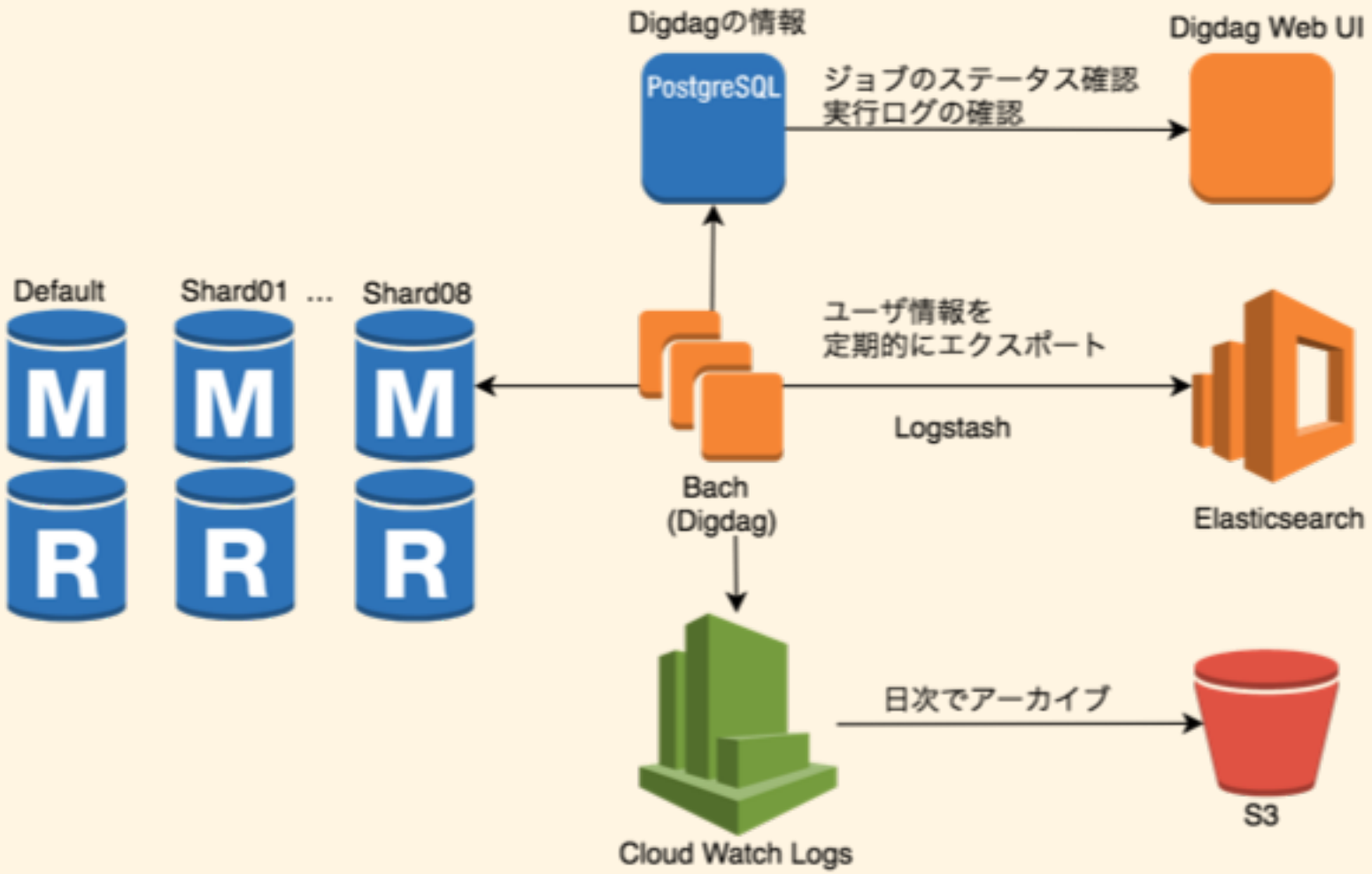
Aurora

- ユーザIDなどを基に8つにシャーディング
- Defaultにマスタなどシャーディングしないデータを保存
- 読み取りはリードレプリカを参照
- シャーディングにより100,000TPSを達成

