Nintendo Switch Online を支える サーバーシステム開発

任天堂　ネットワークシステム部
田中 克磨
吉野 真弘

自己紹介

田中 克磨（4年目）

- 担当：アプリ／インフラ
- これまで：フレンドおすすめ機能 など

吉野 真弘（5年目）

- 担当：アプリ／インフラ／フロントエンド
- これまで：ゲームニュース配信システム など

発表の流れ

1. サービス紹介
2. 権利管理サーバーの開発
3. セーブデータお預かりサービスの開発
4. まとめ

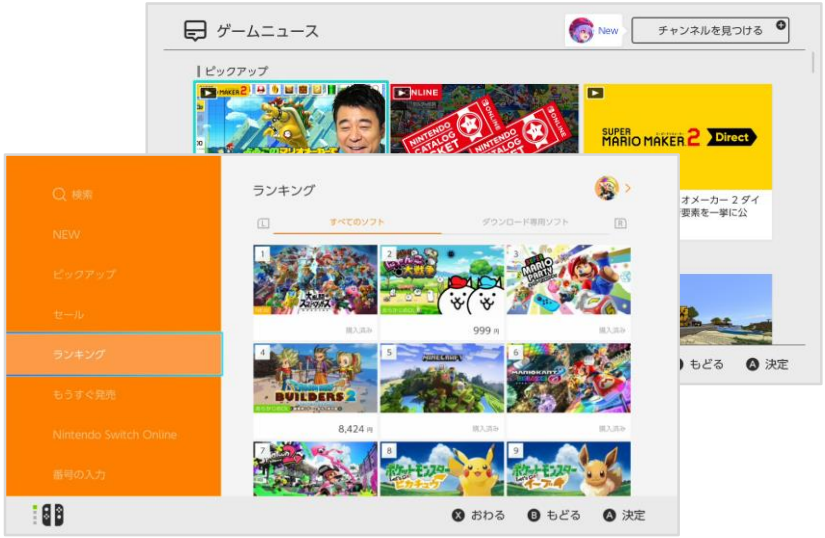
サービス紹介

Nintendo Switch

累計販売数 3,474万台（2019年3月末時点）

Nintendo Switch のネットワーク機能

- アカウント／フレンド
- ニンテンドーeショップ
- ゲームニュース
- プッシュ通知
- など…



※ AWS Summit Tokyo 2018 - Nintendo Switch 向けプッシュ通知システム「NPNS」
https://aws.amazon.com/jp/summit2018-report/

Nintendo Switch Online

ここからの内容

Nintendo Switch Online サーバーシステムの開発事例

- 権利管理サーバー
- セーブデータお預かりサービス

権利管理サーバーの開発

権利管理サーバーとは？

- Nintendo Switch Online 加入者の権利情報を管理
  - アカウント情報
  - プラン（個人／ファミリー）
  - 加入期間
  - 自動更新の有無など
- 将来的な拡張性を考慮し、独立したシステムとして開発
  - アカウントやニンテンドーeショップとは Web API で連携

権利の書き込み

- ユーザーが加入すると、プランや期間などのデータが権利管理サーバーに登録される

権利の参照（1/2）

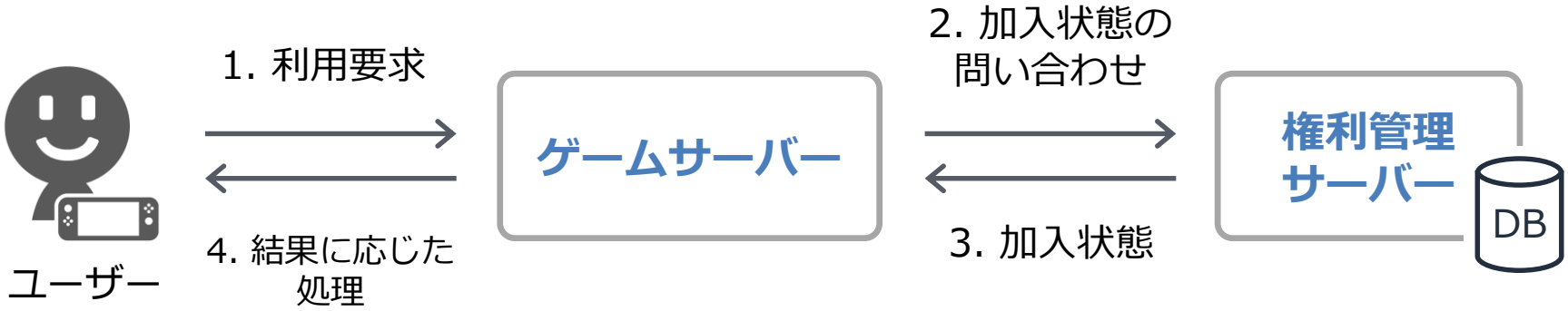
オンラインプレイ利用

加入している場合

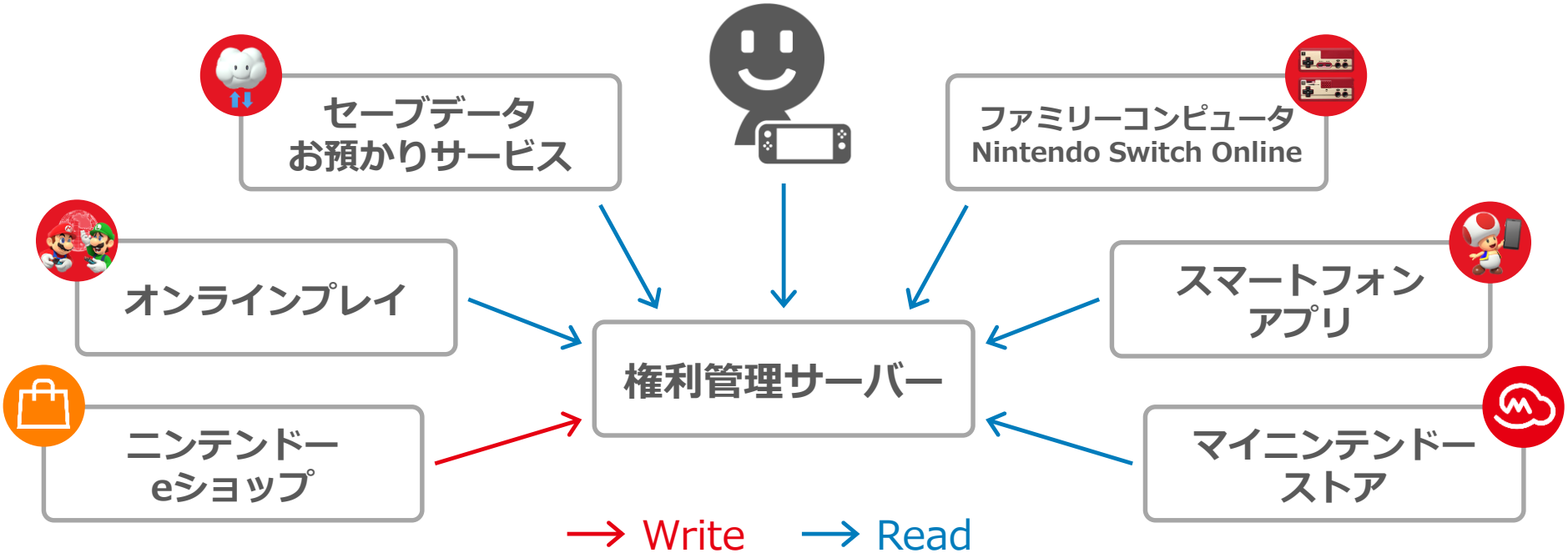
加入していない場合

権利の参照 (2/2)

- サービス利用時にユーザーの加入状態を問い合わせ、その結果に応じて処理を切り替える



権利管理サーバーのクライアント



権利管理サーバーの要件 (抜粋)

可用性

- AWS リージョン規模の障害未満は稼働

スケーラビリティ

- サービス規模の拡大にあわせてスケール
- ゲーム特有のアクセスパターンに対応できる
  - ゲームの発売、イベント、更新データの配信、クリスマスなど

課題：データベース

- 大量の参照アクセスによる負荷への対策
- ゲーム特有のアクセスに応じて柔軟にスケール
- レプリケーション遅延対策
  - 権利の即時反映は必須
- 運用コスト

