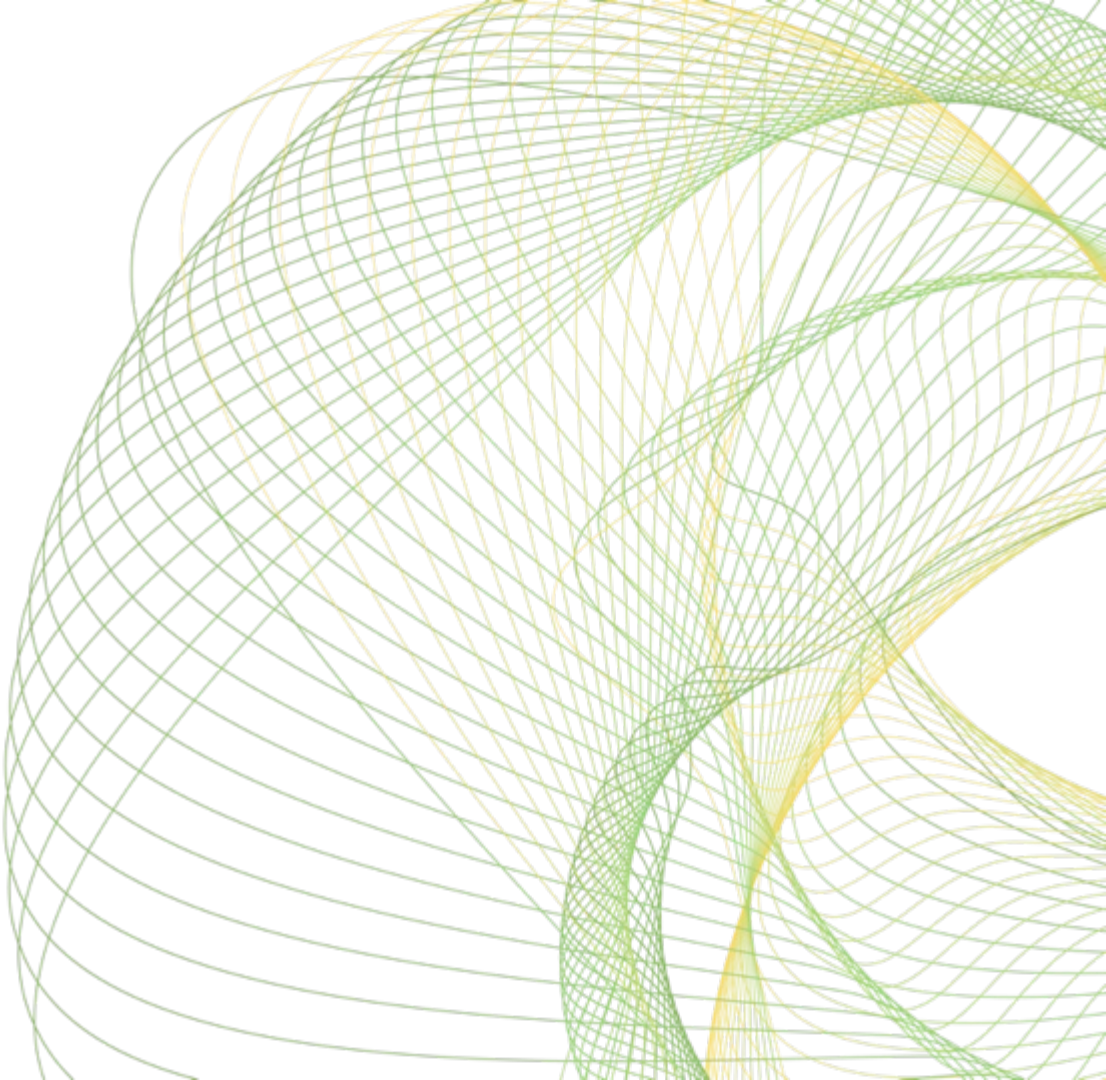
# 課題の回答例



aws financial services

# 本資料の内容についての注意

- この資料は、提案ブートキャンプにて提示された課題の内容に対する解説と、一つの回答例を示すものです。
- この資料に記載されている要件・設計マトリックスの回答やアーキテクチャ図はあくまでも一例です。
  本資料に記載の回答例と一致しない提案についても妥当性が認められる場合は考えられます。
  例えば、本資料の回答例では採用していない「サーバーレス構成」等もひとつの有効な回答となります。
- 本資料の回答例は、2023年2月時点における AWS 公式ドキュメント、
  AWS Well-Architected Framework、その他 参考資料を前提として作成したものとなります。

# 課題内容の整理と深掘り

- 「アプリケーションはコンテナ上で動作するように作られて…(後略)」
  - 開発済みの**コンテナベースのアプリケーションを、低い導入・運用コストで動かす**ためのAWS環境を設計する
- 「アクセス方法やパフォーマンスについては本番相当の構成でテストを実施することが求められています。」
  - 今回のビジネスゴールは、一般ユーザテストを**クラウド上で本番と同じ構成で実施し、ユーザ体験を評価**することであると考えられる。
    例えば、テスト環境では求められることの少ない高可用性や耐障害性といった要素についても考慮する。
    テストの評価は、アプリケーション応答時間などのユーザ体験に直結する項目を重点的に測定することにより行う。
- 「将来的に (中略) 希望する顧客に対して口座情報等と連動した追加サービスを提供する予定です。」
  - 今後取り扱うデータの特性も踏まえ、初期段階から **セキュリティ・監視の面での対策** を考慮する。

**考え方の Point**

- **達成したいビジネスゴール、KPI、業務内容、ならびに取り扱うデータの特性について整理しておく**

# システムの全体像の設計1/2

- 資産管理アプリケーションは、ALB + ECS Fargate + Aurora というシステム構成とする
  - ① コンテナを運用するサービスとして、AWS では ECS or EKS が利用可能。また、計算ノードとしては EC2 or Fargate が利用可能。
    今回は、EKS と比較した ECS の**学習コストの低さ**、EC2 と比較した Fargate の**運用コストの低さ**を狙い、
    **ECS on Fargate を選択**。
  - ② サーバーレス構成は使わず、**既存のオンプレミス環境の運用と比較的近しい形を作ることで、運用負荷の低減を狙う**
- 「ユーザ登録およびログイン」の機能については、Amazon Cognito を利用した認証・認可を行う
  （認証機能を自社開発するとセキュリティリスクと工数増加を伴うため、AWS マネージドサービスを活用して実装する）
- 「追加サービスへの登録」の機能については、Amazon S3 を用いてデータを保存し、Lambda※1 + SNS と SES を用いてメール通知する
  - ① 高可用なオブジェクトストレージとして Amazon S3 を採用し、**保存した画像の高可用性と管理コスト低減**を実現する
  - ② 業務担当者へのメール送信に SNS を活用することで、**自社でメールサーバを管理することなく**、都度イベントが発生するごとにメールを送信する
  - ③ ユーザへのメール送信に SES を活用することで、**大量宛先への効率的なメール送信**が可能(ユーザ側の事前サブスクリプションが不要)

※1 EventBridge + Lambda 構成にすることでオブジェクト名に応じたトリガーの実行等、より細かな制御が可能となります。

# システムの全体像の設計2/2

- 「社内向け管理者画面」の機能については、ALB(WAF) + ECS + Aurora の構成を採用する。
  **WAFのACL機能を用いることで、IPアドレスの制限が可能**となるため、社内拠点に設定された任意のグローバルIPのみからのアクセスを許可するよう設定する

- 「システムのURL」については、CloudFront や ALB 利用時にデフォルトで付与されるドメイン名はAWSが保持するドメイン名となるため、
  **Route 53のパブリックホストゾーンを用いて、カスタムドメインを取得**し、任意のドメインを用いたURLを生成、利用する。