Elasticsearch

- ユーザ検索、サポート選択に利用
- 要件としてユーザ名での曖昧検索があったため
- Auroraで曖昧検索をするわけにはいかない

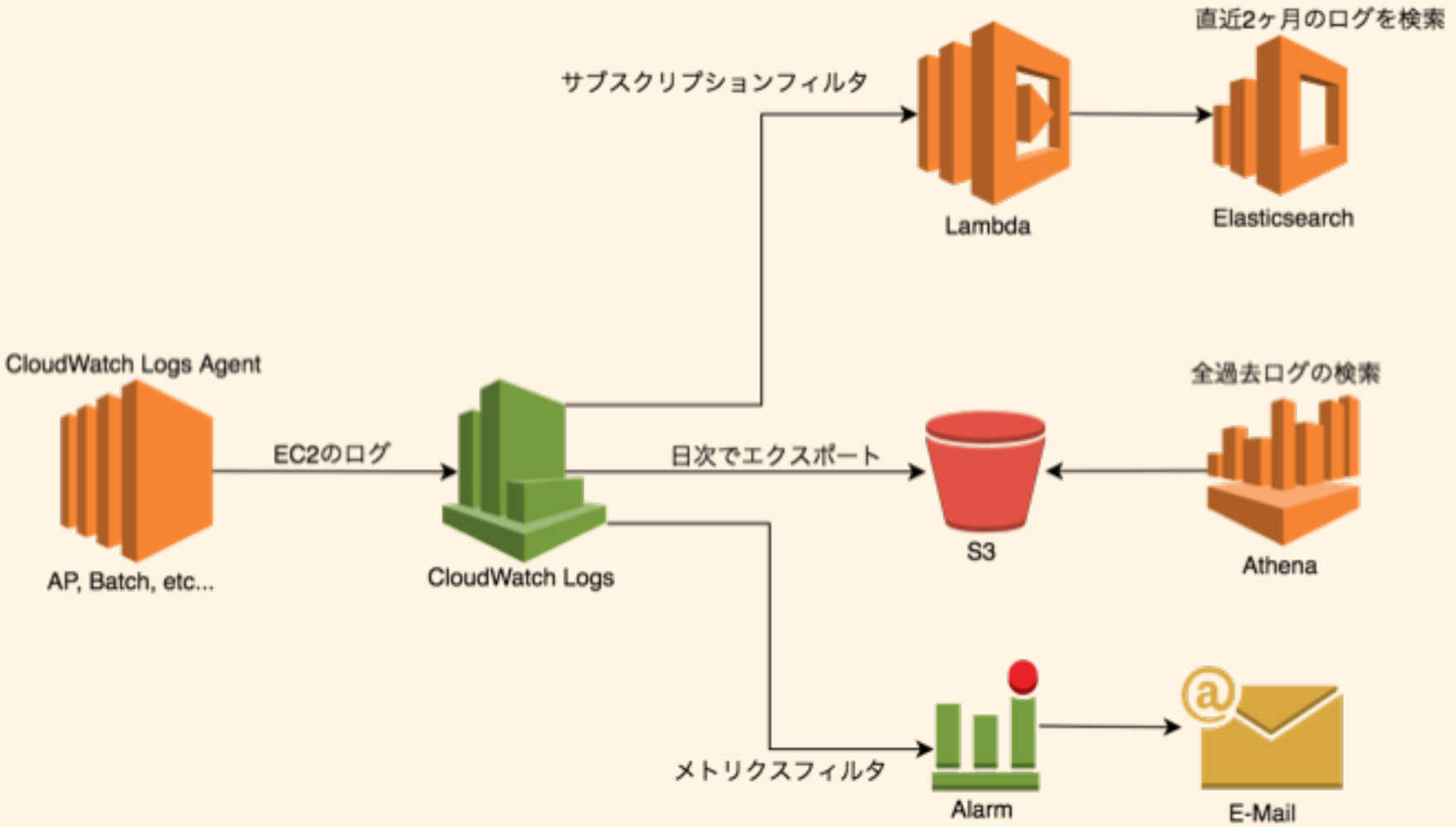
Batch


Digdagの情報
PostgreSQL
ジョブのステータス確認
実行ログの確認
Digdag Web UI
Default
Shard01
...
Shard08
M
M
M
R
R
R
ユーザ情報を
定期的にエクスポート
Logstash
Bach
(Digdag)
Elasticsearch
日次でアーカイブ
S3
Cloud Watch Logs

Digdag

- バッチもスケールできるように分散実行環境を構築
- バッチのステータス確認、実行をGUIから操作
- PostgreSQLにジョブのスケジュールやステータスを格納

Log


直近2ヶ月のログを検索
サブスクリプションフィルタ
Lambda
Elasticsearch
CloudWatch Logs Agent
EC2のログ
AP, Batch, etc...
CloudWatch Logs
日次でエクスポート
S3
全過去ログの検索
Athena
メトリクスフィルタ
Alarm
E-Mail