Amazon Aurora の採用

要件に対して Aurora の特徴がよくマッチしていた

- リードレプリカの追加やサイズ変更が簡単
- 読み取りエンドポイントによる負荷分散
- 低遅延でのレプリケーション
- ストレージの自動拡張
- など…

その他の採用技術

アプリ

- Ruby on Rails

インフラ


ALB


Lambda


Athena


Route 53

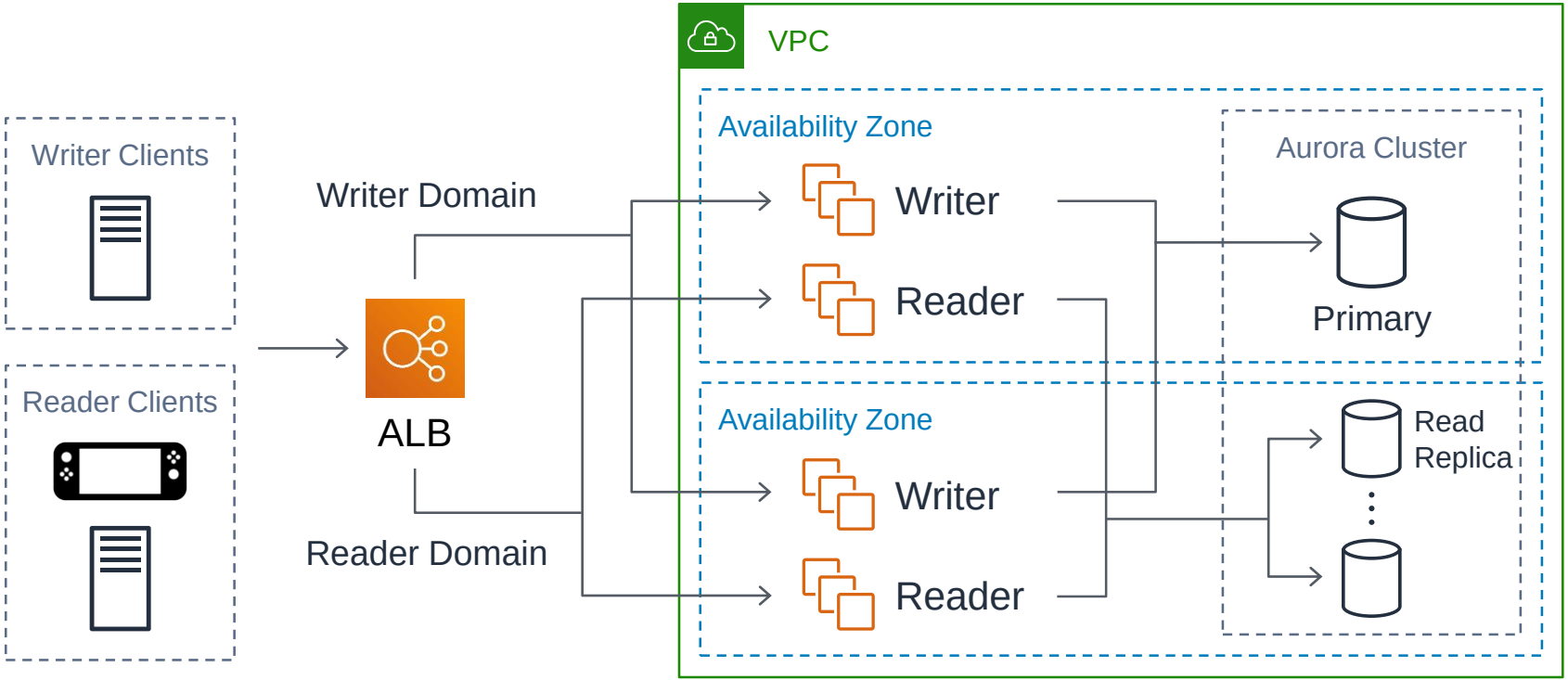

CloudFormation


CloudWatch

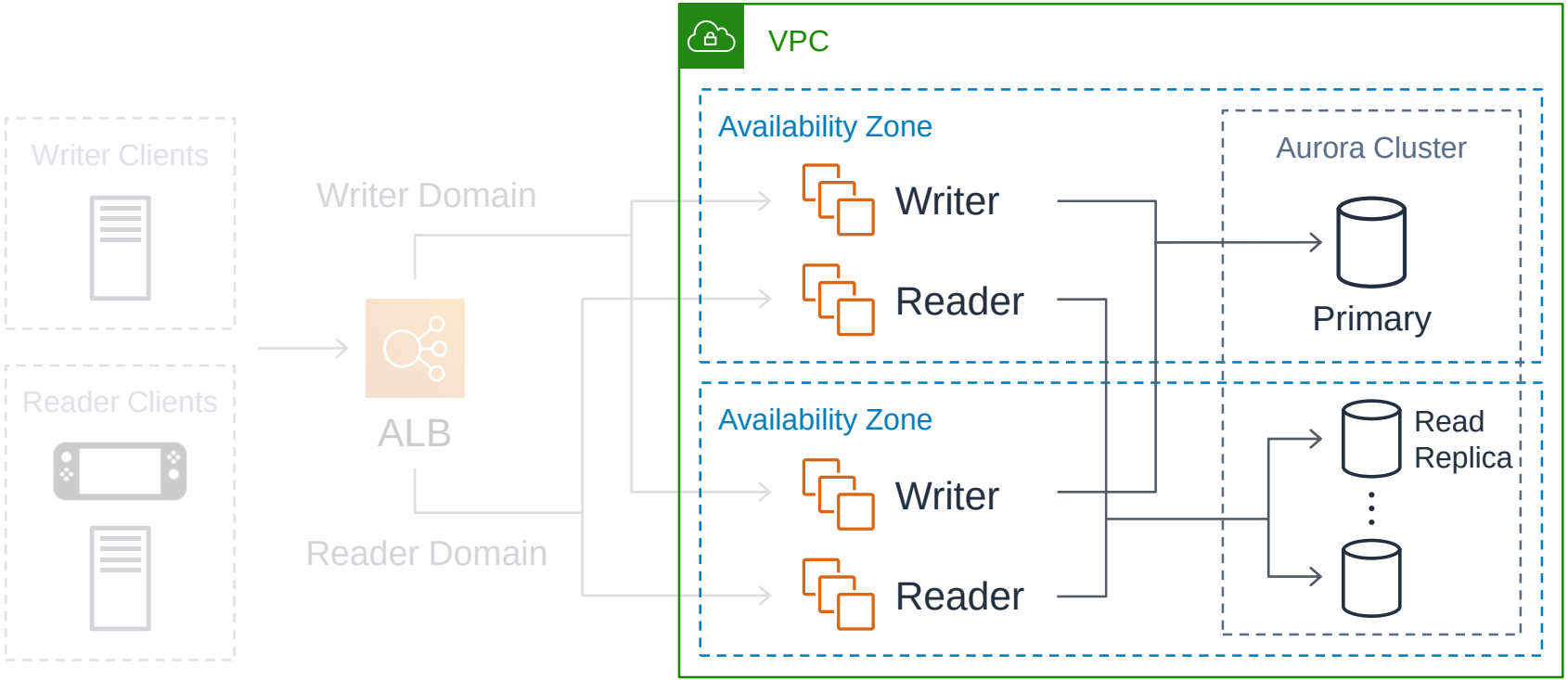
権利管理サーバーの開発

インフラ設計

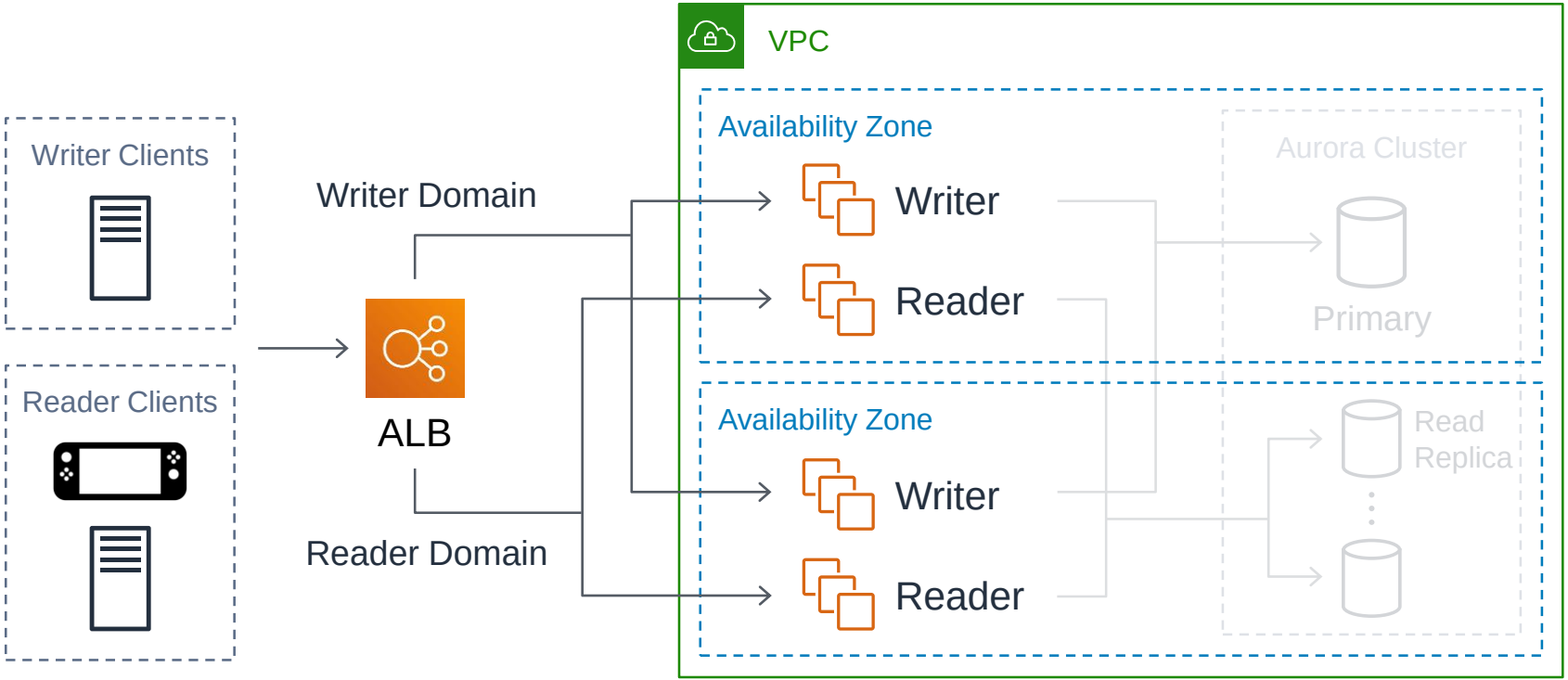
全体構成


VPC
Writer Clients
Reader Clients
Writer Domain
ALB
Reader Domain
Availability Zone
Writer
Reader
Availability Zone
Writer
Reader
Aurora Cluster
Primary
Read
Replica

