Infrastructure Event Readiness

AWS ガイドラインとベストプラクティス

2017 年 *7* 月

注意

本文書は、情報提供の目的のみのために提供されるものです。本書の発行時点における AWS の現行製品と慣行を表したものであり、それらは予告なく変更されることがあります。お客様は本文書の情報および AWS 製品の使用について独自に評価する責任を負うものとします。これらの情報は、明示または黙示を問わずいかなる保証も伴うことなく、「現状のまま」提供されるものです。本書のいかなる内容も、AWS、その関係者、サプライヤー、またはライセンサーからの保証、表明、契約的責任、条件や確約を意味するものではありません。AWS がそのお客様に対して負う責任と義務は AWS の契約によって管理され、本書は、AWS とそのお客様の間のいかなる契約にも属さず、そのような契約を変更するものでもありません。

目次

要約

このホワイトペーパーでは、アマゾン ウェブ サービス (AWS) に生産ワークロードをデプロイしているお客様が、クラウドベースのアプリケーションを設計および準備し、製品のローンチや一時的なトラフィックスパイクのような想定されるスケーリングイベントにもスムーズかつダイナミックに対処できるようにするための、ガイドラインとベストプラクティスを紹介します。全般的なデザインプリンシパルを説明し、ベストプラクティスの具体例と、インフラストラクチャイベントのプランニングに関わる多様な概念領域にわたるガイダンスを提供します。そして、オペレーションのレディネスに関する考察およびプラクティスと、イベント後の活動についてお伝えします。

はじめに

Infrastructure Event Readiness とは、お客様のビジネスに影響を及ぼすと想定される重大なイベントに対していかに備えるかということです。企業のウェブサービスが、いかなる条件やトラフィックパターンの変動の下であれ真価を発揮することのできる信頼性や応答性、耐障害性を身に着けていることが、イベント時において決定的な役割を果たすことがあります。そうしたイベントの例としては、新しい地域への参入や、新製品や機能のローンチ、季節限定イベント、重大なビジネスアナウンスメントまたはマーケティングイベントを挙げることができます。

インフラストラクチャイベントに対する適切な備えをしておかないと、お客様のビジネス評価や、将来性、そして財務にネガティブな影響を被ることになります。インフラストラクチャイベントでの失敗としては、予期せざるサービスの不具合や、負荷によるパフォーマンスの低下、ネットワークレイテンシー、ストレージキャパシティー制限、API コールレートのようなシステム制限、利用可能な IP アドレス数の上限、不十分なモニタリングに起因するアプリケーションスタックのコンポーネントの挙動に対する理解の不足、サードパーティーのサービスまたはスケーリング以外のための導入コンポーネントに対する予期せざる依存を挙げることができます。これら以外にも、予想できないエラー条件がいくつもあります。

重要なイベント中の予期せざる失敗のリスクを最小限に抑えるために、企業は時間とリソースを投資して、計画し、準備し、従業員を訓練し、関連プロセスをデザインし文書化しなければなりません。特定のクラウド対応型アプリケーションまたはアプリケーション全般に関して、インフラストラクチャイベントのための計画を立てるのに必要な投資の量は、システムの複雑さや国際的な射程によって異なります。このホワイトペーパーが提示するデザインプリンシパルとベストプラクティスガイダンスは、企業におけるクラウドの存在感の射程や複雑さを問わず、利用できます。

お客様の企業は、アマゾン ウェブ サービス (AWS) によって、ダイナミックかつ柔軟に、その場に応じてコストを用意する方法で、想定されるスケーリングイベントに備えてインフラストラクチャをスケールアップすることができます。Amazon が提供する豊富かつ伸縮自在でプログラム制御可能な製品とサービスを用いることで、お客様の企業は、Amazon が自身の国際的なネットワークの運用に用いているものと同じ、高度な安全性と信頼性を備えた高速インフラス

トラクチャにアクセスすることができるだけでなく、目まぐるしく変動するビジネス要件に迅速に対応できるようになります。

このホワイトペーパーでは、お客様のインフラストラクチャイベントに対するプランニングと実行の指針となるベストプラクティスとデザインプリンシパルを紹介します。また、ビジネスのニーズに合わせてお客様のアプリケーションをスケールアップまたはスケールアウトするために AWS のサービスを活用する方法を示します。

Infrastructure Event Readiness プランニング

ここでは、プランドインフラストラクチャイベントを構成する要素と、イベント中に発生する典型的なアクティビティの種類を説明します。

プランドインフラストラクチャイベントとは何か？

プランドインフラストラクチャイベントとは、ビジネス主導の、想定された、スケジュールに基づくイベントウィンドウです。このイベント中は、高度な応答性とスケーリング、および耐障害性を備えたウェブサービスを維持することがビジネスにとって重要です。この要件は、マーケティングキャンペーンや、企業のビジネスラインに関係するニュースイベント、製品のローンチ、地理的な拡大など、企業のウェブベースのアプリケーションや基盤インフラストラクチャに追加のトラフィックが発生するアクティビティによって引き起こされます。

プランドインフラストラクチャイベント中に何が起きるか？

大半のプランドインフラストラクチャイベントにおける主要な関心事は、お客様のウェブインフラストラクチャのキャパシティーを増加して、多くのトラフィックデマンドに対応できるようにすることです。物理的なコンピューティングリソース、ストレージリソース、ネットワーキングリソースによる従来のオンプレミス環境プロビジョンでは、企業の IT 部門が理論的なピークの最大値を試算して、追加のキャパシティーを準備しておく必要があります。結果として、準備したキャパシティーが不足したり、ウェブサーバーの負担超過や応答