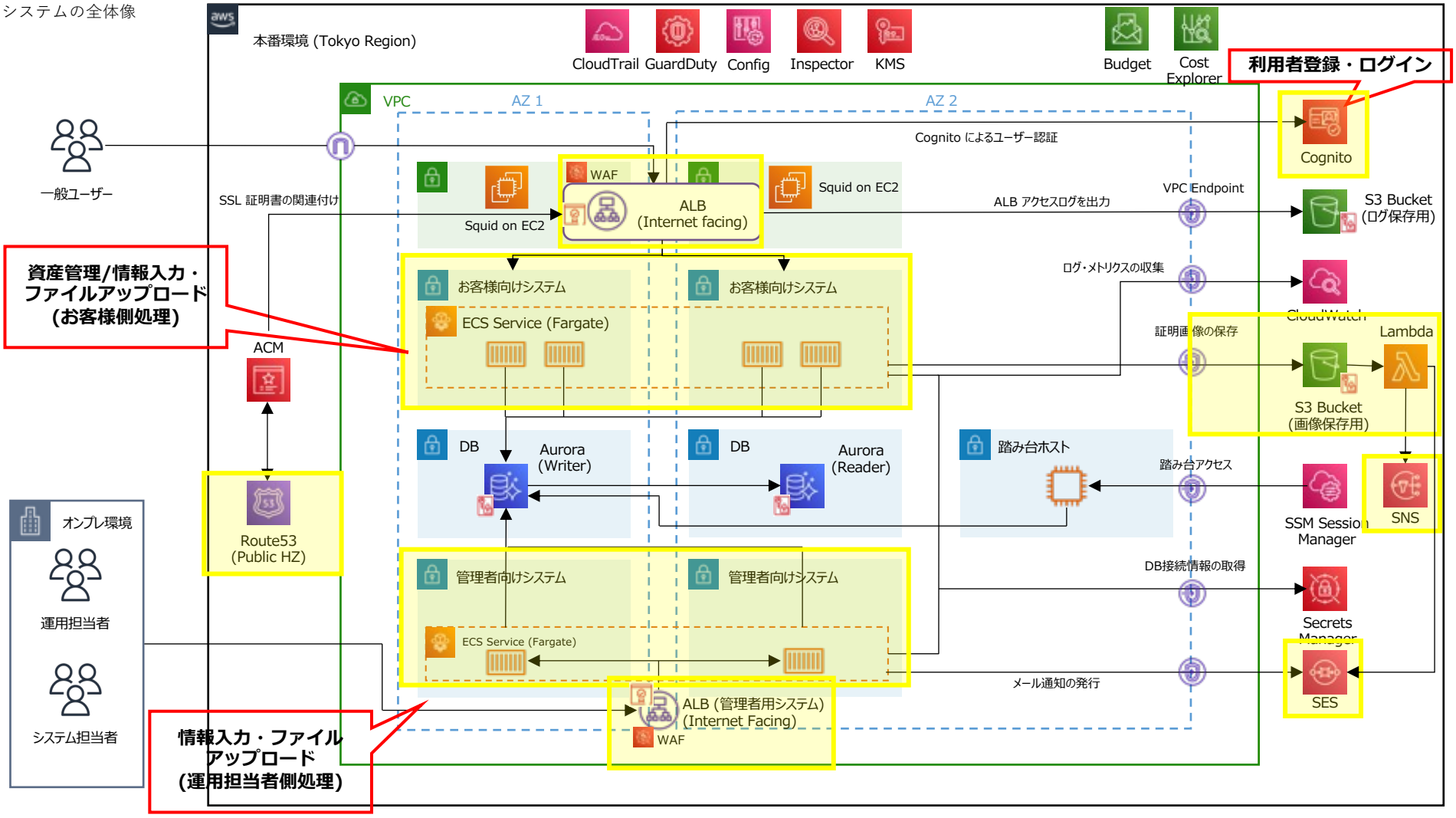
# 要件・設計マトリックス（必須）

| 区分 | 項目 | システム要件 （**自分で記入**） | 設計方針（**自分で記入**） |
|---|---|---|---|
| 機能 | アプリケーションの基本機能 | Web ブラウザを通してアプリケーションへアクセスする。ユーザはアプリケーションを通して、自分の資産情報の入力・保存・参照ができる。開発が完了しているコンテナ上で動作するアプリケーションを利用する | • ALB + **ECS** + Aurora の構成を採用する<br>• お客様情報については Aurora へ保管する |
| 機能 | ユーザ登録とログイン | ユーザの初回アクセス時は、氏名やメールアドレス、生年月日などの情報と、ログインパスワードをシステムに登録する。2回目以降のアクセスでは、メールアドレスとログインパスワードによる認証を行い、認証済みユーザのみがアプリケーションへアクセスできる | • **Amazon Cognito ユーザープール**を用いたユーザー認証機能を実装する<br>• ALB 側で認証を行う。ヘッダー/クエリ文字列/ Cookie は CloudFront からALB へ転送する |
| 機能 | 追加サービスへの登録 | お客様企業の銀行口座を持っているユーザは、システムに口座番号を登録し、身分証明書画像をアップロードすることで、追加サービスの適用を申請することができる。追加サービスの申請完了時と承認時に、登録されたメールアドレスへ通知メールが送付される | • 身分証明画像は **S3** へ保存する<br>• 身分証明画像が S3 へ保存されたときに、**Lambda を起動し、SNSを用いて業務担当者へメール通知**する<br>• 身分証明画像が S3 へ保存されたときに、**Lambda を起動し、SESを用いてユーザへメール通知**する<br>• 業務担当者がシステムから承認実施時に、**SESを用いてユーザへメール通知**する。 |
| 機能 | 社内向け管理者画面 | 社内拠点内からのみアクセスできる管理者画面が用意されていること | • 管理者画面のインフラとしては、ALB + ECS + Aurora の構成を採用する。**WAFによるIP制御**を行う。 |
| 機能 | システムのURL | カスタムドメインを用いたURLが設定されていること | • **Route 53のパブリックホストゾーン**を用いて、**カスタムドメインを取得**し、任意のドメインを用いたURLを生成、利用する。 |



aws financial services

システムの全体像


本番環境 (Tokyo Region)
CloudTrail
GuardDuty
Config
Inspector
KMS
Budget
Cost Explorer
利用者登録・ログイン
VPC
AZ 1
AZ 2
一般ユーザー
Cognito によるユーザー認証
Cognito
WAF
ALB (Internet facing)
Squid on EC2
Squid on EC2
SSL 証明書の関連付け
ALB アクセスログを出力
VPC Endpoint
S3 Bucket (ログ保存用)
資産管理/情報入力・ファイルアップロード (お客様側処理)
お客様向けシステム
お客様向けシステム
ECS Service (Fargate)
ログ・メトリクスの収集
CloudWatch
証明画像の保存
Lambda
ACM
S3 Bucket (画像保存用)
DB
Aurora (Writer)
DB
Aurora (Reader)
踏み台ホスト
踏み台アクセス
SSM Session Manager
SNS
Route53 (Public HZ)
オンプレ環境
運用担当者
管理者向けシステム
管理者向けシステム
DB接続情報の取得
Secrets Manager
ECS Service (Fargate)
メール通知の発行
SES
システム担当者
ALB (管理者用システム) (Internet Facing)
WAF
情報入力・ファイルアップロード (運用担当者側処理)

# メリット・デメリットの比較表

コンテナを動作させるコンピュートノードとして、Fargate 方式と EC2 方式を比較した。
結果として、運用性に優れた Fargate 方式を選択した。

| | | ECS + Fargate 方式 | | ECS + EC2 方式 | |
|---|---|---|---|---|---|
| 方式の説明 | | コンテナを動作させるノードとして AWS Fargate を利用する方式 | | コンテナを動作させるノードとして Amazon EC2 を利用する方式 | |
| 比較観点 | 観点 1 | △ | サービス利用料 | ○ | サービス利用料 |
| | 観点 2 | ○ | 運用コスト | × | 運用コスト |
| | 観点 3 | △ | カスタマイズ性 | ○ | カスタマイズ性 |

# 可用性・耐障害性とセキュリティについて検討する

- AWS Well-Architected フレームワーク – 信頼性の柱「可用性 99.9% の実装例」[1] を参考にアーキテクチャを検討する
  - テスト段階ではあるが、本番稼働と同等の可用性を 99.9% と仮定し、この可用性目標をターゲットとする
    - ALB + ECS Fargate (**Auto Scaling**) + Aurora リードレプリカの活用
    - **マルチ AZ 配置**

- セキュリティ対策については、 AWS Well-Architected フレームワーク – セキュリティの柱 [2] を参考にアーキテクチャを検討する
  - **データ保護**: Aurora, S3 上の保管データは KMS カスタマー管理キー (CMK) で暗号化、転送中のデータは TLS で暗号化
  - **インフラ保護**: アプリケーションレイヤーのファイアウォール (**WAF**) の導入、**脆弱性診断(ソフトウェア)**の実施
  - **脅威の検出**: CloudTrail, Config を用いた**操作・変更監視**と各種サービスの**ログ監視**

- (今回のワークショップでは範囲とならないが) 実際に障害やセキュリティインシデントが発生した事態を想定した**訓練を行う**

**考え方の Point**

- **可用性については、AWS Well-Architected フレームワーク – 信頼性の柱 を参照する**
- **セキュリティ目標の達成については、AWS Well-Architected フレームワーク – セキュリティの柱 を参照する**

[1] https://docs.aws.amazon.com/ja_jp/wellarchitected/latest/reliability-pillar/s-99.9-scenario.html
[2] https://docs.aws.amazon.com/wellarchitected/latest/security-pillar/welcome.html