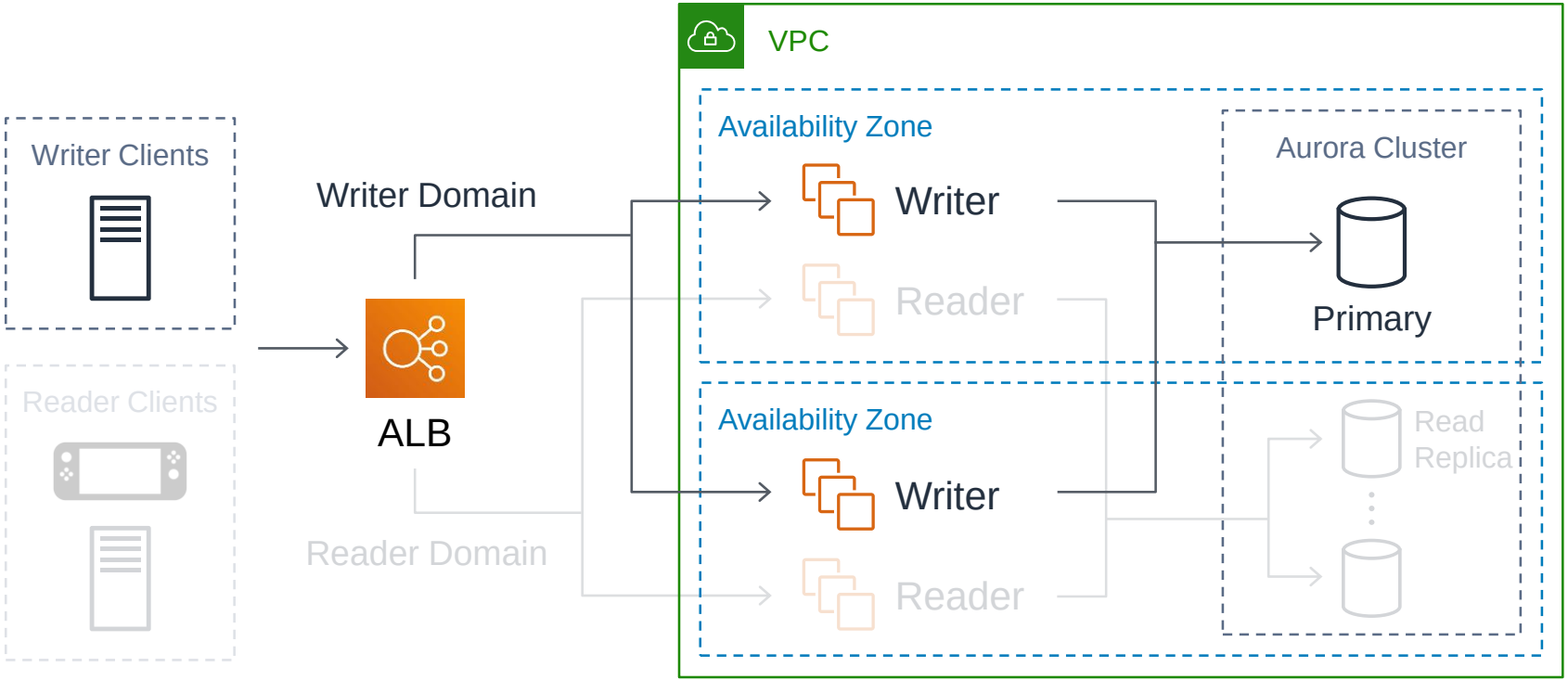
Reader/Writer の分離



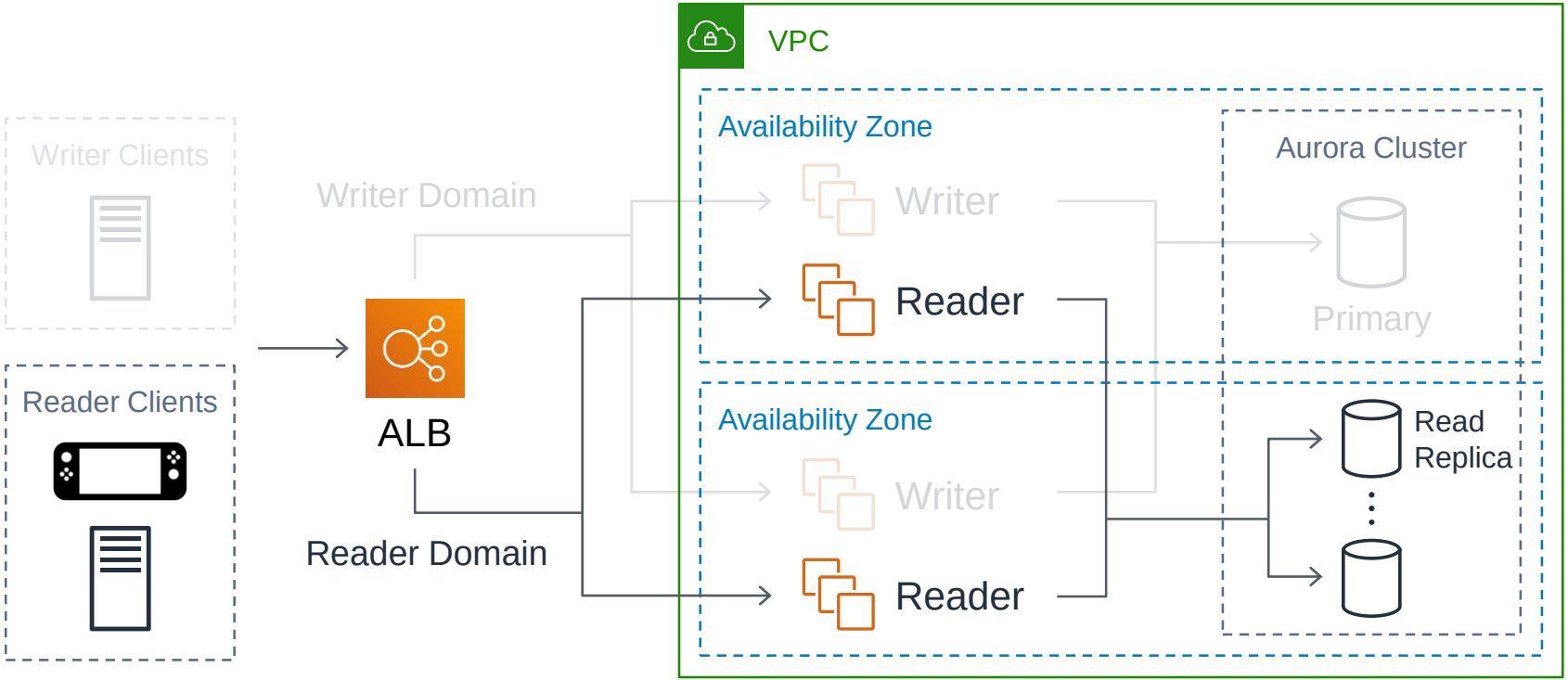
Reader/Writer の分離



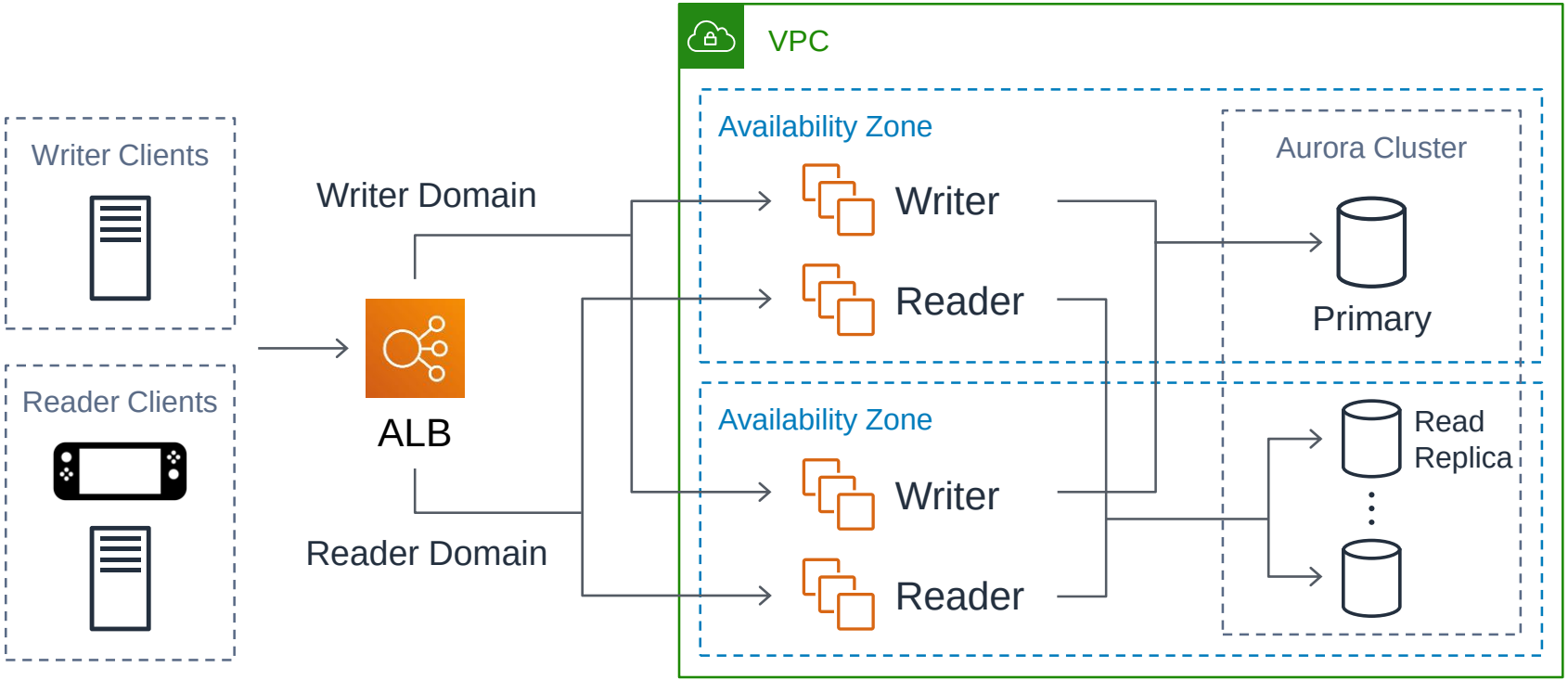
Reader/Writer の分離



Reader/Writer の分離



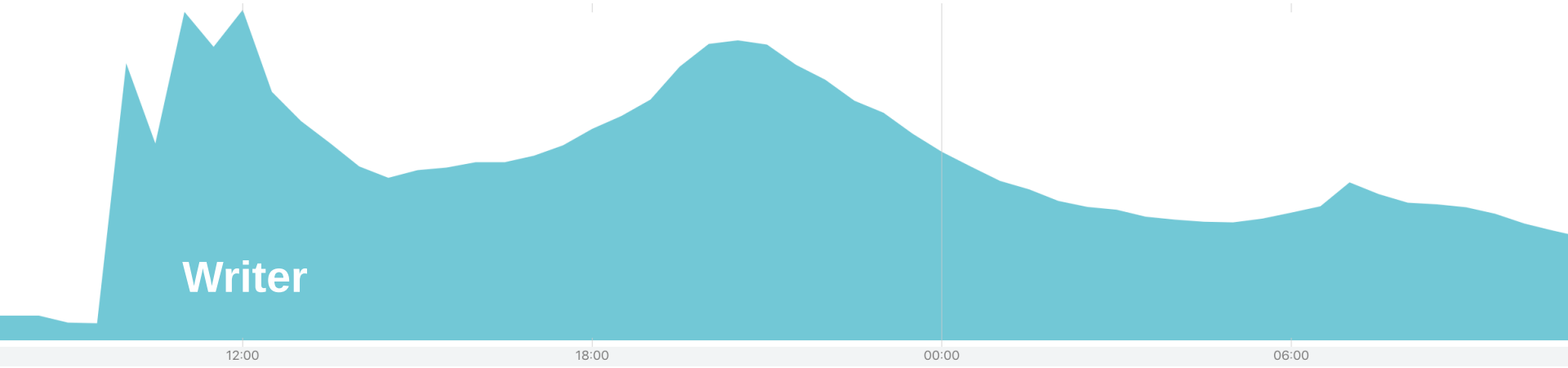
Reader/Writer の分離

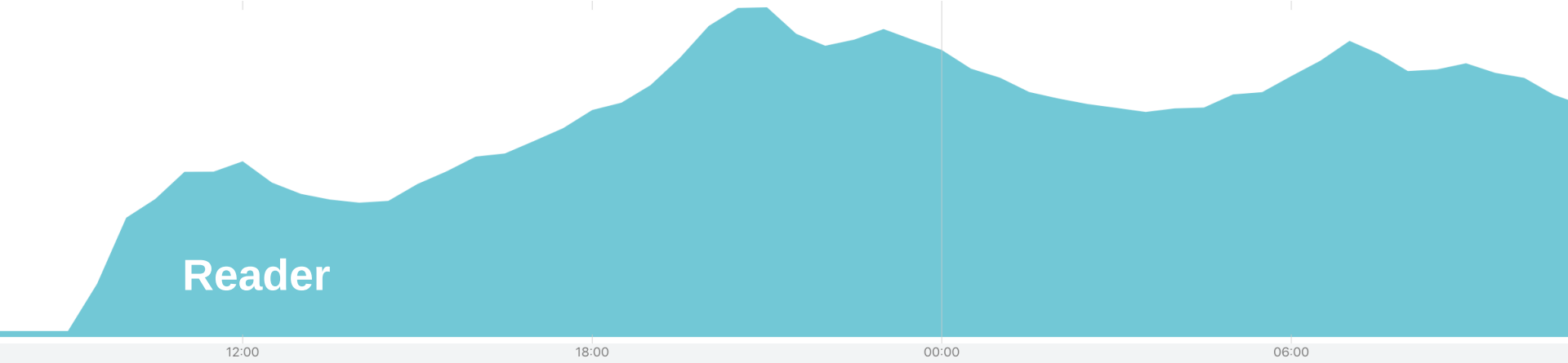


権利管理サーバーの開発