時間の遅さ、その他のランタイムエラーによって企業がビジネスで損害を被ったりすることがあります。

AWS クラウドであれば、インフラストラクチャがプログラム可能であり伸縮自在です。すなわち、リアルタイムなデマンドに対応して迅速に準備できるということです。更に、必要とあらば、ウェブサーバーのクラスターやプロビジョンドスループット、ストレージキャパシティー、利用可能なコンピューティングコア、多くのストリーミングシャードなどのような、自動化され、インテリジェントで、ダイナミックに拡大縮小するリソースにおけるシステムメトリクスに応じた設定をすることもできます。

さらに、AWS のサービスの多くは完全にマネージされています。サービスの例としては、ストレージやデータベース、分析、アプリケーション、そしてデプロイメントサービスがあります。AWS のお客様は、ハイトラフィックイベントのためにこれらのサービスを設定する際の複雑さを心配する必要はありません。AWS の完全にマネージされたサービスは、スケーラビリティと高度なアベイラビリティーを追求してデザインされています。

通例、プランドインフラストラクチャイベントのための準備において、AWS のお客様は、スケーラビリティと耐障害性の両方を考慮に入れて、アプリケーションアーキテクチャとオペレーションのレディネスを評価するためのシステムレビューを実施します。トラフィック計測が考慮され、普段のビジネスアクティビティのパフォーマンスと比較されます。そしてキャパシティーメトリクスと必要な追加キャパシティーが決定します。潜在的なボトルネックやサードパーティーの上り回線および下り回線への依存度合いも特定され対処がなされます。プランドイベントが地理的な拡大や新しいオーディエンスの導入を伴う場合は、地理的要因も考慮されます。AWS リージョンやアベイラビリティーゾーンの追加は、プランドイベントに先行して行われます。Auto Scaling やロードバランサー、ジオルーティング、ハイアベイラビリティー、フェイルオーバー測定のような AWS のダイナミックなシステム設定をお客様ご自身がレビューすることで、想定されるボリュームやトランザクションレートに正しく対応した設定になっているかを確認することもできます。AWS のリソース制限やコンテンツ配信ネットワーク (CDN) のオリジンサーバーのようなスタティックな設定も、必要に応じて考慮し修正することができます。

モニタリングと通知のメカニズムもレビューすることが可能であり、必要に応じて、発生しているイベントをリアルタイムに可視化し、プランドイベント終了後の事後分析の提供に向けて、補強することができます。

プランドイベント中に、サーバーダウンのようにトラブルシューティングまたはリアルタイムのサポートが必要となった場合、AWS のお客様はサポート事例を閲覧することもできます。AWS エンタープライズサポートにサブスクライブしたお客様は、迅速な対応が必要な場合にサポートエンジニアと素早く連絡をとり、極めて深刻な事態にも素早く対応できる、柔軟性の高いサービスを得られます。

イベントの後で、AWS のリソースは、イベントの必要に応じて、トラフィックレベルに合わせて適切な水準に自動でスケールダウンするか、またはスケールアップし続けるようにデザインされています。

デザインプリンシパル

プランドイベントに向けた準備は、最初にクラウドベースのアプリケーションスタックまたはワークロードを実装する際の優れたデザインから始まります。

個別のワークロード

プランドイベントのワークロードを、通常時および増加時の両方のトラフィックレベルで効率的にマネジメントするには、優れたデザインが不可欠です。最初から特定のビジネスアプリケーションや製品を中心に、リソースのグルーピングを個別にかつ機能的にデザインするようにしてください。このセクションでは、目標となるデザインについて多角的に説明します。

タグ付け

タグは、リソースをラベリングして体系的に管理するために用いられます。プランドインフラストラクチャイベント中にインフラストラクチャリソースを管理するために、不可欠な要素です。AWS では、タグはお客様が管理し、キー値のラベルはロードバランサーや Amazon Elastic Compute Cloud (EC2) インスタンスのようなリソースを個別に管理するために適用されます。AWS のリソースに添付された明確に定義されたタグを参照することで、お客様のインフラストラクチャ全体の中で、どのリソースがプランドイベントのワークロードを構成しているかを容易に特定することができます。また、この情報を用いることで、準備を進めるための分析を行うことができます。タグは、コスト配分を決めるために用いることもできます。

例えば、タグは、EC2 インスタンスや、Amazon Machine Image (AMI) のイメージ、ロードバランサー、セキュリティグループ、Amazon Relational Database Service (RDS) のリソース、Amazon Virtual Private Cloud (VPC) のリソース、Amazon Route 53 のヘルスチェック、Amazon Simple Storage Service (S3) のバケットの体系的管理のために用いることができます。

効率的なタグ付け戦略の詳細については、AWS のタグ付け戦略をご覧ください。[1]

タグの作成と管理、リソースグループへのタグ付けの例については、AWS のリソースグループとタグ付けをご覧ください。[2]

疎結合

クラウド向けのアーキテクチャを作るときは、アプリケーションスタックの各コンポーネントが、可能な限り個別にオペレーションするようにデザインすることをお勧めします。これにより、クラウドベースのワークロードに、弾力性とスケーラビリティという利点を持たせることができるのです。

クラウドベースのアプリケーションスタックのコンポーネント間の相互依存を緩和させるためには、インプットとアウトプット用の明確に定義されたインターフェイスを備えたブラックボックスとして各コンポーネントをデザインします (例えば、RESTful API)。コンポーネントがアプリケーションではなく、アプリケーションを構成するサービスである場合、これはマイクロサービスアーキテクチャと呼ばれます。アプリケーションのコンポーネント間のコミュニケーションと協調動作のために、コンポーネント間でメッセージをパスさせる AWS メッセージキューのような、イベントによる通知メカニズムを活用することができます。図 1 をご覧ください。