# 要件・設計マトリックス（必須）

| 区分 | 項目 | システム要件 （**自分で記入**） | 設計方針（**自分で記入**） |
|---|---|---|---|
| 非機能 | 可用性・スケーラビリティ | 特定要素の単一障害により一般ユーザテストが中断しないよう、構成要素を冗長化する。アクセス数の増加に応じたスケールアップ／スケールアウトが可能な構成とする。 | • 「可用性 99.9% の実装例」をベースに設計検討する<br>• ALB + ECS + Aurora の構成<br>• ECS Fargate を用いた**自動スケーリング**<br>• Aurora の **MAZ 配置** |
| 非機能 | バックアップ | 利用者が登録したデータは毎日自動でバックアップ する | • **Aurora の自動バックアップ**、および **S3 のバージョニング設定**の有効化により、個人情報データのバックアップを行う |
| 非機能 | データ保護 | 個人情報を扱うため、転送中および保管時の両面でデータの安全性を保つ対策を講じる | • **ACM で発行した SSL/TLS 証明書**を CloudFront/ALB に関連付け、**HTTPS アクセス**を有効化する<br>• Aurora, S3 に保存された機密データに対しては **KMS カスタマー管理キー**により暗号化する |
| 非機能 | インフラ保護 | パッチ適用や脆弱性診断など、サーバーやアプリケーションを保護する対策を講じる | • DB 接続情報は **Secrets Manager** に保存して利用する<br>• **Inspectorを用いてコンテナに含まれるソフトウェアの脆弱性チェック**の仕組みを取り入れる |

※補足
バックアップでは、AWS Backupを用いたバックアップ取得を行うことも可能です。
AWS Backupの利用により、バックアップの取得やリストアの実施をアカウント内で一元的に管理できるメリットが得られます。
ただし、AWS Backupでは、AuroraのPITRがサポートされないなど、一部制約事項に注意する必要があります。



aws financial services

# 要件・設計マトリックス（必須）

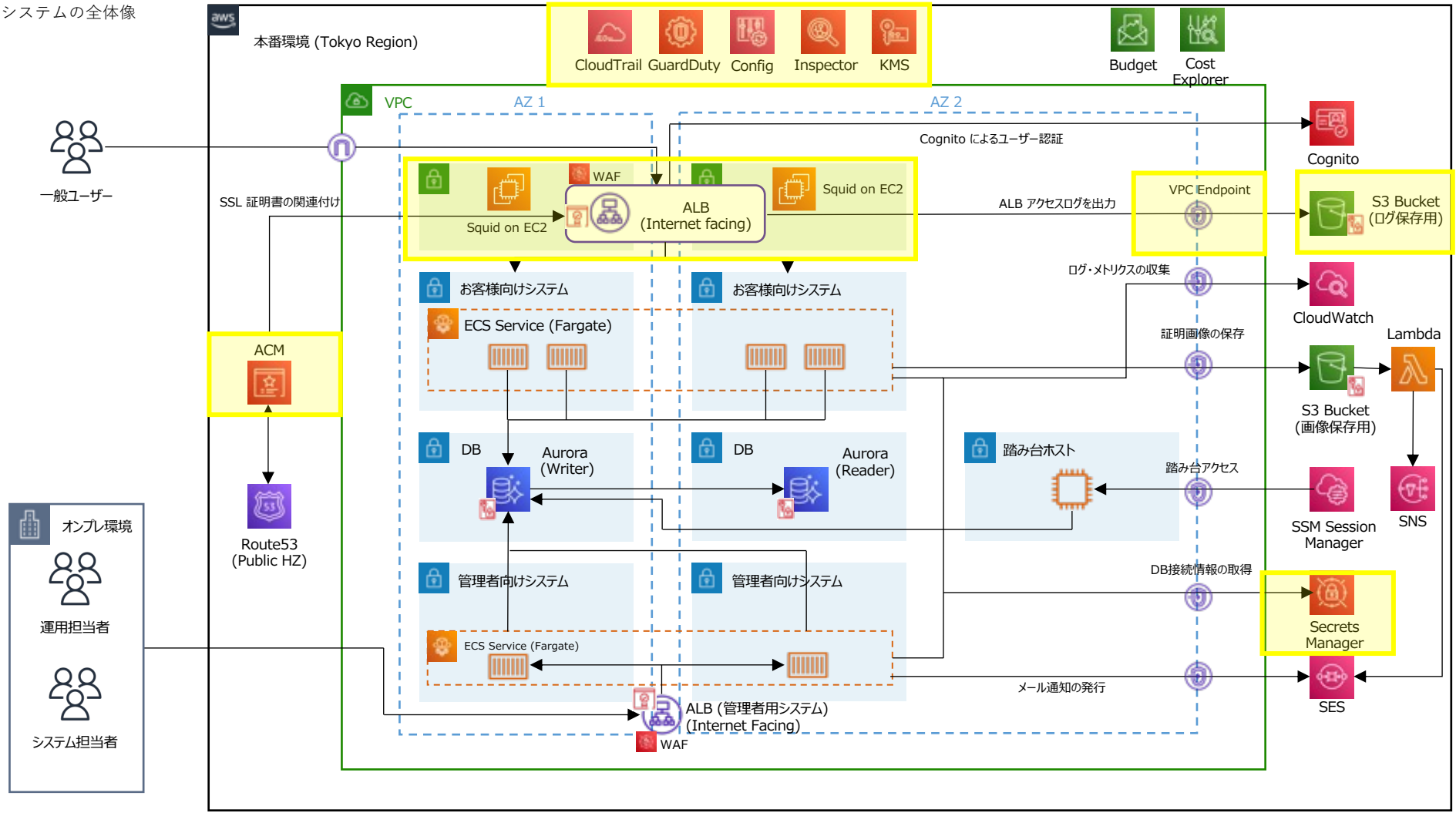
| 区分 | 項目 | システム要件 （**自分で記入**） | 設計方針（**自分で記入**） |
|---|---|---|---|
| 非機能 | ネットワーク保護 | ファイアウォールなど、ネットワークを保護する対策を講じる。加えて、クラウド外の通信については、プロキシサーバーを経由することでURLフィルタリングを実施する。 | • **ALB に AWS WAF を関連づけ**、SQL Injection などの外部攻撃からシステムを保護する<br>• **Squid on EC2**によりAWSクラウド外への通信をURLフィルタリングを実施する<br>• VPC外へのAWSサービスへの通信については、**VPCエンドポイント**による通信を行う |
| 非機能 | ログの収集・保管 | システム管理者は障害発生時等に、収集・保管されたログを参照する。その他、利用者やシステム管理者、業務担当者がシステムにアクセスした形跡を保管する | • **CloudTrail, AWS Config** を用いて環境内のログを収集し、**VPC Flowlog や ALB アクセスログ**を別途 保存する |

※補足
ネットワーク保護：Squid on EC2の代わりに、AWS Network Firewallを用いる方式も考えられます。Network Firewallでは制御方式がドメイン単位(URL指定は不可)となる点に留意してください。



aws financial services

システムの全体像


本番環境 (Tokyo Region)
CloudTrail
GuardDuty
Config
Inspector
KMS
Budget
Cost Explorer
VPC
AZ 1
AZ 2
一般ユーザー
SSL 証明書の関連付け
WAF
Squid on EC2
ALB (Internet facing)
Squid on EC2
Cognito によるユーザー認証
Cognito
ALB アクセスログを出力
VPC Endpoint
S3 Bucket (ログ保存用)
ログ・メトリクスの収集
CloudWatch
お客様向けシステム
お客様向けシステム
ECS Service (Fargate)
証明画像の保存
Lambda
ACM
S3 Bucket (画像保存用)
DB
Aurora (Writer)
DB
Aurora (Reader)
踏み台ホスト
踏み台アクセス
SSM Session Manager
SNS
Route53 (Public HZ)
DB接続情報の取得
管理者向けシステム
管理者向けシステム
Secrets Manager
ECS Service (Fargate)
メール通知の発行
SES
ALB (管理者用システム) (Internet Facing)
WAF
オンプレ環境
運用担当者
システム担当者

# システム監視とデータの可視化

- システム管理者が行う監視業務を AWS 上で実現する
  - **システム稼働状況の監視**: CloudWatch ダッシュボードを用いたメトリクスの監視
  - **システム障害の検出**: CloudWatch によるサーバのメトリクス監視とアラームを用いた異常通知
  - **ログの分析**: 収集したログを S3 から Athena を用いて分析、QuickSight で可視化
  - **セキュリティインシデントの監視**: ログ監視に加え、GuardDuty を用いた**ふるまい検知の活用**
  - **各種通知**: Simple Notification Service (SNS) を用いたメール等の通知
- 業務担当者が行う業務を AWS 上で実現する
  - **利用状況等の分析**: ログや DB 上のデータを QuickSightで**直接データソースにアクセスすることなく可視化**
  - **クラウド利用料の把握と制限**: AWS Budgets による予算管理と超過通知、Cost Explorer によるコスト分析