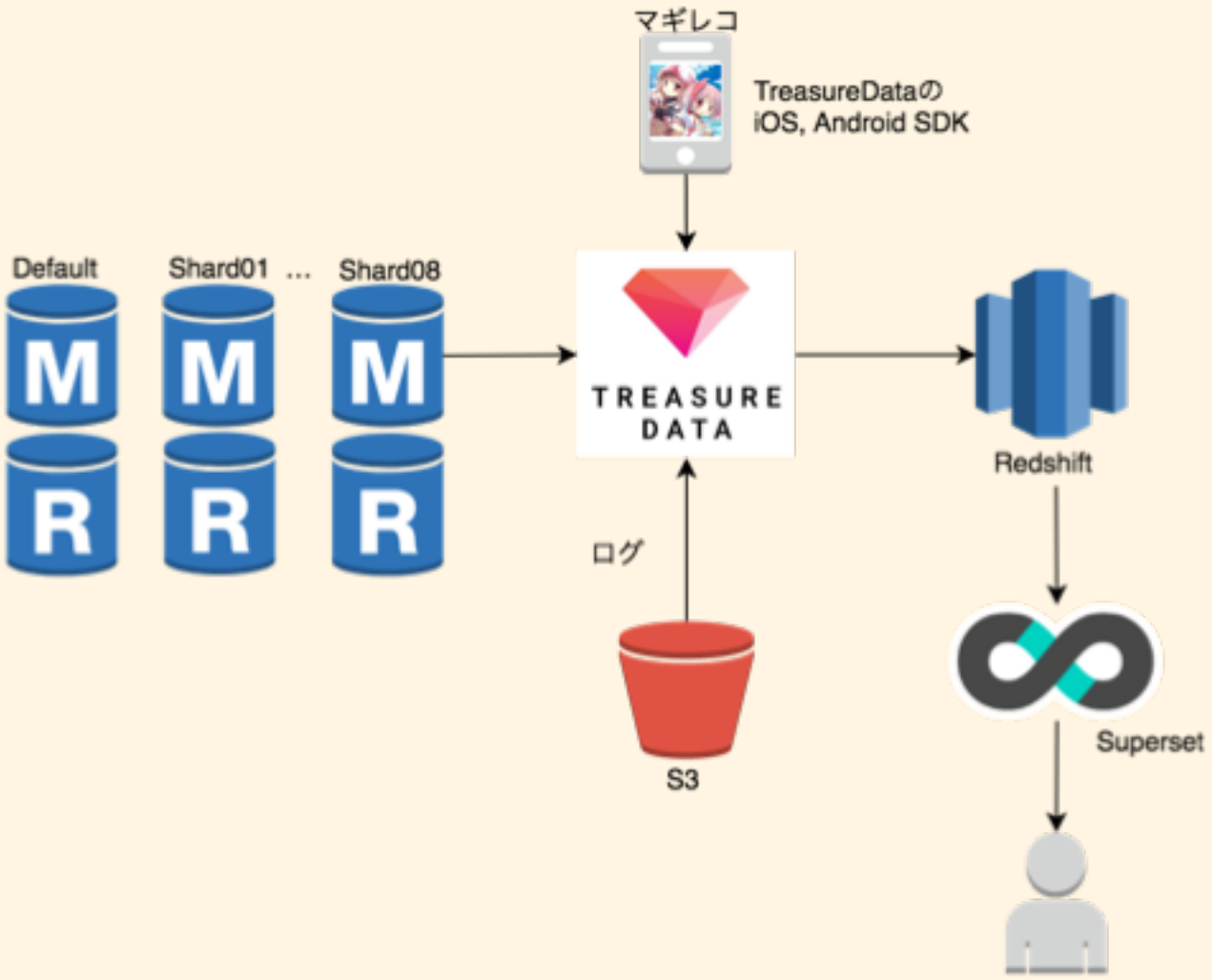
CloudWatch Logs

- EC2から発生するログを収集
  - アクセスログ、エラーログなど
- サブスクリプションフィルタでElasticsearchへストリーミング
- Elasticsearchにてアクセス解析、エラーログの調査

ログのアーカイブ

- 日次でCloudWatch Logsのログを S3へエクスポート
- S3の全過去ログをAthena、 Redshiftから検索

KPI


マギレコ
TreasureDataの
iOS, Android SDK
Default
Shard01
...
Shard08
M
M
M
R
R
R
TREASURE
DATA
Redshift
ログ
S3
Superset

KPI

- Treasure Dataへ各種データを集め、Redshiftへエクスポート
- RedshiftにてKPI、分析のクエリを実行
- Supersetにて可視化

インフラの管理

Terraform

- AWSのほぼすべてをTerraformで管理
- 複数環境を同じコードで管理
  - テスト、調整、ステージング、本番環境を管理
  - 環境の差異は変数ファイルに定義

Ansible

- 設定管理に使用
  - OSの設定
  - ミドルウェアのインストール、設定

Ansibleの適用方法

- JenkinsでAnsibleのソースコードをS3へアップロード
- EC2はイミュータブルに管理しているので、再作成