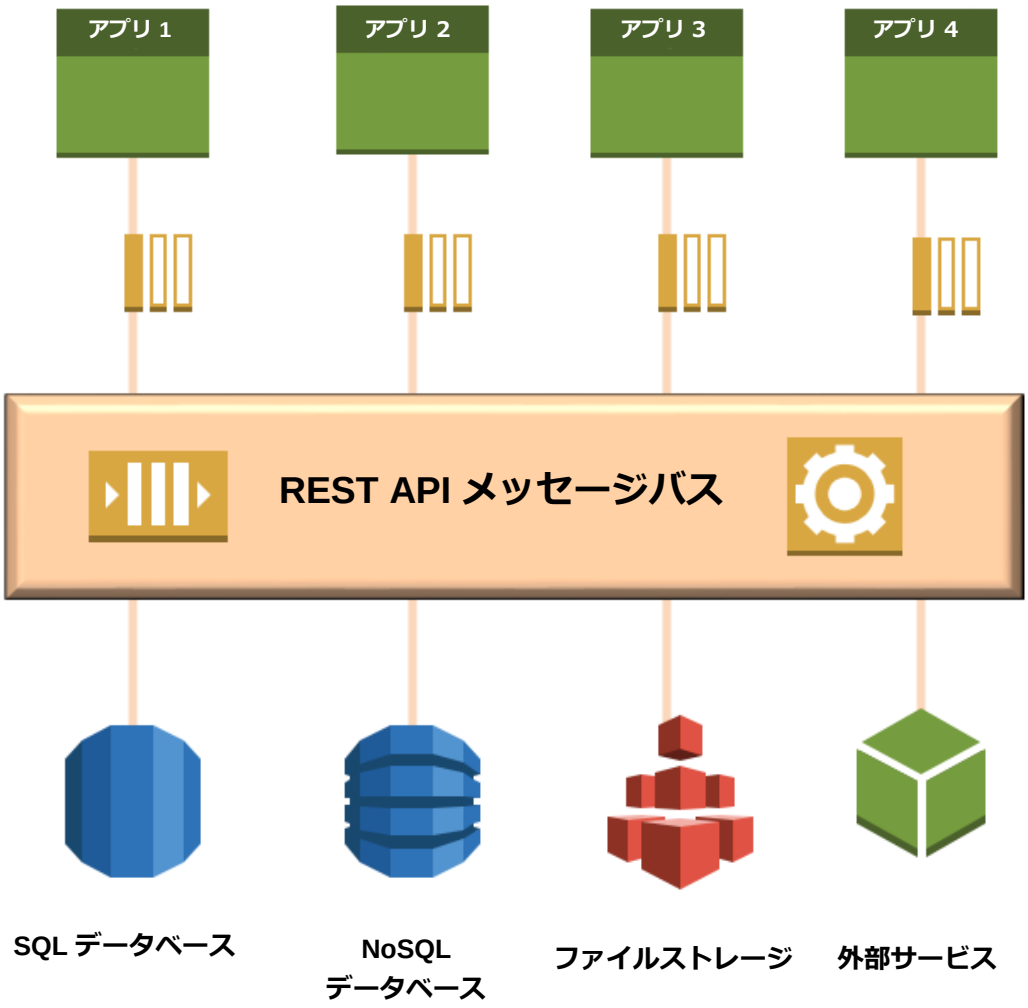

図 1. RESTful インターフェイスとメッセージキューを用いた疎結合

このようなメカニズムを用いることで、1 つのコンポーネントにおける変更や不具合を他のコンポーネントにカスケードする可能性を抑えることができます。例えば、複層アプリケーションスタックのサーバーからの応答がなくなった場合、疎結合のアプリケーションは、応答のなくなった層を迂回したり、デグレードモードの代替トランザクションに切り替えることができます。

メッセージキューを媒体として用いる疎結合のアプリケーションコンポーネントでは、非同時性の統合向けにデザインすることがより簡単です。アプリケーションコンポーネントが、直接的なポイントツーポイント形式のコミュニケーションではなく、仲介となる一貫性のあるメッセージングレイヤー (例えば、Amazon Simple Queue Service (SQS) キュー、または Amazon Kinesis Streams のようなストリーミングデータメカニズム) を採用しているため、下りのコンポーネントがキューを処理しているあいだにも 1 つのコンポーネントにおける

急激なアクティビティ増加に耐えることができます。コンポーネントに不具合がある場合、コンポーネントが復旧するまで、メッセージはキューまたはストリーミングで繰り返されます。

AWS が提供するメッセージキューと通知サービスの詳細については、Amazon Simple Queue Service をご覧ください。[3]

サーバーではなくサービス

マネージドサービスとサービスエンドポイントは、セキュリティやアクセス、バックアップ、復旧、パッチマネジメント、チェンジコントロール、モニタリング、レポーティングの設定、また従来の様々なシステムマネジメントの詳細な管理に対する不安から、お客様を解放します。これらのクラウドリソースは、複層的なアベイラビリティーゾーン (ときにはマルチプルリージョン) 設定を用いて、ハイアベイラビリティーと弾力性のためにあらかじめ準備することができます。これらは、ときに休止時間を設けることなくスケールアップまたはスケールダウンすることが可能であり、AWS マネジメントコンソールや API / CLI コールを縦横無尽に活用して、設定することができます。

マネージドサービスとサービスエンドポイントは、リレーショナルデータベースや NoSQL データベースシステム、データ保存、イベント通知、オブジェクトおよびファイルのストレージ、リアルタイムストリーミング、ビッグデータ分析、機械学習、検索、トランスコーディングなどのような様々な機能によって、お客様のアプリケーションスタックを強化します。エンドポイントは AWS のサービスへのエントリポイントとなる URL です。例えば、https://dynamodb.us-west-2.amazonaws.com が、Amazon DynamoDB サービスのエントリポイントになります。