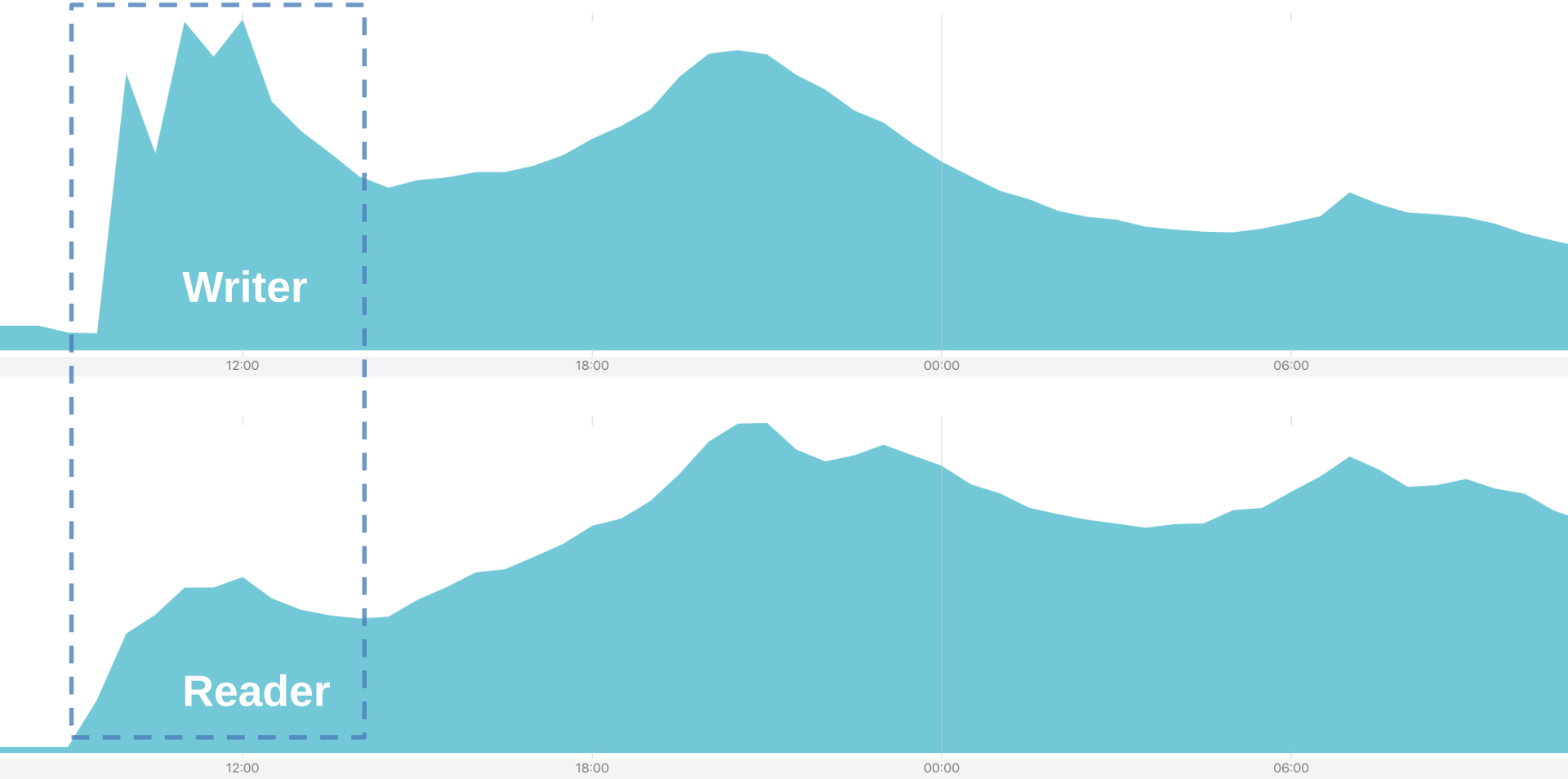
振り返り

2018年9月19日　Nintendo Switch Online 開始

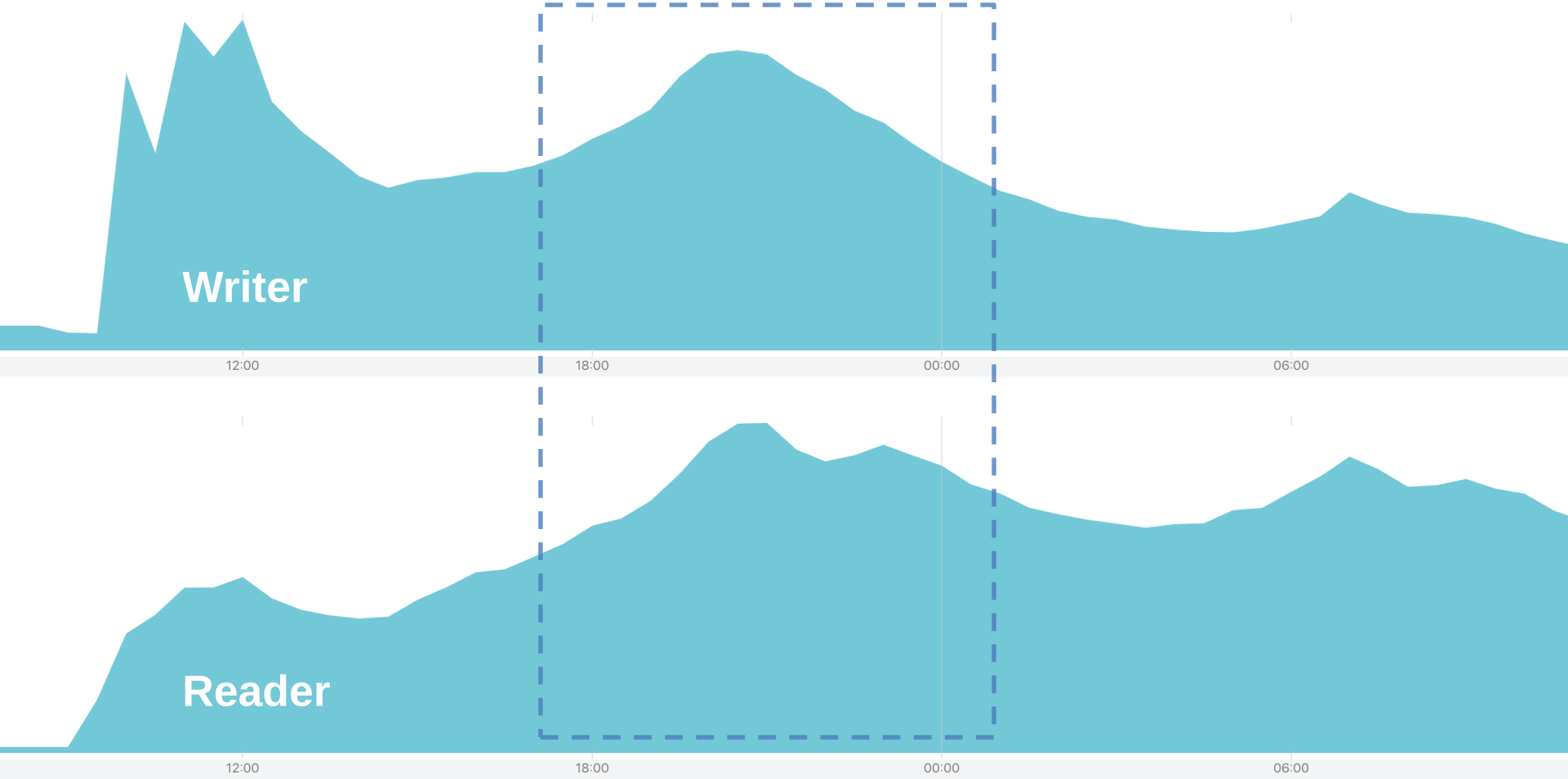




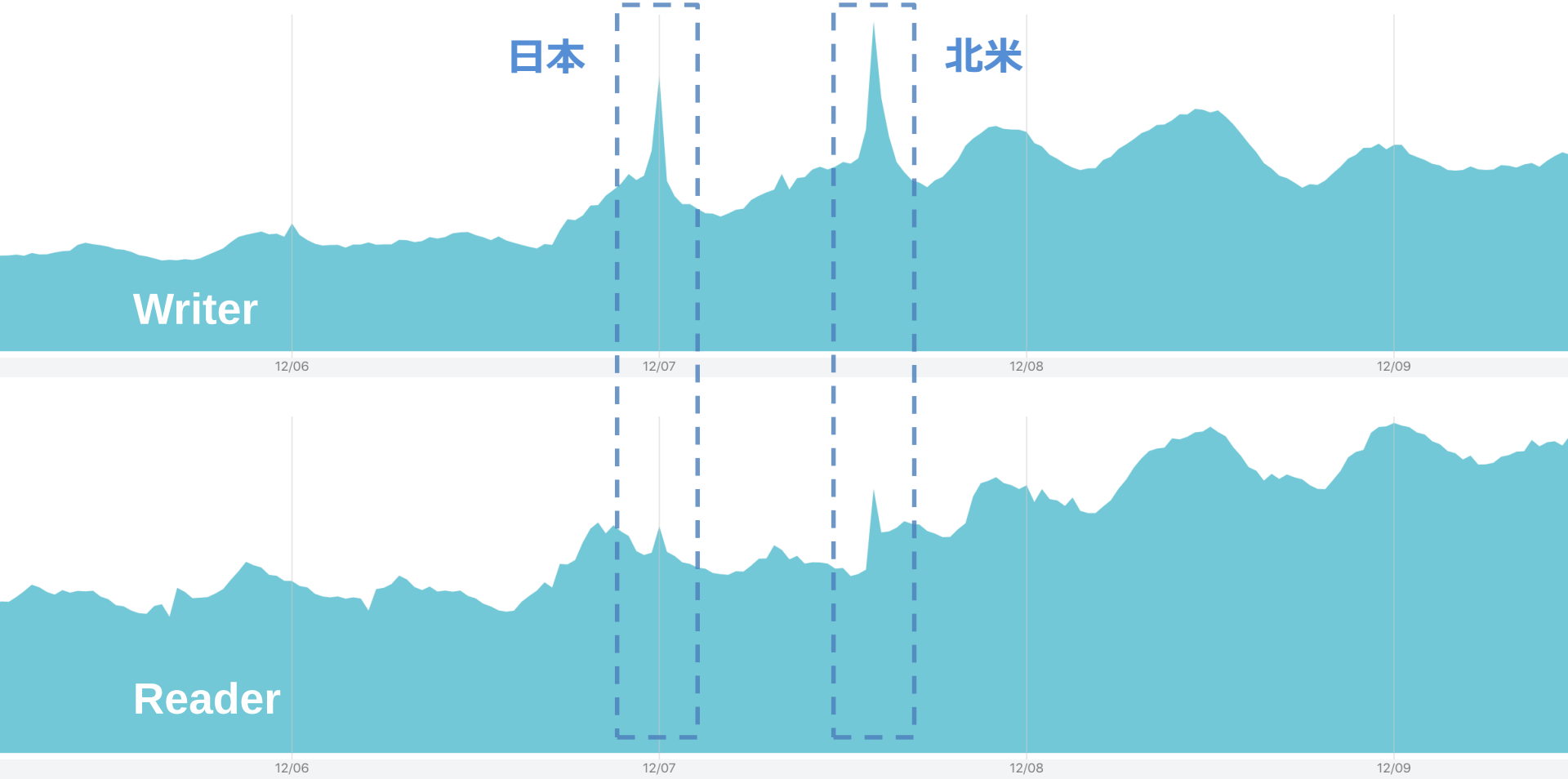
2018年9月19日　Nintendo Switch Online 開始



2018年9月19日　Nintendo Switch Online 開始



2018年12月7日　大乱闘スマッシュブラザーズ SPECIAL 発売



サービス規模

- **加入数 980万アカウント**（2019年4月末時点）
- Twitch Prime 加入者向けキャンペーンも実施中

Aurora を採用してみて

良かった点

- 特に大きな障害もなく、安定して稼働
- 最大のメリットは運用コストの低さ

気になった点

- コネクションプールの利用には工夫が必要
- Zero Downtime Patching 適用可否の検証が難しい

権利管理サーバーの開発

前半のまとめ

前半のまとめ

- Nintendo Switch Online 加入者の権利情報を管理する権利管理サーバーの開発事例を紹介
- Amazon Aurora の採用により高い可用性を実現し、ゲーム特有のアクセスパターンにも柔軟に対応
- サービス規模は順調に成長