- EC2起動時に cloud-init にて初期化処理を実行
  - Ansibleのインストール
  - S3からソースコードをダウンロード
  - Ansibleの実行

AWSを使ってみて

開発体制 (40名)

- プランナー　８名
- サーバエンジニア　３名
- クライアントエンジニア　４名
- フロントエンジニア　２名
- デザイナー　１８名
- シナリオライター　５名
- その他 協力会社様

サーバエンジニア３名

- サーバサイドエンジニア
  １.５名
- インフラエンジニア
  １.５名

メイン担当者が１名ずつ

インフラエンジニアとして私

- 今まではサーバサイドエンジニア、今は兼任
- インフラ経験無し
- AWS経験そんなに無し

開発期間

- 2016/08　プロジェクトジョイン
- 2016/10　AWSの採用決定
- 2017/05　リリース延期
- 2017/08　リリース

なぜできたか

- 充実したドキュメント
- ベストプラクティス、リファレンスアーキテクチャの活用
- マネージドサービスの活用
- ソリューションアーキテクトのサポート活用
- AWS Black Belt Online Seminarの活用

全職種で採用募集中！

- サーバサイドエンジニア (Java)
- Unityエンジニア/cocos2d-xエンジニア
- フロントエンドエンジニア (JavaScript, CSS)
- ゲームプランナー
- 3Dデザイナー/2Dデザイナー
- イラストレーター
- シナリオライター/スクリプター

ご清聴ありがとうございました。