マネージドサービスとサービスエンドポイントを使うことで、すぐに使うことのできるリソースをお客様自身のデザインソリューションの一部として活用して、プランドインフラストラクチャイベント中の、ボリュームや射程、トランザクションレートの増加に対処することができるようになります。マネージドサービスと同じ機能を果たすようにお客様自身のサーバーを準備し、管理する必要はありません。

AWS のサービスエンドポイントの詳細については、AWS のリージョンとエンドポイントをご覧ください。[4]また、エンドポイントのあるマネージドサービスの例については、Amazon EMR、[5] Amazon RDS、[6]および Amazon ECS[7]をご覧ください。

サーバーレスアーキテクチャ

プランドインフラストラクチャイベント中にダイナミックに変動する処理負荷に効率よく対応するための他の戦略としては、AWS Lambda の活用があります。Lambda はイベントによるサーバーレスのコンピューティングプラットフォームです。これは、ダイナミックに呼び出されるサービスであり、イベントに応じて Python や、Node.js、または Java コードを実行し、コードによって特定されたコンピューティングリソースを自動で管理します。Lambda では、Amazon EC2 のコンピューティングリソースをあらかじめ準備する必要がありません。Amazon Simple Notification Service (Amazon SNS) は Lambda の関数をトリガーするように設定することができます。Amazon SNS の詳細については、Amazon Push Notification Service をご覧ください。[8]

Lambda のサーバーレス関数は、データベースオペレーションや、データトランスフォーメーション、オブジェクトまたはファイルの取得、更には外部的なイベントや内部的なシステム負荷メトリクスに応じてオペレーションをスケーリングするなどの、他の AWS のサービスへのアクセスや呼び出しのためのコードを実行することができます。AWS Lambda は、それ自体が新しい通知やイベントを生成することもできるだけでなく、他の Lambda 関数を起動することさえできます。

AWS Lambda は、プランドインフラストラクチャイベント中にオペレーションをスケーリングする精細なコントロールを行うための手段を提供します。例えば、Lambda を用いて Auto Scaling オペレーションの機能を拡張し、同じくスケーリングを必要とするサードパーティーシステムの通知のようなアクションを実行したり、新しいインスタンスが準備されたときに追加のネットワークインターフェイスを増加させることができます。スケーリングオペレーションをカスタマイズするために Lambda を用いる方法の例については、Auto Scaling ライフサイクルフックと一緒に AWS Lambda を使用する方法 [9]をご覧ください。

AWS Lambda の詳細については、 AWS Lambda とは何か?[10]をご覧ください。

自動化

Auto Scaling

インフラストラクチャイベントのプランニングで非常に重要な要素は、Auto Scaling です。あらかじめ定義された条件に従ってアプリケーションのキャパシティーを自動的にスケールアップまたはスケールダウンできるようになると、

プランドインフラストラクチャイベント中にトラフィックパターンやボリュームが変動したときに、アプリケーションのアベイラビリティーを維持するのが容易になります。

AWS は、EC2 インスタンスやデータベースキャパシティー、コンテナなどの多くのリソースで、Auto Scaling 機能を提供します。

Auto Scaling は、インスタンスのグルーピングをスケールするのに用いることができます。例えば、クラウドベースのアプリケーションを構成するサーバー群を、特定の基準に従って自動でスケールします。Auto Scaling は、1 つのインスタンスが正常でなくなったときでさえも、インスタンスを一定の数に保つことができます。自動スケーリングとインスタンスの数の維持は、Auto Scaling サービスのコアとなる機能です。

Auto Scaling は、グループ内のインスタンスに定期的なヘルスチェックを実行することで、お客様が指定した数のインスタンスを維持します。あるインスタンスが正常でなくなった場合、グループは、正常でなくなったインスタンスを削除し、他のインスタンスを起動し、代替させます。

Auto Scaling ポリシーは、変動し続ける条件に合わせてサーバーグループ内で実行される EC2 インスタンスの数を、自動で増減させるために用いることができます。スケーリングポリシーが有効化されているときには、トラフィックに既知の予想可能な満ち引きがあれば、必要に応じて、Auto Scaling グループがスケジュールをダイナミックに変化させ、または代替しつつ、望ましいグループのキャパシティーを調節し、インスタンスを起動または削除します。

再起動とリカバリ

あらゆるプランドインフラストラクチャイベントにおいて重要なデザイン要素は、危険のあるインスタンスやサーバーに対処して、中断することなくそれらを修復または再起動することができるようにするための手順とオートメーションを用意しておくことです。

EC2 インスタンスは、基盤ハードウェアのシステムステータスチェックが失敗したときに、自動で修復するように設定することができます。インスタンスは (必要に応じて新しいハードウェアで) リブートされますが、インスタンス ID や、IP アドレス、Elastic IP アドレス、Amazon Elastic Block Store (EBS) のボリュームアタッチメント、およびその他の設定詳細を保持します。EC2 インス

タンスの自動リカバリの詳細については、Amazon EC2 の自動修復をご覧ください。[11]

設定管理とオーケストレーション

頑強で、信頼性と応答性のあるプランドインフラストラクチャイベントの戦略を実現するために不可欠なのは、個別のリソースステート管理やアプリケーションスタックのデプロイに用いられている、設定管理とオーケストレーションのツールを組み合わせることです。