**考え方の Point**

- **ログ・メトリクスの両面の監視を行い、異常が発生した場合に自動で管理者に通知するようにする**
- **AWS のマネージドサービスを用いて監視業務の運用負荷を減らす**
- **業務担当者に対しては、データストアに直接アクセスする代わりに、ダッシュボードからクエリを実行したり KPI を閲覧可能にする**

# ユーザテストにおける効果の測定

- ユーザテストにおいては、サーバの CPU 使用率など、ユーザ体験に直接結びつかないメトリクスやログを監視しても効果は薄い。
  **ユーザ視点でのアプリケーション監視**を導入し、システムの可用性向上に加えて、**ユーザ体験の定量評価**を実現する

- **フロントエンドモニタリング**を導入する
  利用者から見えるパフォーマンスをモニタリングする手法で、Synthetic 監視と Real User 監視の2種類に分類される
  - **Synthetic 監視**:
    外形監視とも呼ばれ、**監視対象のアプリケーションにリクエストを送信**し、HTTP のステータスコード監視や、
    アプリケーションの応答時間の測定など、ユーザ側から見えるアプリケーションの挙動の監視を行う。
    AWS では CloudWatch Synthetics が使用できる
  - **Real User 監視**:
    **実ユーザ側にモニタリングコードを埋め込み**、ユーザがどの順番でアプリケーションを使ったかの測定などを行う。
    AWS では CloudWatch Real User Monitoring (RUM) が使用できる

**考え方の Point**

- **ユーザテストでは、ユーザー目線のサービス監視が最重要。ユーザ視点での監視として、フロントエンドモニタリングの導入を検討する**
- **ユーザから見えるアプリの外形の監視と、ユーザが実際にアプリをどう使用したかの監視を組み合わせて行う**

# 要件・設計マトリックス（必須）

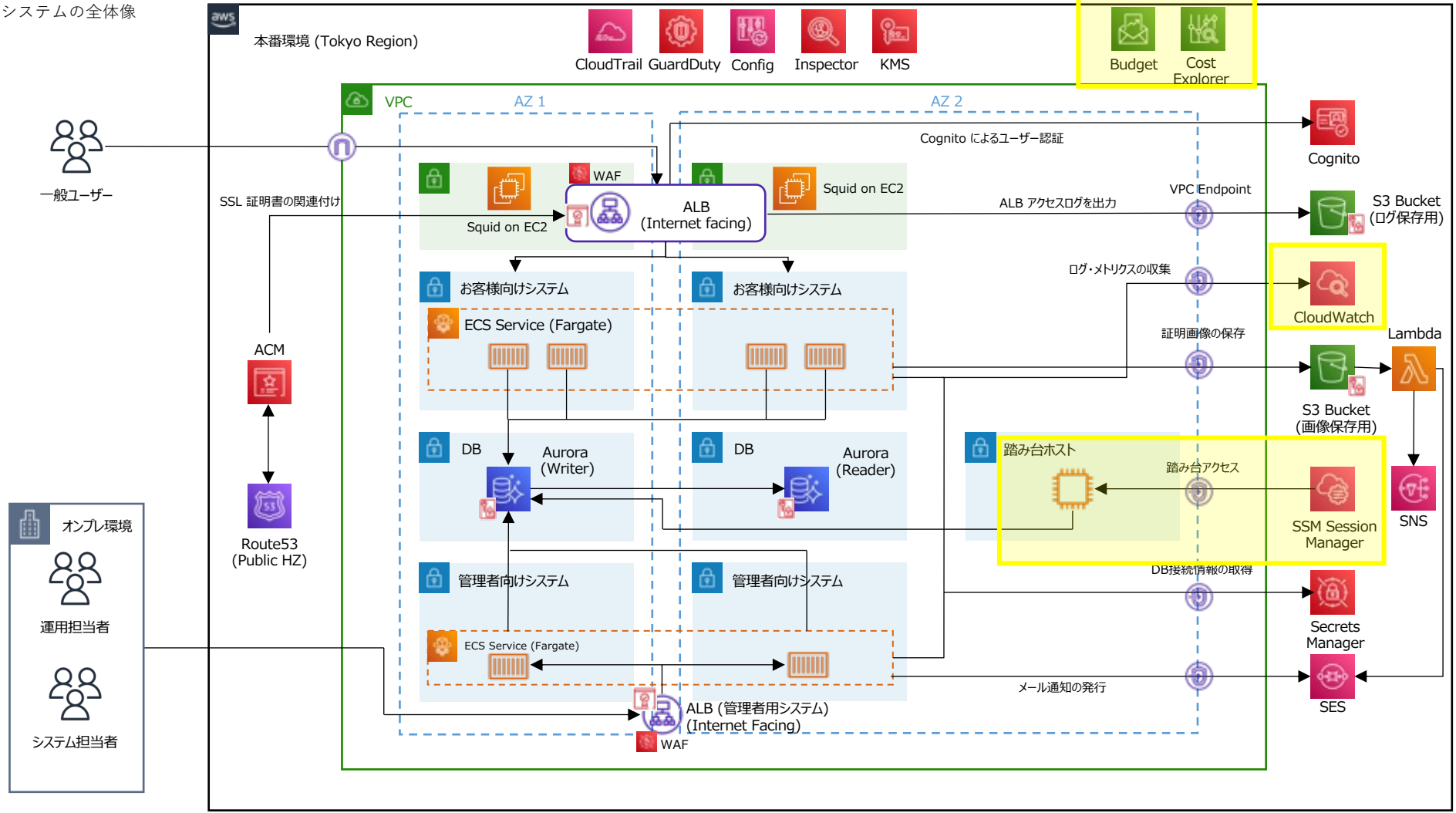
*1: 追加検討課題の取り組みは任意です。もし余力があれば「7.追加検討課題」から1-2つの課題を選び、検討・提案してください

| 区分 | 項目 | システム要件 （**自分で記入**） | 設計方針（**自分で記入**） |
|---|---|---|---|
| 開発・運用 | システム運行状況の把握 | システム管理者が、ダッシュボードなどを通じてシステムの稼働状況を確認できる。システムに障害が発生した場合や、セキュリティインシデントの発生時に、システム管理者へ通知を送ることができる | • **CloudWatch を用いたログ・メトリクス監視**を行う<br>• **Athena + QuickSight によるログバケットの検索・可視化**を可能にする<br>• **GuardDuty を有効化**し、収集したログ情報から不正なアクティビティを検知する<br>• インシデント発生時は担当者へメール通知する |
| 開発・運用 | システム利用状況の確認 | ユーザ登録者数や最終ログイン時刻を確認できる。クラウド利用料金が一定値を超えた場合、業務担当者に通知を送ることができる | • **QuickSight によるアプリケーションデータの可視化**を可能にする<br>• AWS Budgets に予算を設定し、予算を超えそうな場合は担当者へ通知する<br>• Cost Explorer を用いて、クラウドリソースの使用量について把握する |
| 開発・運用 | 運用業務タスク負荷軽減 | パッチ適用、DB バックアップ、DB メンテナンスなどの運用業務の自動化・負荷低減の仕組みを準備 | • OS やミドルウェアに対するパッチ適用：<br>AWS Fargate のプラットフォーム側パッチ適用は AWS の責任範囲であるため考慮不要<br>• データベースに対するパッチ適用：<br>Aurora のセキュリティパッチをメンテナンスウィンドウ期間に自動適用する<br>• データベースのメンテナンス：<br>一時的に起動した踏み台ホストへ SMS Session Manager 経由で接続し DB メンテナンスを行う |