セーブデータお預かりサービスの開発

セーブデータお預かりサービスとは？

- サービス名称:
  - セーブデータお預かりサービス（日本）
  - Save Data Cloud（北米）

- 目的: セーブデータのバックアップ
  - Switch 本体が壊れた / なくなったとき
    でもセーブデータは復旧できるように

セーブデータとは？

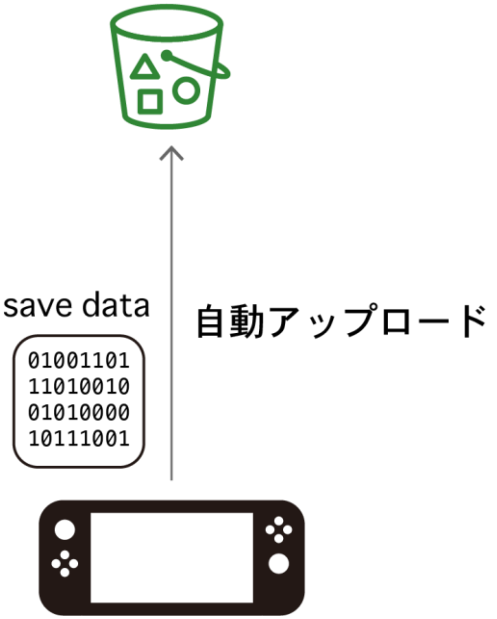
- セーブデータの中身
  - 進捗状況
  - もちもの
  - etc...
- データ構造: ゲームによって様々
- サイズ: 数MBから数GB
- 価値: お客様が数十から数千時間遊んだ記録

バックアップとは？

- どこにバックアップするのか？
  - S3
    - 高い耐久性と可用性に注目
    - Switch から HTTPS で直接アップロードできる
- いつバックアップするのか？
  - 自動で定期的にバックアップ
  - 最新のセーブデータは遊んだあとに必ず

セーブデータお預かりサービスとは

- セーブデータお預かりサービス
  ＝ **セーブデータを S3 に自動で**
  バックアップするサービス



課題

1. セーブデータの一括転送
2. S3 読み書きの権限管理
3. DB の書き込み性能

課題1:セーブデータの一括転送

- 目的: 世界中の回線環境で数GBのデータをアップロードできるように

平均アップロード速度	1GBアップロードにかかる時間
1Mbps	2時間23分
5Mbps	28分
10Mbps	14分

- アップロード中に接続が切れる可能性
  Switch がモバイル回線でインターネット接続されていることも

分割してアップロードする作戦

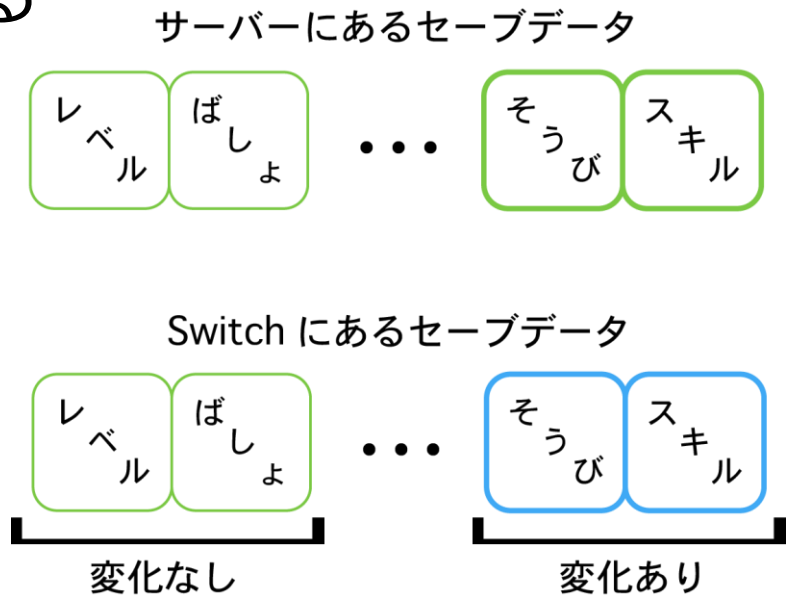
- 対策: セーブデータを分割して管理する
- 例えば1GBのセーブデータを10分割します

平均アップロード速度	100MBアップロードにかかる時間
1Mbps	13分
5Mbps	2分
10Mbps	1分

- 1Mbpsでも13分間接続が維持できれば大丈夫

効率的なアップロードの実現

- セーブデータを分割して管理すると、効率的なアップロードが実現できる
- 変化のあった領域だけサーバーと同期する
（※2回目以降）

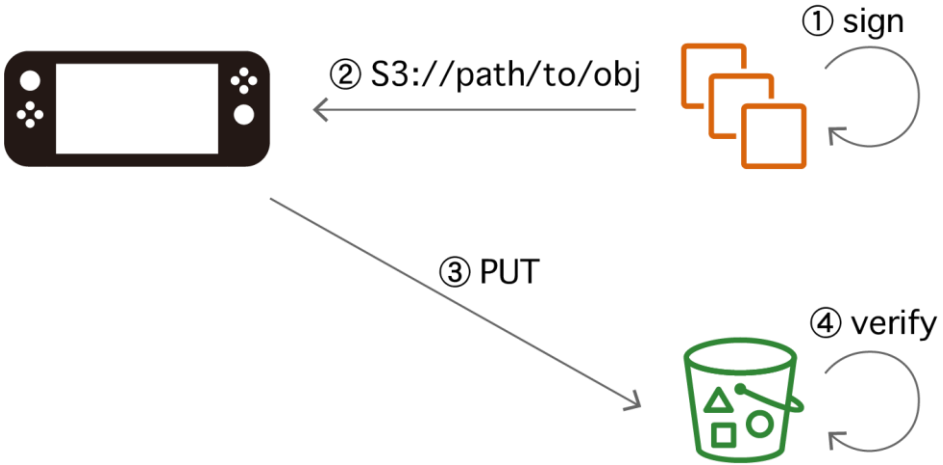


課題2: S3 読み書きの権限管理

- 目的: Switch → S3 bucket の読み書き権限管理
- 機能要件: 読み書きの権限は厳重に管理したい
- 性能要件: 払い出し処理がスパイクに対応できる
  - スパイク例: 人気ゲームの配信時に一斉に
  - 懸念: 読み書き権限を払い出す API の Rate Limit
  - できれば EC2 だけで権限払い出ししたい

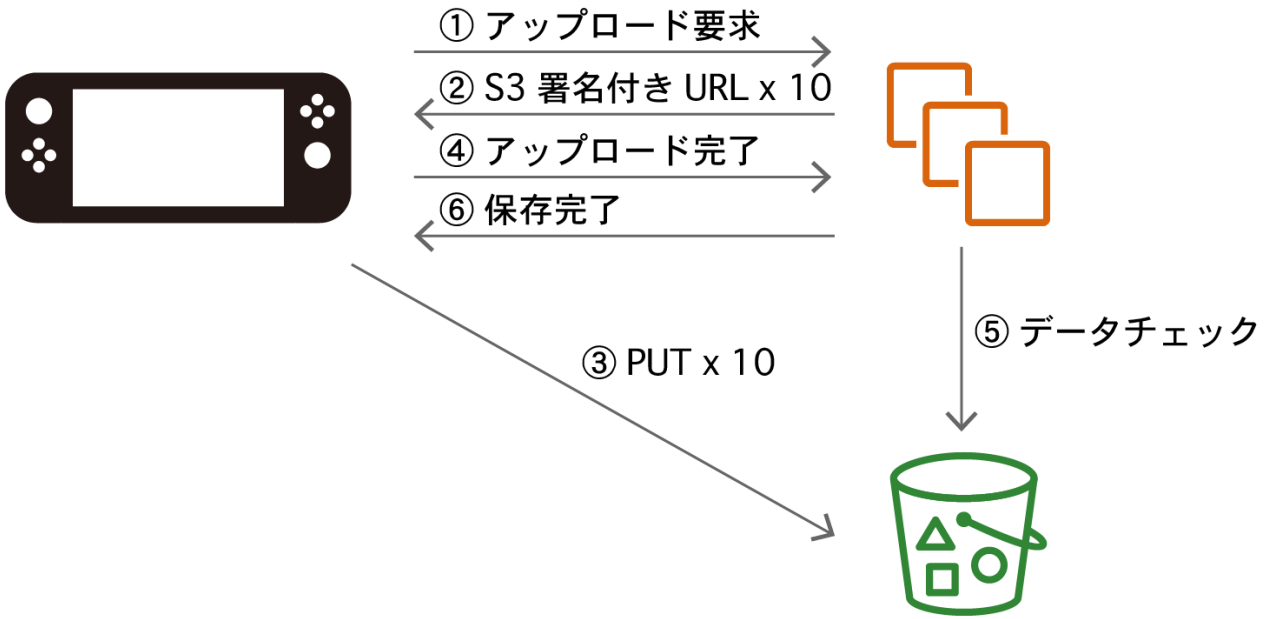
署名付きURLによる権限払い出し

```
s3 = Aws::S3::Resource.new(region:'us-east-1')
obj = s3.bucket('savedata').object('/path/to/file0')
url = obj.presigned_url(:put, expires_in: 3600)
# 署名付きURLをresponseにいれてSwitchにわたす
```