通例、設定管理ツールは、サーバーインスタンスやロードバランサー、Auto Scaling、個別のアプリケーションのデプロイ、アプリケーションのヘルスモニタリングの準備と設定を扱うために用いられます。また、データベースやストレージボリューム、キャッシングレイヤーのような追加のサービスを統合する機能も提供します。

オーケストレーションツールは、設定管理上の抽象化レイヤーの 1 つであり、様々なリソースの関係性を特定する手段を提供し、お客様が、リソースごとの依存を心配することなく、マルチプルなリソースを単一のクラウドアプリケーションインフラストラクチャとして準備し管理することができるように助けます。

これらのツールが、個別のリソースとその関係性をコードとして定義し記述しているため、コードをバージョン管理することが可能であり、過去のバージョンに戻したり、テストや開発目的で新しいコードを試したりすることができます。インフラストラクチャイベントに最適化されたオーケストレーションと設定を定義したり、イベント次第ではスタンダードの設定に戻したりすることもできます。

アマゾン ウェブ サービスでは、コードデプロイメントおよびオーケストレーションのためのハードウェアを実現するためのツールとして、以下のものを推奨しています。

- **AWS Config with Config Rules**、または AWS Config パートナーは、詳細で視覚化された、検索可能なインベントリを AWS のリソースや設定履歴、およびリソース設定コンプライアンスに提供します。
- **AWS CloudFormation**、またはサードパーティーの AWS リソースオーケストレーションツールは、AWS リソースのプロビジョニングやアップデート、および削除を管理します。

- **AWS OpsWorks、Elastic Beanstalk**、またはサードパーティーのサーバー設定管理ツールは、オペレーティングシステム (OS) とアプリケーションの設定の変更を管理します。

ハードウェアをコードとして管理するための方法の詳細については、インフラストラクチャ設定管理をご覧ください。[12]

多様性 / 弾力性

単一障害点とボトルネックを除去

インフラストラクチャイベントのためのプランニングをするときには、単一障害点 (SPOF) やパフォーマンスボトルネックを想定して、アプリケーションスタックを分析する必要があります。例えば、不具合を起こしたときにアプリケーションの全体や大部分を停止させてしまうような、サーバーのインスタンスや、データボリューム、データベース、NAT ゲートウェイ、ロードバランサーはありませんか?

または、クラウドベースのアプリケーションでトラフィックやトランザクションボリュームがスケールアップした際に、データフローパスと一緒にデータのボリュームが増大したときの CPU プロセスサイクルや、ネットワークの帯域幅のような物理的制限や制約に直面するインフラストラクチャの要素はありませんか?

こうしたリスクは、いったん特定したら、様々な方法で緩和することができます。

不具合を想定したデザイン

先に述べたように、RESTful インターフェイスで疎結合とメッセージキューを使用することが、個別のリソースの不具合やトラフィックまたはトランザクションボリュームの変動に対する弾力性を達成するための優れた戦略です。異なるディメンションで弾力性を達成するためのデザインは、アプリケーションコンポーネントを可能な限りステートレスに設定することです。

ステートレスアプリケーションには先行のトランザクションに関する知識は必要ありませんし、他のアプリケーションコンポーネントへの依存もゆるやかです。ステートレスアプリケーションはセッション情報を保存しません。ステートレスアプリケーションは、プールやクラスター内のインスタンスによってリ

クエストを扱うことができるため、プールやクラスターの要素と同じように水平方向にスケールします。必要に応じて、Auto Scaling やヘルスチェック基準を使って、より多くのリソースを追加するだけで、変動し続けるコンピューティングやキャパシティー、スループット要件をプログラムによって処理することができます。ステートレスとしてデザインされたアプリケーションは、EC2 インスタンスの代わりに Lambda 関数を用いることで、サーバーレスアーキテクチャにリファクターすることが可能です。Lambda 関数は、もともとダイナミックなスケーリング能力を備えています。

ウェブサーバーのようなアプリケーションリソースがトランザクションに関するステートデータを持つことが避けられない場合、お客様のアプリケーションのステートフルな部分がサーバー自体から干渉されないようにデザインする必要があります。例えば、HTTP クッキーや同種のステートデータを、DynamoDB のようなデータベースや S3 バケット、EBS ボリュームに保存することができます。

複雑な多段階のワークフローを用いていて、各段階の現在のステートをトラッキングする必要がある場合、Amazon Simple Workflow サービス (SWF) を用いることで、実行履歴を一元的に保存し、ワークロードをステートレスにすることができます。

他に弾力性を得る方策は、分散処理の採用です。1 つのコンピューティングリソースが要求を満たすことができないときに、適宜、大量のデータを処理することが必要になるユースケースでは、タスクやデータをいくつもの小さな部分に分割して、コンピューティングリソースのクラスターで並行して実行することができるようにワークロードをデザインすることができます。分散処理はステートレスな処理であり、分割されたデータやタスクが処理されている個別のノードに不具合が発生する可能性があります。この場合には、分散処理スケジューリングエンジンによって、不具合の起きたタスクを分散処理クラスターの別のノードで自動的に再開させることができます。

AWS は、Amazon EMR や、Amazon Athena、Amazon Machine Learning のような、様々な分散データ処理エンジンを提供しています。これらはそれぞれ、エンドポイントを提供しているマネージドサービスであり、パッチングやメンテナンス、スケーリング、フェイルオーバーなどのあらゆる複雑さからお客様を解放します。

ストリーミングデータをリアルタイムで処理するために、Amazon Kinesis Streams は、データを複数のシャードに分割して、Lambda 関数や EC2 インスタンスのような、複数のデータコンシューマによって処理できるようにします。

このような種類のワークロードの詳細については、AWS のビッグデータ分析オプションをご覧ください。[13]