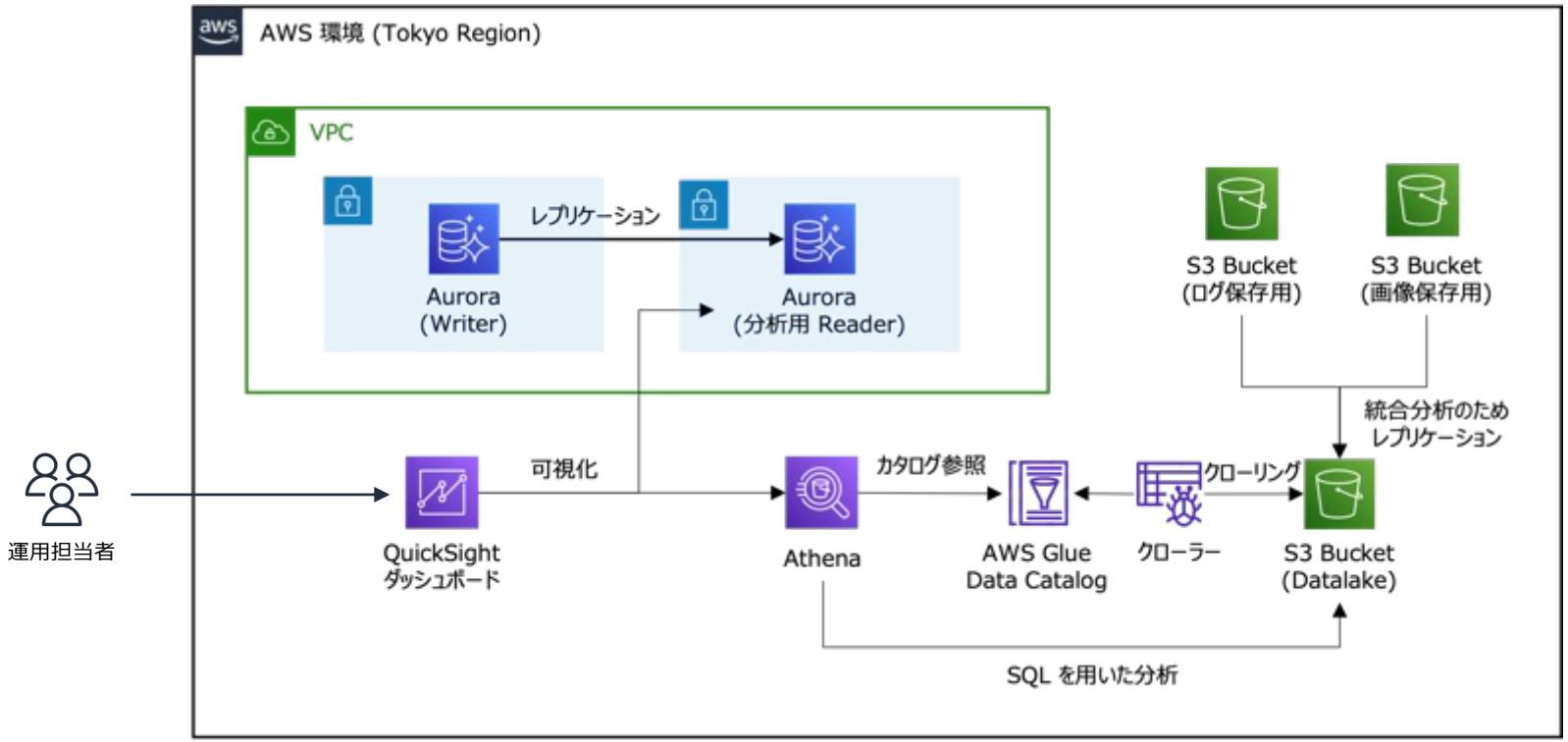


aws financial services

システムの全体像


本番環境 (Tokyo Region)
CloudTrail
GuardDuty
Config
Inspector
KMS
Budget
Cost Explorer
VPC
AZ 1
AZ 2
一般ユーザー
SSL 証明書の関連付け
ACM
Route53 (Public HZ)
WAF
Squid on EC2
ALB (Internet facing)
Squid on EC2
Cognito によるユーザー認証
Cognito
VPC Endpoint
ALB アクセスログを出力
S3 Bucket (ログ保存用)
ログ・メトリクスの収集
CloudWatch
お客様向けシステム
お客様向けシステム
ECS Service (Fargate)
証明画像の保存
Lambda
S3 Bucket (画像保存用)
DB
Aurora (Writer)
DB
Aurora (Reader)
踏み台ホスト
踏み台アクセス
SSM Session Manager
SNS
オンプレ環境
運用担当者
システム担当者
管理者向けシステム
管理者向けシステム
DB接続情報の取得
Secrets Manager
ECS Service (Fargate)
メール通知の発行
SES
ALB (管理者用システム) (Internet Facing)
WAF

# データ分析基盤のアーキテクチャ例 (補足)



# 要件・設計マトリックス（必須）

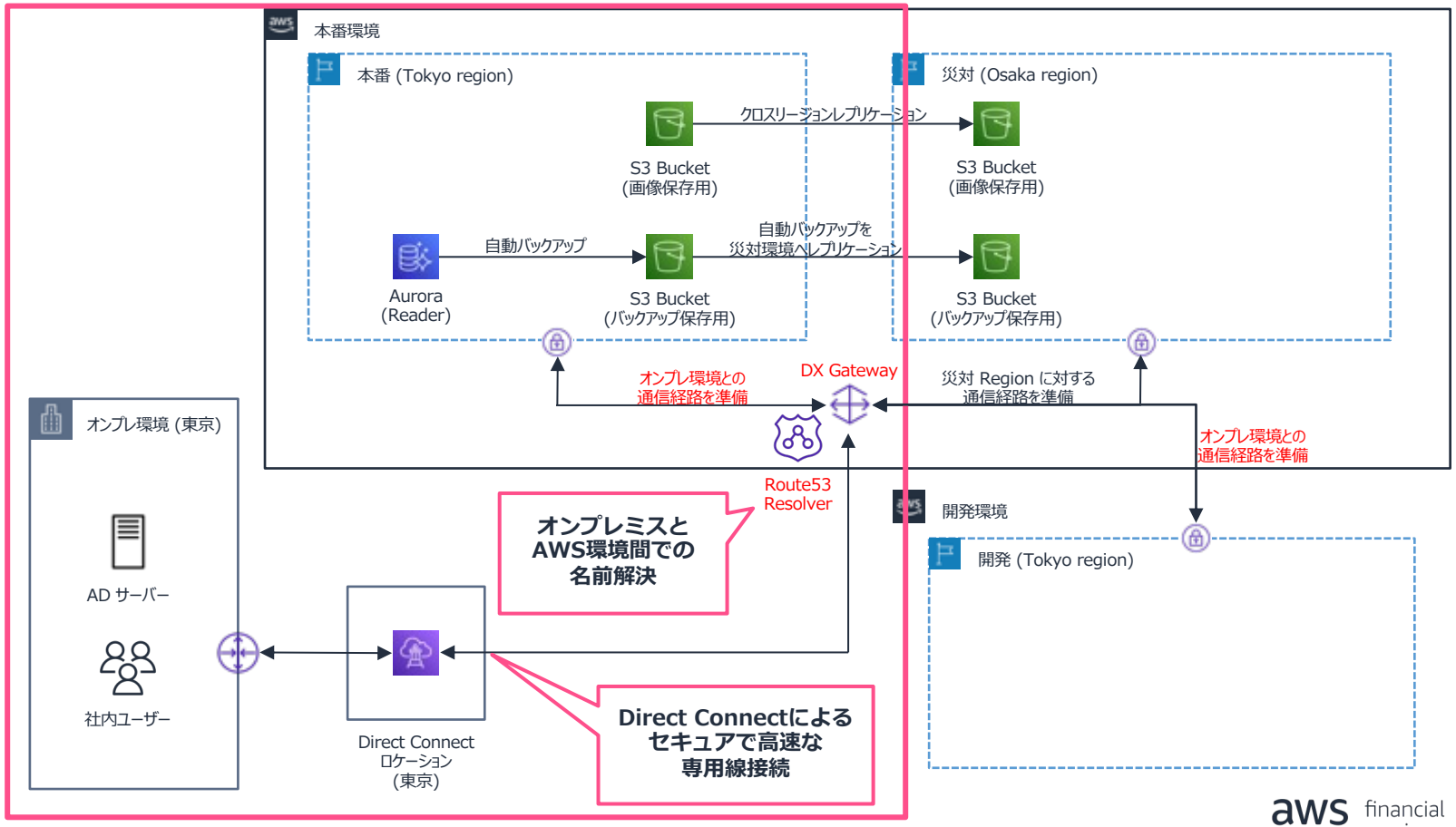
*1: 追加検討課題の取り組みは任意です。もし余力があれば「7.追加検討課題」から1-2つの課題を選び、検討・提案してください

| 区分 | 項目 | システム要件 （**自分で記入**） | 設計方針（**自分で記入**） |
|---|---|---|---|
| 追加検討課題 ① *1 | オンプレミスデータセンターとの接続 | 将来的にはオンプレミスデータセンターとの接続を予定しています。これを実現するためのアーキテクチャを提案してください | • **Direct Connect** を用いてオンプレミスDCと接続する。その際、**Route53 Resolver** を使って名前解決を行う<br>• オンプレミス - AWS 間の接続については Direct Connect から DX Gateway を経由し、Tokyo region の Virtual Private Gateway へ接続する |
| 追加検討課題 ② *1 | 大規模障害に向けたバックアップ | 大規模障害(リージョンレベルの障害)発生に備え、データを定期的に別リージョンへバックアップする。 | • Amazon Auroraの自動バックアップにより、S3へデータの保存<br>• **クロスリージョンレプリケーション**により、別リージョンのS3バケットにデータのバックアップを行う |