- EC2 内の処理だけで権限を発行できる
- S3 との通信が不要

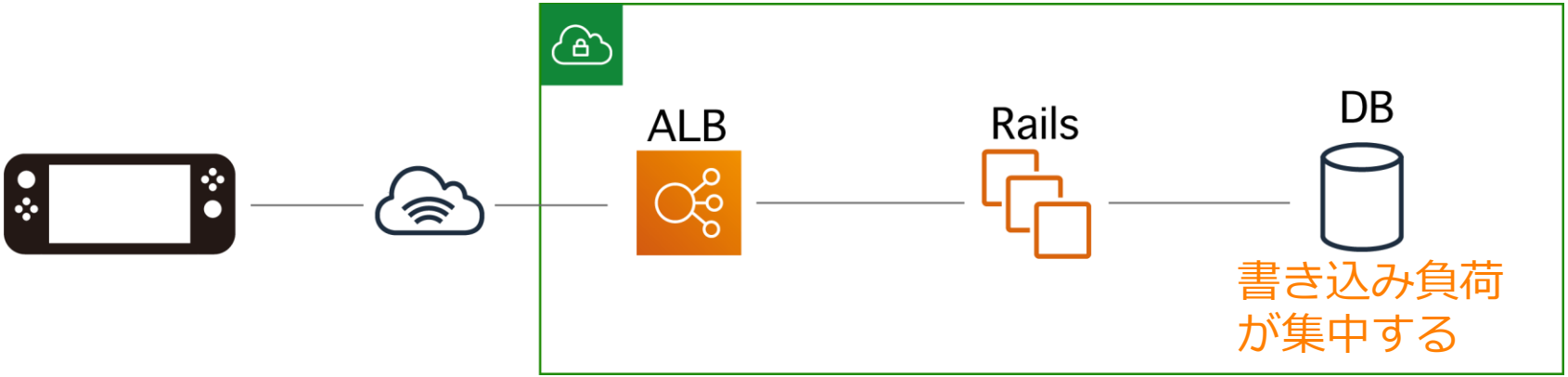
アップロードのしくみ



※分割数: 10とします

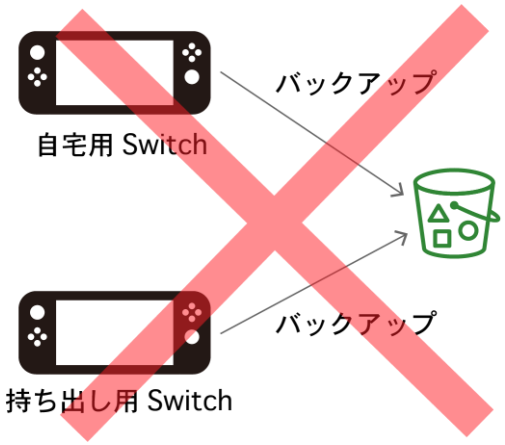
課題3: DB の書き込み性能

- 目的: 大量の書き込みを遅延なく処理
- 書き込み負荷
  - 払い出した S3 Key をすべて INSERT



DB の要件

- ユーザ数: 1000万人以上を想定
- レコード数: 数億〜
- INSERT, UPDATE が多い
- **排他制御必須**
- 数年後もスケールアップで
  サービスできる

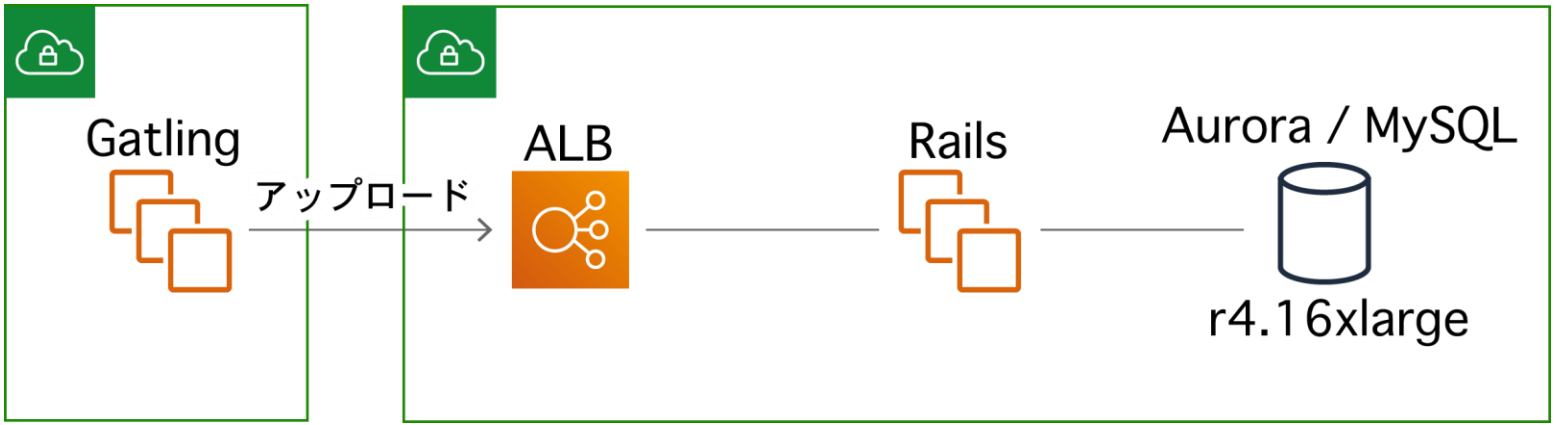


ストレージをSwitch間で共有
⇒ 同時書き込みを禁止

DB の選択肢

- MySQL on RDS
- Aurora on RDS
- Dynamo DB
  - 制約: Rails での開発、排他制御が必要
    → 開発当時には Dynamo DB の
    transaction support がなかった
- MySQL と Aurora で性能を比較

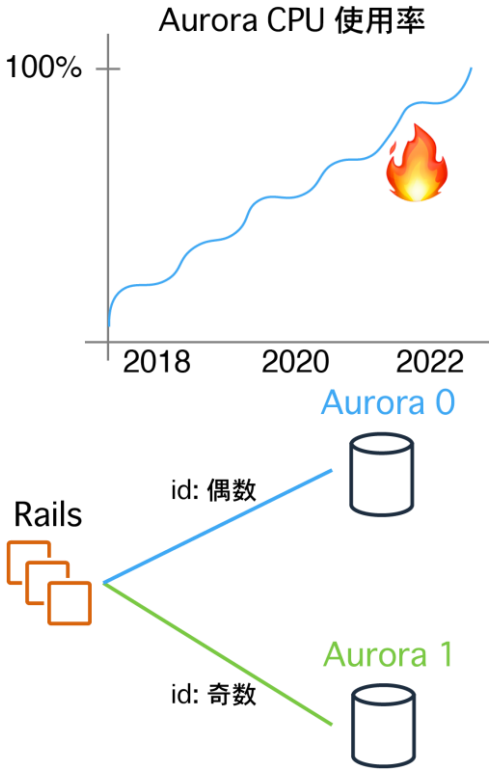
負荷試験で DB を決める



- Gatling でセーブデータをアップロードし、RDS の CPU を使い切るまで負荷をかける
- **結果: Aurora を利用することに決定**

数年後に性能が足りなくなる可能性

- 数年後のwrite性能が心配
- 対策
  - Aurora の性能強化に期待する
    - 例: r5.24xlarge が Aurora に対応
  - シャーディング
    - 例: user.id の偶奇で DB ふりわけ



各アプローチの比較

	pros	cons
性能強化に期待	• DB 1台なので運用が楽	• 性能が強化されるかどうかは不明
シャーディング	• n倍までスケールする	• シャーディングの特殊処理を開発し続けるコスト • システムが複雑になったぶん運用コストが上がる