マルチゾーンとマルチリージョン

AWS のサービスは、世界各地の複数の場所で運営されています。これらの場所は、リージョンとアベイラビリティーゾーンから構成されます。リージョンとは、個別の地理的なエリアのことです。各リージョンには、互いに孤立した場所が複数あります。これがアベイラビリティーゾーンです。AWS は、インスタンスのようなリソースとデータを複数の場所に配置する方法を提供します。

お客様のアプリケーションを、複数のアベイラビリティーゾーンとリージョンに分散させることをお勧めします。複数のアベイラビリティーゾーンとリージョンにおけるリソースのディストリビューションとレプリケーションを組み合わせる際には、ロードバランシングとフェイルオーバーのメカニズムを用いて、お客様のアプリケーションスタックが、不具合時に自動的にデータフローやトラフィックを別の場所にリルートするようにアプリをデザインする必要があります。

ロードバランシング

Elastic Load Balancing サービス (ELB) によって、アプリケーションサーバー群をロードバランサーに取り付けて、複数のアベイラビリティーゾーンに分散させることもできます。ロードバランサーの背後にある特定のアベイラビリティーゾーンにおける EC2 インスタンスがヘルスチェックに失敗すると、ロードバランサーはこれらのノードへのトラフィックの転送を中止します。Auto Scaling と組み合わせると、手作業なしで自動的に、多くの正常なノードを他のアベイラビリティーゾーンに再調整します。

Amazon Route 53 や、レイテンシーベースの DNS ルーティングアルゴリスムを用いることで、複数のリージョンにまたがるロードバランシングも可能になります。詳細については、レイテンシーベースのルーティングをご覧ください。[14]

ロードシェディング戦略

クラウドベースのインフラストラクチャにおけるロードシェディングのコンセプトは、トラフィックをどこかにリダイレクトまたはプロキシングして、主要なシステムにかかる負荷を軽減することです。ロードシェディング戦略は、トリアージの実践になることがあります。つまり、特定のトラフィックストリーミングを脱落させたり、お客様のアプリケーションの機能を削減したりして、処理負荷を軽減し、やってくるリクエスト群の処理だけはできるようにするという選択です。

ロードシェディングのために活用できるテクニックはいくつもあります。レイテンシーベースの DNS ルーティングも 1 つの方法です。他にはキャッシングを使うこともできます。キャッシングは、Amazon ElastiCache のようなインメモリキャッシングレイヤーを用いて、アプリケーションの近くで行うことができます。または、Amazon CloudFront のようなグローバルなコンテンツディストリビューションネットワークを用いて、ユーザーのエッジにより近いキャッシングレイヤーを使うこともできます。

ElastiCache や、CloudFront の詳細については、開始方法と ElastiCache[15] Amazon CloudFront CDN をご覧ください。[16]

コストの最適化

リザーブド vs スポット vs オンデマンド

システムメトリクスとその他のパフォーマンスやヘルスチェックの基準に基づいて、クラウド内にリソースをダイナミックにプロビジョンする能力は、クラウド内にリソースをプロビジョンするためのコストを管理する能力と密接に結びついています。Auto Scaling によって、リソースの利用を、実際の処理とストレージのニーズにぴったりと一致させ、無駄な費用と活用されていないリソースを最小限に抑えることができます。

また、クラウド内でのコスト管理として、オンデマンドインスタンス、リザーブドインスタンス (RI)、スポットインスタンスの中から選ぶこともできます。DynamoDB 向けにはリザベーションキャパシティーも用意されています。

オンデマンドインスタンスでは、使用している EC2 インスタンスに対してのみ料金が発生します。オンデマンドインスタンスは、長期間の契約なしに、時間単位で、コンピューティング性能に対して料金が発生します。

Amazon EC2 リザーブドインスタンスは、オンデマンドインスタンスの料金に比べて際立って安く (最大 75% 割引)、特定のアベイラビリティーゾーンで使用した場合は、キャパシティーリザベーションを提供します。ただし、アベイラビリティーリザベーションと料金の安さ以外に、リザーブドインスタンスとオンデマンドインスタンスの間には違いがありません。

スポットインスタンスとは、Amazon EC2 の余剰のコンピューティング性能に対しお客様が入札できるシステムです。スポットインスタンスは、たいていオンデマンドの料金よりも安く利用できるため、お客様のクラウドベースのアプリケーションを運用するコストを大幅に削減します。

クラウド用にデザインする場合、一部のユースケースは、他のインスタンスよりもスポットインスタンスの使用に向いています。例えば、利用価格がお客様の入札価格を上回った場合、スポットインスタンスが終了することがあります。そのため、お客様は、比較的ステートレスで水平方向にスケールしたアプリケーションスタックのためにのみ、スポットインスタンスを運用することを考える必要があります。ステートフルなアプリケーションまたは費用のかかる処理負荷には、リザーブドインスタンスやオンデマンドインスタンスのほうがより適しているでしょう。キャパシティー制限をかけられないミッションクリティカルなアプリケーションには、リザーブドインスタンスが最適です。

詳細については、リザーブドインスタンス [17]およびスポットインスタンス [18]をご覧ください。

イベント管理プロセス

インフラストラクチャイベントの計画は、アプリケーション開発者や管理者、ビジネスステークホルダーを巻き込んだグループ活動です。インフラストラクチャイベントに先行する数週間のあいだ、ウェブサービスのキーとなるインフラストラクチャコンポーネントを所有し運用する、重要な技術スタッフを巻き込んで、定期的かつ頻繁に会議を行う必要があります。