aws financial services

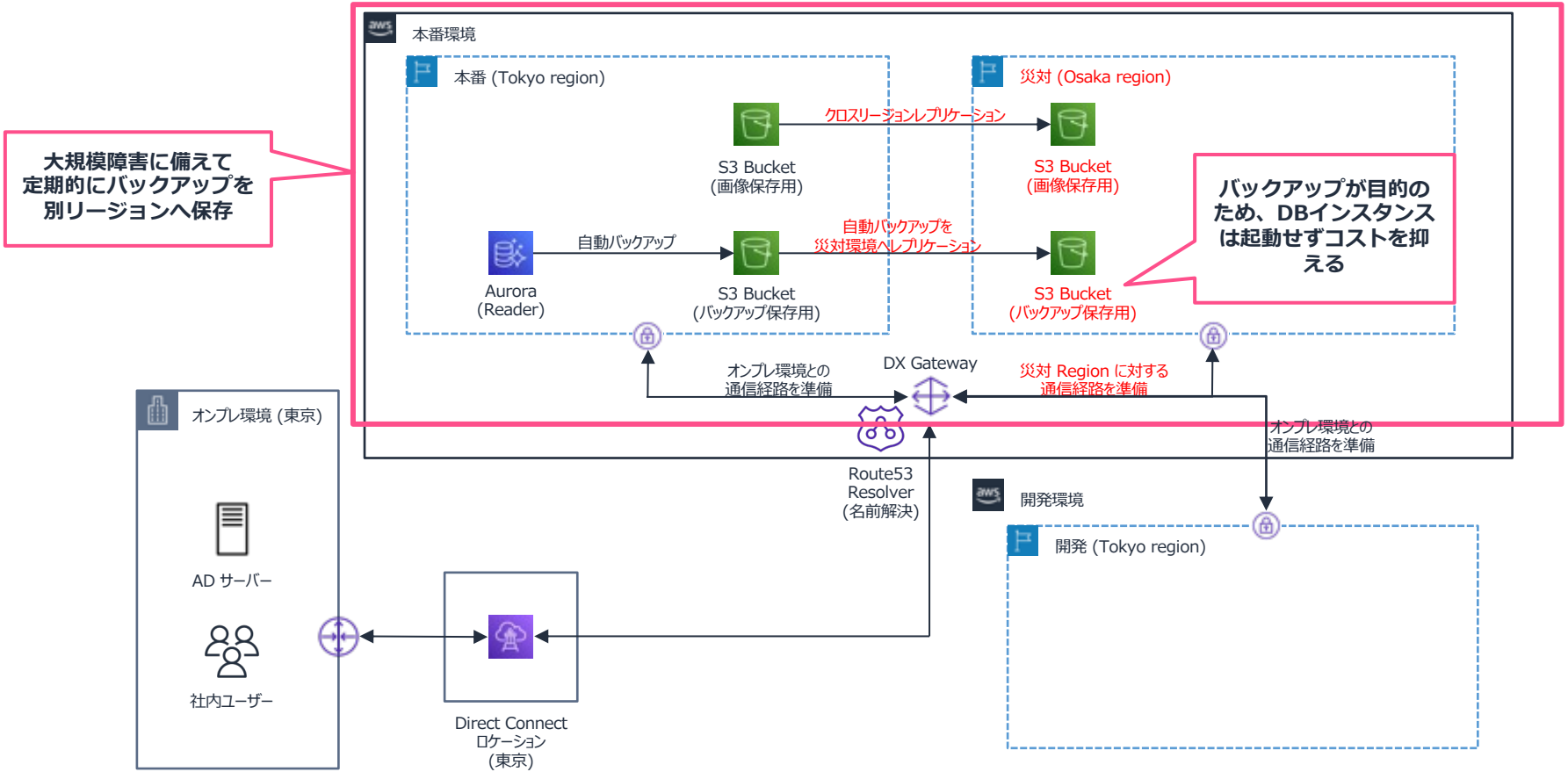
# オンプレミスデータセンターとの接続と大規模障害対策のアーキテクチャ例 - 追加課題 1・2






aws financial services

# オンプレミスデータセンターとの接続と大規模障害対策のアーキテクチャ例 - 追加課題 1・2






aws financial services

# 開発・本番環境のスムーズな連携と 脆弱性・脅威のモニタリング - **追加課題 3・4**

- 「テスト済みのアップデートをテスト環境、ステージング環境、および本番環境へ迅速に反映する…(後略) 」(課題３)
  「高度なセキュリティ監視が必要です。アプリケーションに含まれる脆弱性や脅威をモニタリングする方法…(後略) 」(課題４)
  - 重要な検討ポイントは、(1) 複数環境間の連携、(2) テスト済みアップデートの迅速な反映、 (3) 脆弱性・脅威のモニタリング の３点
- 理想的には、(i) 開発環境、(ii) テスト環境、(iii) ステージング環境、(iv) 本番環境の4面 が用意されていることが望ましい
  - 今回の課題ではアカウントに関する記述が無いため追加のヒアリングが必要だが、１例として以下のようにアカウントを分離する
    (i) 開発環境 ～ (iii) ステージング環境を開発用アカウントに、(iv) 本番環境を本番用アカウントでホストする
- テスト・承認済み、かつ 脆弱性のチェックを行ったアップデートのみを本番環境へ展開する
  - 本番環境でのアップデートの展開時には責任者の承認プロセスを取り入れ、不適切なアップデートによる本番影響を抑制する
  - インフラやネットワーク等のセキュリティ対策に加え、アプリケーションの 開発-デプロイ の中でも脆弱性チェックを行う
    （参考）金融サービス向けの CI/CD パイプラインについては、NRI 様の JAWS DAYS 2020 登壇資料 が詳しい
- その他の脅威のモニタリングやセキュリティ対策としては、GuardDutyによる脅威検出や Security Hubによる様々なセキュリティチェックの一元化とアラートの自動化 (Amazon Macie による個人情報・機密データ検出、ログの集約とアカウントの分離も考えられるが、要検討)

**考え方の Point**

- **これまでの検討と同様に、CI/CD パイプラインについても、お客様の期待値や要望から逆算して、重要な論点を洗い出すことが必要**
- **どういう環境をいくつ利用するか、リリースパイプラインの中でどのような要件を満たす必要があるか、についてはしっかり確認**
- **詳細検討が必要な箇所としては、クロスアカウント連携をどう実装するかという点が挙げられる**

# 要件・設計マトリックス（必須）

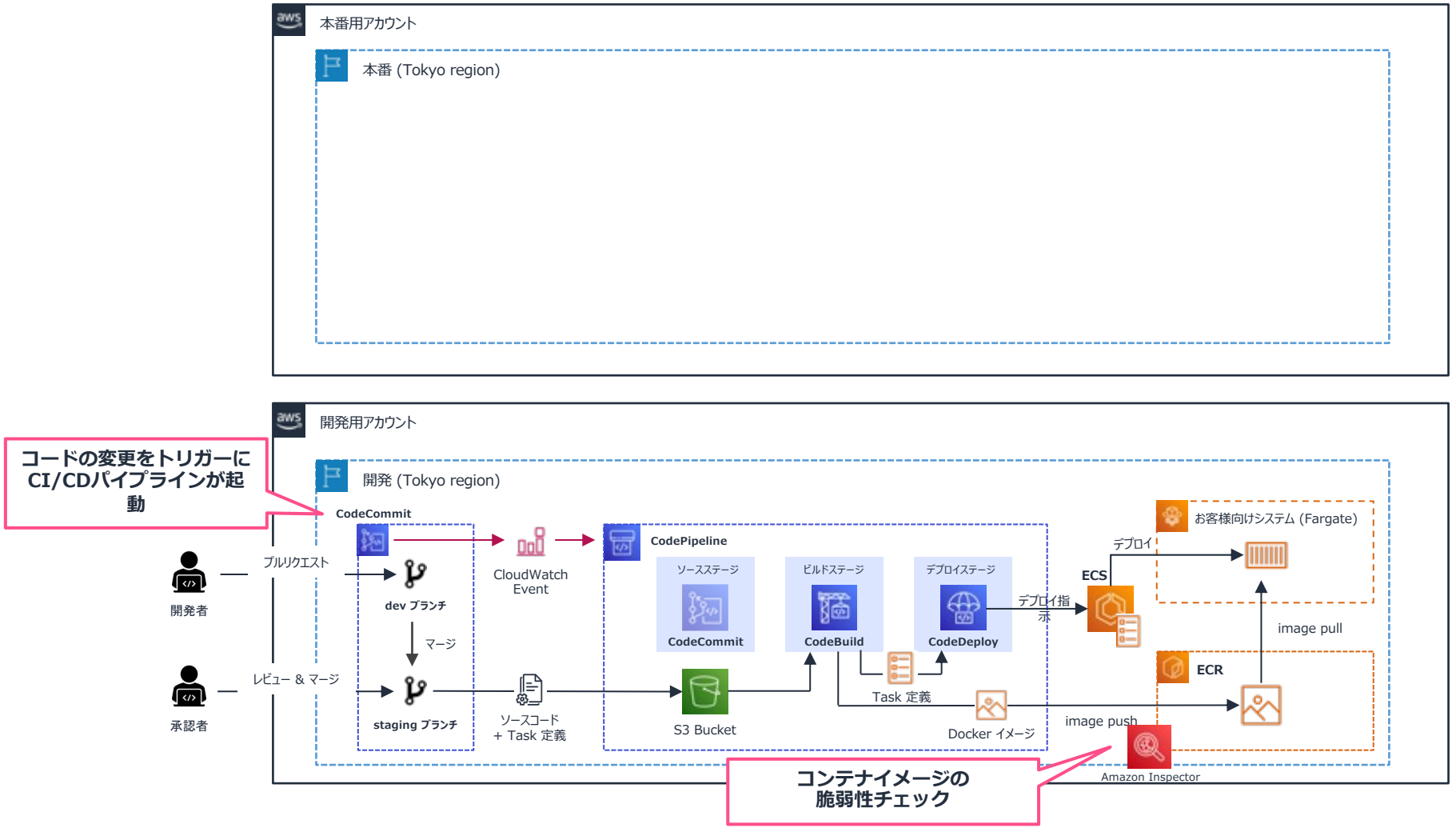
*1: 追加検討課題の取り組みは任意です。もし余力があれば「7.追加検討課題」から1-2つの課題を選び、検討・提案してください