- 性能強化に賭けても、数年後シャーディングせざるをえない可能性
  → **ローンチ当初からシャーディングしたほうが運用が楽**

どうやってシャーディングをするのか

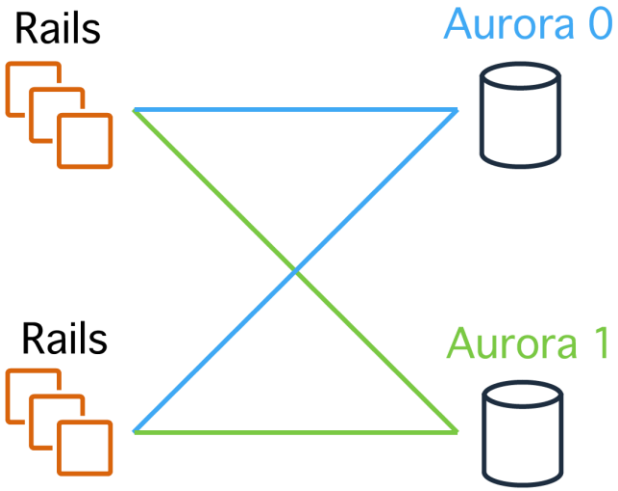
- 制約: Rails は標準機能でシャーディングを
  サポートしていない
- 問題点
  - 何かしらの gem を使うことになる
    - もし、メンテナンスされなくなったら……
  - ユーザー振り分けのロジックを Rails に実装
    - 本来の仕様とは関係のない複雑さ

別解: Rails でシャーディングしない

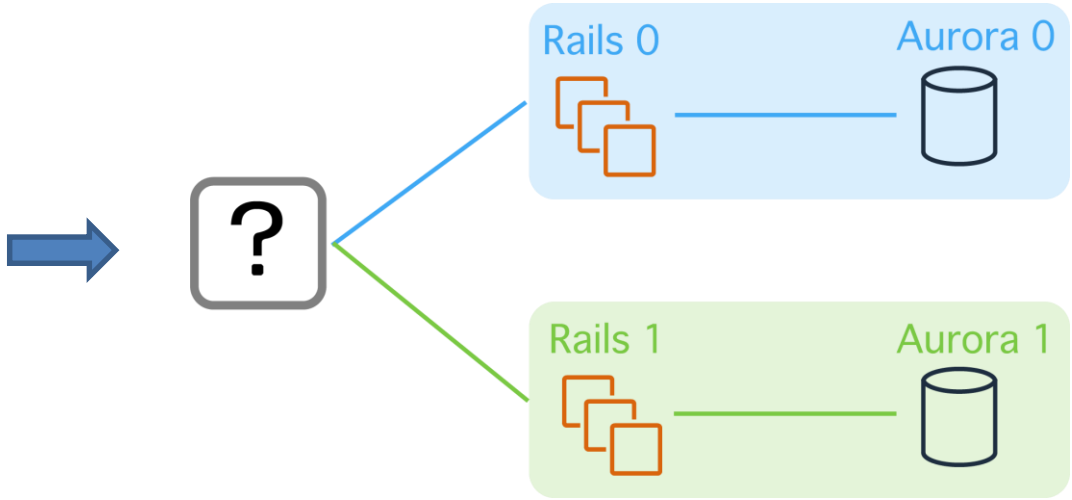
- 目的: 複数 DB 構成でサービスがスケールする
- 要求: アプリケーションサーバーをシンプルに保ちたい
- Rails にふりわけロジックを持たずにユーザーふりわけをしたい

→ **アプリケーションサーバーの前段で ユーザーをふりわける**

別解: Rails でシャーディングしない



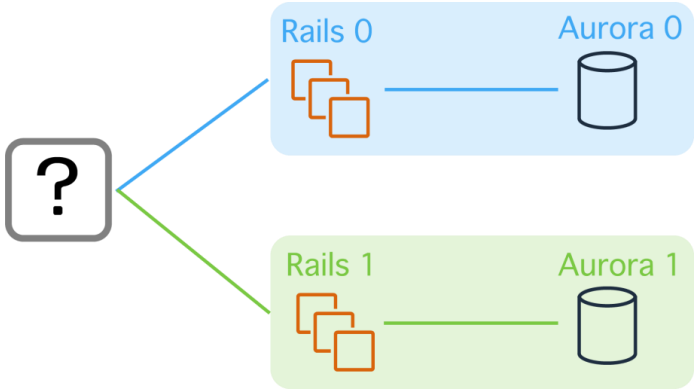
ふつうのシャーディング



Rails でふりわけない方法

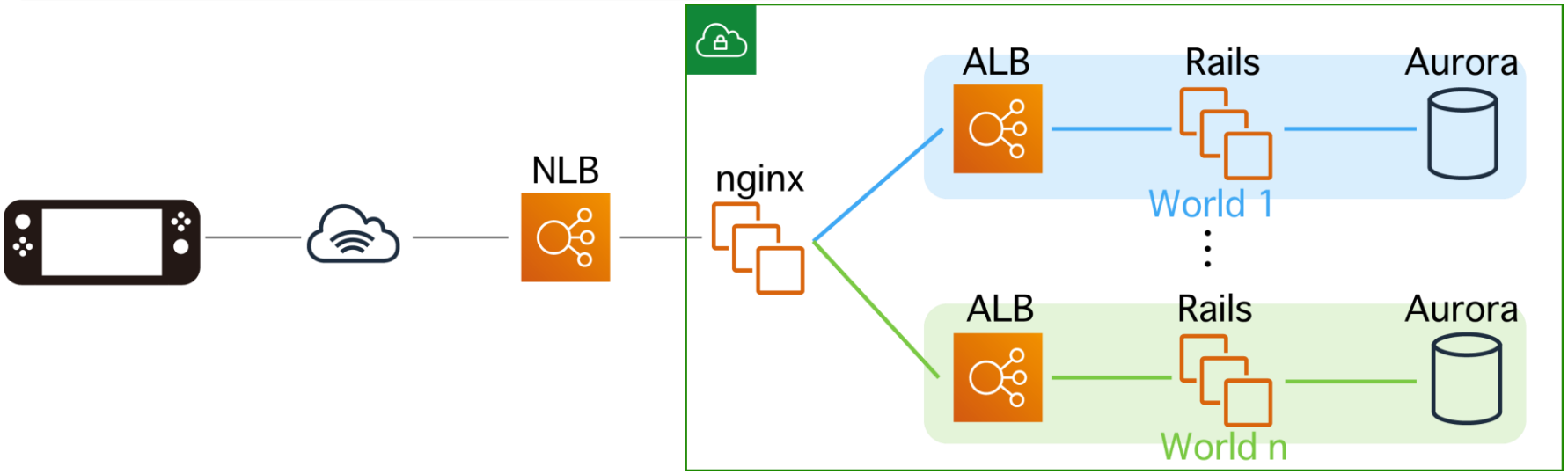
nginx でふりわける

- ？ = nginx
- HTTP Header や query string などにユーザー識別子を入れて複数の backendへふりわけ
- map ディレクティブだけで実現




```
map $http_x_user_id $backend {
    pattern0 backend_url0;
    pattern1 backend_url1;
}
location /hoge/ {
    proxy_pass $backend;
}
```

全体構成



- nginx で TLS 終端、World ふりわけ
- ALB - Rails – Aurora のセット: World と呼ぶ

nginx ワールド分割構成の考察(pros)

- ユーザーふりわけロジックをアプリケーションサーバーから分離できた
  - アプリケーションサーバーは要求仕様に専念
  - ユーザーふりわけは nginx に任せる
- 複数 DB を使いながらシンプルな構成に
  - ALB - Rails - Aurora, NLB - nginx: ごく普通の構成
  - ふつうのシャーディングより日々の運用がやりやすい

nginx ワールド分割構成の考察(cons)

- DB をまたいだ JOIN はできない
  - 今回は不要
- DB 単体運用よりも、運用対象が増えた
  - スキーマ変更、RDS のメンテナンスなど → 自動化
- world を増減させるのが難しい
  - コスト最適化
  - world 数は変えずに RDS スケールアップ / ダウン

後半まとめ

- 課題1: セーブデータの一括転送
  - 対策: 分割アップロード・差分アップロード
- 課題2: S3 読み書きの権限管理
  - 対策: 署名付き URL を EC2 内で大量生成
- 課題3: DBの書き込み性能
  - 対策: 複数 world 構成・nginx でのふりわけ

まとめ

まとめ

- Nintendo Switch Online のシステムの開発事例を紹介
  - 権利管理サーバー
  - セーブデータお預かりサービス
- Nintendo Switch をより便利に、楽しくするために
  - お客様の快適さを重視
  - 長期的にスケールできる設計

We Are Hiring!

- Web エンジニア（Rails, Spring Boot, ...）
- ネットワークインフラエンジニア（Public Cloud, k8s, ...）
- ネットワークサービスシステムエンジニア
- サーバーセキュリティエンジニア
- …

「任天堂 キャリア採用」で検索

https://www.nintendo.co.jp/jobs/career/

ご清聴ありがとうございました