インフラストラクチャイベントのスケジュール

インフラストラクチャイベントの計画は、イベントの数週間前に開始しましょう。プランドイベントのライフサイクルにおける典型的なタイムラインを、図 2 に示しました。

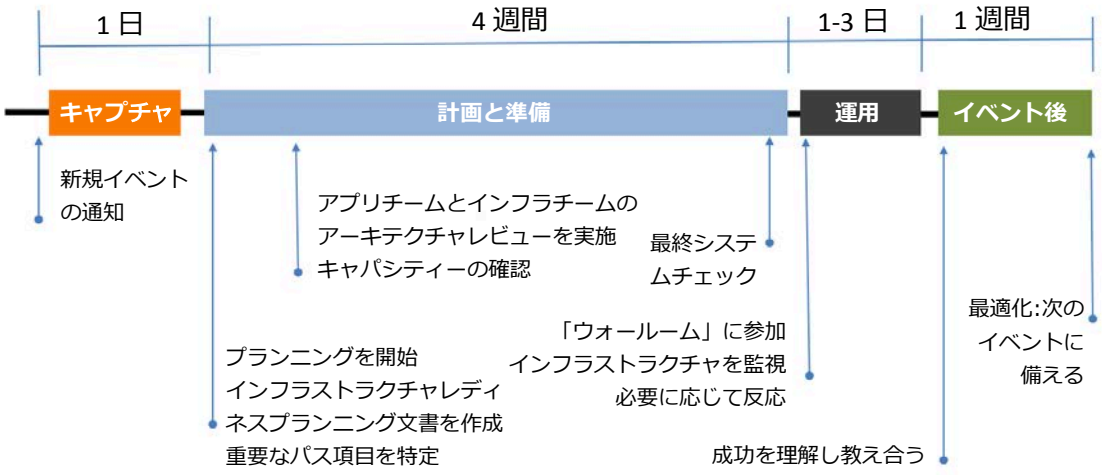

図 2. インフラストラクチャイベントの典型的なタイムライン

計画と準備

スケジュール

インフラストラクチャイベントに先行する数週間のアクティビティのスケジュールとしては、以下のものをお勧めします。

第 1 週:

- インフラストラクチャイベントのための計画とエンジニアリングを主導するチームを任命します。
- イベントのパラメーター (スケール、期間、時間、地理的範囲、影響を受けるワークロード) と成功の基準を理解するために、ステークホルダー間で会議を実施します。
- 下流または上流のパートナーとベンダーにも参加してもらいます。

第 2-3 週:

- アーキテクチャをレビューして、必要に応じて調整します。
- オペレーションをレビューして、必要に応じて調整します。
- このペーパーと補足の参考資料で説明されているベストプラクティスを参照します。
- リスクを特定し、ミティゲーションプランを作成します。
- プランドイベントの運用指示書を作成します。

第 4 週:

- 想定される負荷に基づいて、スケーリングが必要なクラウドベンダーサービスをすべてレビューします。
- サービスの上限をチェックして、必要に応じて上限を増加させます。
- モニタリングダッシュボードと、定義されたスレショルドによるアラートを設けます。

アーキテクチャレビュー

インフラストラクチャイベントを準備するにあたって不可欠な要素は、トラフィックの急増を被ると想定されるアプリケーションスタックのアーキテクチャレビューです。レビューの目的は、アプリケーションのスケーラビリティや信頼性に潜在的なリスクのある部分を確認し特定すること、およびイベントに先んじて最適化する機会を判断することです。

AWS は、エンタープライズサポートのお客様に、5 本の柱を軸にしてお客様のアプリケーションスタックをレビューするためのフレームワークを提供します。5 本の柱とはすなわち、セキュリティ、信頼性、パフォーマンス効率、コスト最適化、オペレーションの卓越性です。下記をご覧ください。

表 1: 優れたアプリケーションアーキテクトの柱

柱の名前	柱の定義	関連する関心領域
セキュリティ	リスクアセスメントとミティゲーション戦略を通じてビジネスバリューをお届けする一方で、情報やシステム、アセットを保護する能力のことです。	アイデンティティ管理、暗号化、モニタリング、ログ記録、キー管理、ハードウェア専有インスタンス、コンプライアンス、ガバナンス
信頼性	インフラストラクチャまたはサービスの不具合から復旧し、需要に合わせてダイナミックにコンピューティングリソースを取得し、設定ミスや一時的なネットワーク問題のような混乱を緩和する能力です。	サービス上限、複数のアベイラビリティーゾーンとリージョン、スケーラビリティ、ヘルスチェック / モニタリング、バックアップ / DR、ネットワーキング、自己修復オートメーション
パフォーマンス効率	コンピューティングリソースを効率的に使ってシステム要件を満たし、需要が変化し技術が発展するのに合わせて効率性を維持します。	正しい AWS のサービス、リソースの利用、ストレージアーキテクチャ、キャッシング、レイテンシー要件
コストの最適化	不要なコストや最適でないリソースを削除する能力です。	スポット / リザーブドインスタンス、環境チューニング、サービス選択、ボリュームチューニング、アカウント管理、一括請求 (コンソリデーティッドビリング)、リソース撤去
オペレーションの卓越性	ビジネスバリューをお届けし、継続的にプロセスと手順のサポートを改善するために、システムを運用しモニタリングする能力です。	運用指示書、プレイブック、CI / CD、Game Days、コードとしてのインフラストラクチャ、RCA

このホワイトペーパーの添付書類にある、アーキテクチャレビューで検討する項目の詳細なチェックリストは、AWS ベースのアプリケーションスタックのレビューに役立ちます。

オペレーションレビュー

アプリケーションのデザインコンポーネントに焦点を当てるアーキテクチャレビューに加えて、クラウドオペレーションや管理プラクティスをレビューして、お客様がご自身のクラウドワークロードを管理する能力を評価する必要があり