| 区分 | 項目 | システム要件 （**自分で記入**） | 設計方針（**自分で記入**） |
|---|---|---|---|
| 追加検討<br>課題 ③ *1 | 開発環境と本番環境の<br>スムーズな連携 | 開発用と本番用の2つの AWS アカウントを利用する。<br>開発環境上でテスト・承認されたアップデートを本番環境へ迅速に展開できる仕組みを構築する。 | • **CodeBuild や CodeDeploy** 等のCode系サービスを用いてCI/CDパイプラインを構築する<br>• **テスト済み、かつ 承認済みのアップデートのみ**を本番環境へ展開するような仕組みを構築<br>• 本番環境でのアップデートの展開時には**責任者の承認プロセスを取り入れる** |
| 追加検討<br>課題 ④ *1 | 高度なセキュリティ対策 | CI/CDパイプラインの中で、本番環境に展開される前にアプリケーションの脆弱性や脅威を発見する。<br>AWS環境に対する攻撃や脅威を自動的にモニタリングする。<br>セキュリティ対策のための各サービス結果の一元化と評価を行う。 | • ビルド時に**Amazon Inspector** による、コンテナイメージの**脆弱性チェック**を行う<br>• GuardDutyによる脅威検出<br>• **Security Hub** によるセキュリティチェックの一元化と評価 |

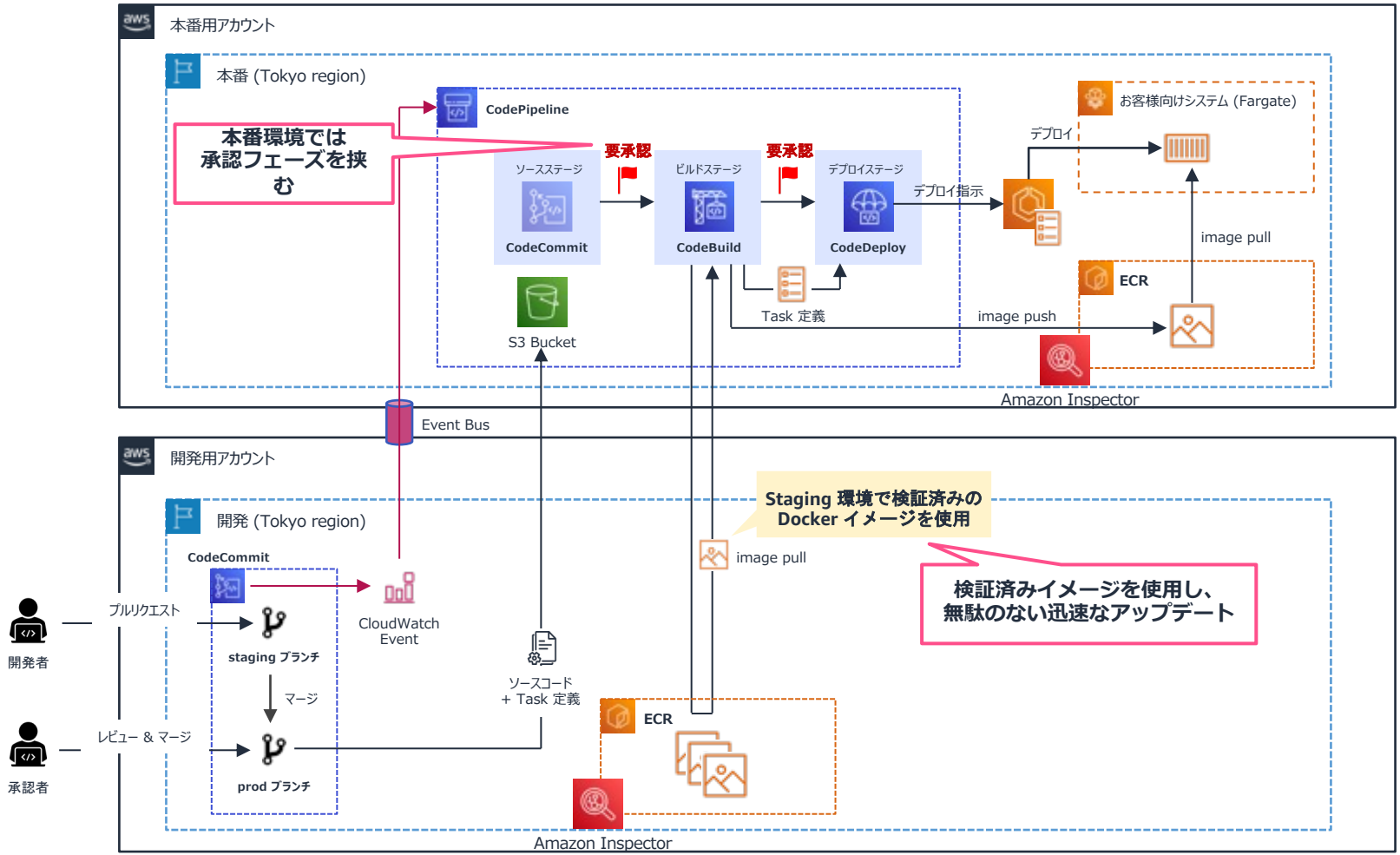


aws financial services

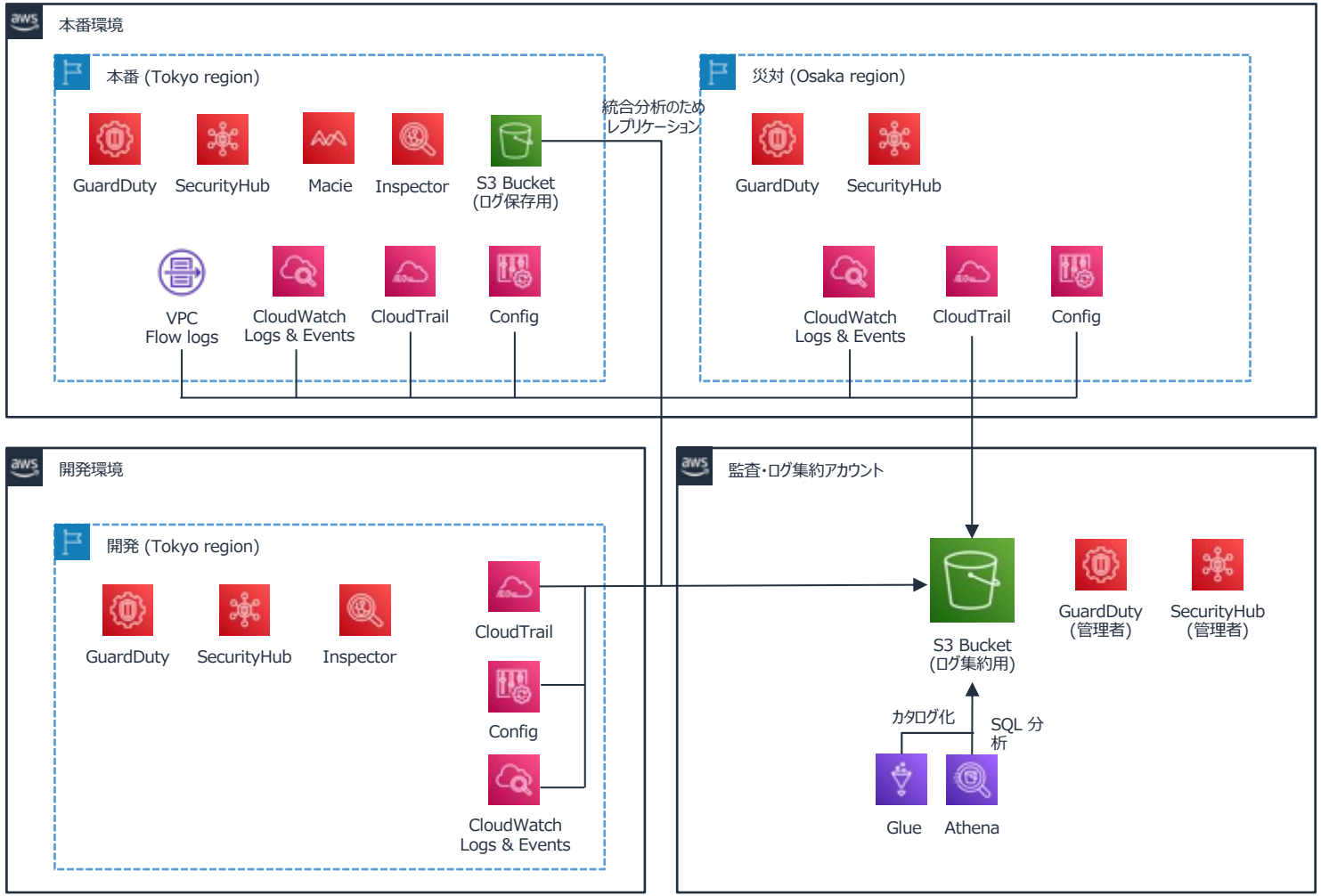
ステージング環境へのデプロイ



本番環境へのデプロイ


本番用アカウント
本番 (Tokyo region)
CodePipeline
本番環境では
承認フェーズを挟
む
ソースステージ
要承認
ビルドステージ
要承認
デプロイステージ
CodeCommit
CodeBuild
CodeDeploy
デプロイ指示
デプロイ
お客様向けシステム (Fargate)
image pull
ECR
Task 定義
image push
S3 Bucket
Amazon Inspector
Event Bus
開発用アカウント
開発 (Tokyo region)
CodeCommit
CloudWatch
Event
image pull
Staging 環境で検証済みの
Docker イメージを使用
検証済みイメージを使用し、
無駄のない迅速なアップデート
プルリクエスト
開発者
staging ブランチ
マージ
レビュー & マージ
承認者
prod ブランチ
ソースコード
+ Task 定義
ECR
Amazon Inspector

高度なセキュリティ監視例 (補足)



# 要件・設計マトリックス（必須）

*1: 追加検討課題の取り組みは任意です。もし余力があれば「7.追加検討課題」から1-2つの課題を選び、検討・提案してください

| 区分 | 項目 | システム要件 （**自分で記入**） | 設計方針（**自分で記入**） |
|---|---|---|---|
| 追加検討<br>課題 ⑤ *1 | CDNの利用 | DDoS攻撃に備えてCDNサービスを利用する<br>オリジンサーバへの直接アクセスは禁止する | • **ALBの前段にCloudFront**を設置する<br>• ALBのセキュリティグループで**CloudFrontのマネージドプレフィックスを許可**する |

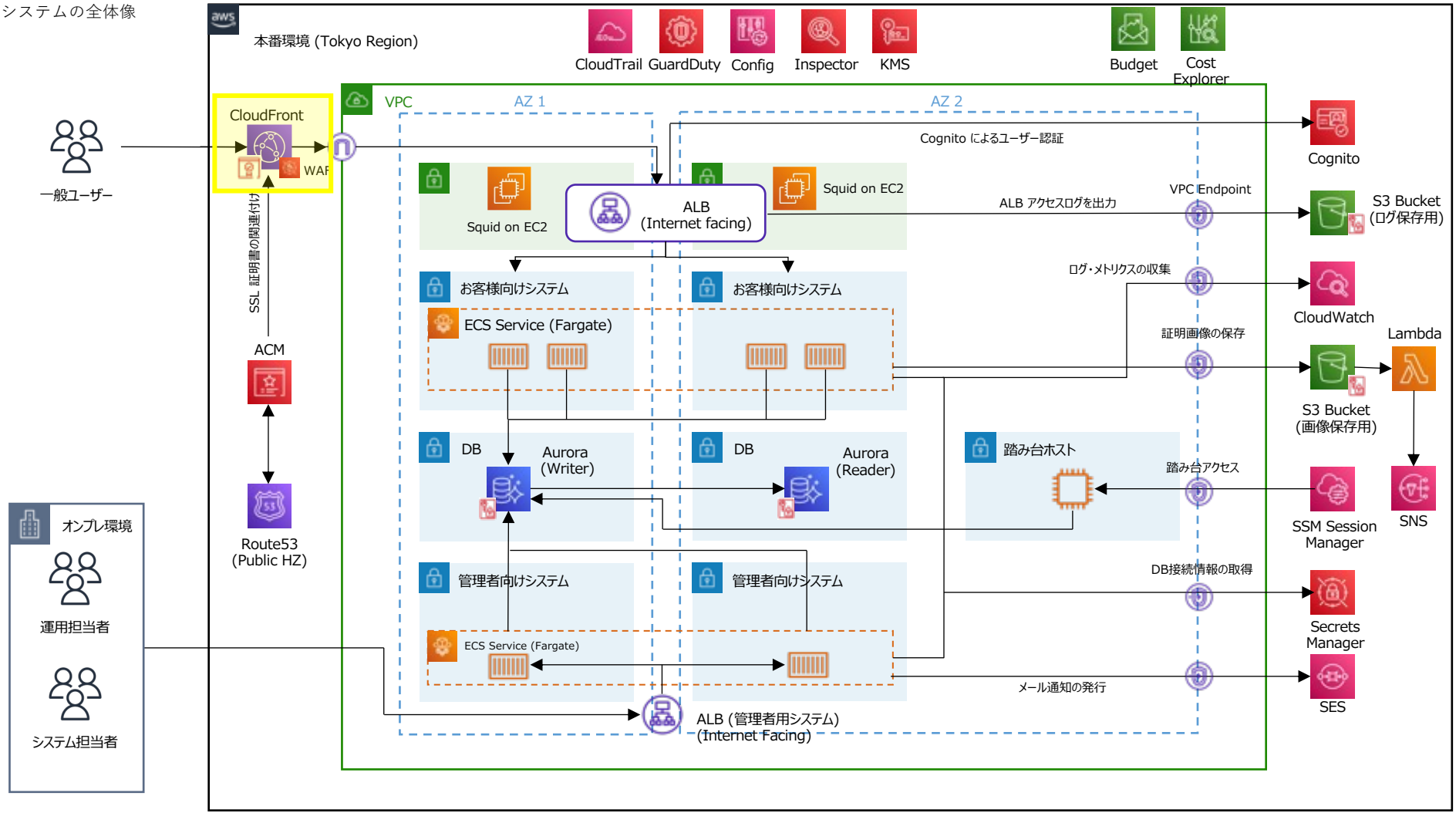
※補足

セキュリティ：AWSサービスはデフォルトでAWS Shieldによる防護により、SYN/UDP フラッド攻撃やリフレクション攻撃といった最も頻繁に発生する一般的なインフラストラクチャ (レイヤー 3 およびレイヤー 4) 攻撃を防御する機能が備わっています。
CDNの利用により、オリジンサーバの保護や同一IPアドレスからのリクエストのレート制限が可能です。



aws financial services

システムの全体像


本番環境 (Tokyo Region)
CloudTrail
GuardDuty
Config
Inspector
KMS
Budget
Cost Explorer
一般ユーザー
CloudFront
WAF
SSL 証明書の関連付け
ACM
Route53 (Public HZ)
VPC
AZ 1
AZ 2
Cognito によるユーザー認証
Cognito
Squid on EC2
ALB (Internet facing)
Squid on EC2
VPC Endpoint
ALB アクセスログを出力
S3 Bucket (ログ保存用)
ログ・メトリクスの収集
CloudWatch
お客様向けシステム
お客様向けシステム
ECS Service (Fargate)
証明画像の保存
Lambda
S3 Bucket (画像保存用)
DB
Aurora (Writer)
DB
Aurora (Reader)
踏み台ホスト
踏み台アクセス
SSM Session Manager
SNS
オンプレ環境
運用担当者
システム担当者
管理者向けシステム
管理者向けシステム
DB接続情報の取得
Secrets Manager
ECS Service (Fargate)
メール通知の発行
SES
ALB (管理者用システム) (Internet Facing)