ます。レビューの目的は、オペレーションのギャップと問題を特定し、それらを、イベント前に最小にするためのアクションを実行することです。

AWS は、エンタープライズサポートのお客様に、インフラストラクチャイベントの準備に役立つクラウドオペレーションレビューのツールを提供します。レビューでは以下の領域の評価に焦点を当てます。

- 準備–組織構造とプロセス、技術を正しく組み合わせましょう。お客様のアプリケーションスタックを管理するスタッフに、明確な役割と責任を設けましょう。イベントに取り組む前にプロセスを定義しましょう。可能なら手順を自動化しましょう。
- モニタリング-アプリケーションのパフォーマンスを測定するための効率的なモニタリング。モニタリングは、異常が問題となる前に発見し、不利なイベントの影響を最小に抑えるチャンスをもたらします。
- オペレーション-オペレーション活動は、エスカレーションを必要とする予期せざるオペレーションイベントに対処している間であっても、可能ならばオートメーションを活用して、タイムリーに信頼できる方法で実行する必要があります。
- 最適化-集めたメトリクスやオペレーションのトレンド、教訓を用いて事後分析を実施することで、将来起り得るイベントでの改善の機会を獲得し、報告します。最適化と準備の組み合わせが、オペレーションの問題に対処し、問題の再発を防ぐためのフィードバックループを作り出します。

AWS のサービス上限を理解

プランドインフラストラクチャイベント中には、アプリケーションまたはワークロードをスケーリングする際にも、クラウドのプロバイダによって設けられたサービス上限を超過しないようにする必要があります。

クラウドサービスのプロバイダは、通例、お客様が使用することのできるリソースに上限を設定します。上限はアカウントおよびリージョンごとに設けられるのが普通です。上限のあるリソースの例としては、インスタンスやボリューム、ストリーミング、サーバーレスな呼び出し、スナップショット、VPC の数、セキュリティルールなどがあります。上限は、リソースの不正使用を試みる暴走コードや悪意のあるアクターを想定した安全対策であり、ビリングリスクを最小化するためのコントロールです。

お客様がクラウド内のフットプリントを拡張するのに従って、一部のサービス上限は自動的に増大しますが、サービスの大半は、お客様ご自身でサポートケースを開いて、上限の増大をリクエストする必要があります。サポートケースを通じて上限を増大させることのできるサービスもありますが、変更できないサービスもあります。

AWS は、将来を見越してすべてのサービス上限を管理するための上限チェックダッシュボードを伴った Trusted Advisor により、エンタープライズサポートおよびビジネスサポートをお客様に提供します。

様々な AWS のサービスに設けられた上限と、それらをチェックする方法の詳細については、 AWS のサービス上限 [19]Trusted Advisor をご覧ください。[20]

パターンの理解

ベースライン
インフラストラクチャイベントの開始前に、キーメトリクス用の[正常な状態への復旧]の値を記録しておきましょう。イベントの完了や終了後にアプリケーションやサービスを安全に通常レベルに戻すタイミングを決定する役に立ちます。例えば、ロードバランサーを通じた通常のトラフィック率が秒間 2,500 リクエストだと特定することができれば、イベント後に手順を下げる安全なタイミングを判断するのが容易になります。

データフローと依存
アプリケーションの様々なコンポーネントを通じてどのようにデータがフローしているかを理解することは、潜在的なボトルネックと依存を特定する助けになります。データを生み出すアプリケーションスタック内のティアーとコンポーネントがスケールアップした場合に、データフロー内のデータを消費するアプリケーションのティアーとコンポーネントが正しいサイズになっており、自動でスケールするように設定されていますか？コンポーネントに不具合が起きたときに、コンポーネントが修復されるまでデータがキューアップできるようになっていますか？下流または上流のデータプロバイダやコンシューマが、お客様のイベントに応じてスケールできるようになっていますか？

比率
インフラストラクチャイベントの準備におけるレビューで他に考慮しなければいけないことは、アプリケーションスタックの様々なコンポーネントに必要なスケーリングの比率です。比率は必ずしも 1 対 1 のものではありません。例えば、ロードバランサーでの秒間トランザクションが 10 倍に増加すると、フロ

ントフェーシングアプリケーションで行われる処理のために、ストレージキャパシティーやストリーミングシャードの数、データベースの読み取りと書き込みオペレーションの数が 20 倍必要になります。

コミュニケーションプラン

イベントに先んじて、コミュニケーションプランを作成する必要があります。内部のステークホルダーとサポートグループのリストを作り、様々なシナリオを想定してイベントの各段階で連絡を取るべき相手を特定しましょう。各段階というのは、例えば、イベントの開始、イベントの進行中、イベントの終了時、事後分析、緊急連絡、トラブルシューティング中の連絡などです。

以下は、連絡するべき人物およびグループの例です。

- ステークホルダー
- オペレーションマネージャー
- 開発者
- サポートチーム
- クラウドサービスのプロバイダチーム
- ネットワークオペレーションセンター (NOC) のチーム

内部の連絡相手のリストを作成するときには、アプリケーションの継続的なライブ配信に関係する外部のステークホルダーの連絡リストも作成する必要があります。これらのステークホルダーとは、例えば、スタックのキーコンポーネントをサポートするパートナーおよびベンダー、外部サービスやデータフィード、認証サービスを提供する下流および上流のベンダーなどです。

こうした外部向け連絡リストには、以下を含むようにしましょう。

- インフラストラクチャホスティングベンダー
- テレコミュニケーションベンダー
- ライブデータストリーミングパートナー
- PR マーケティングの担当者
- 宣伝広告パートナー
- サービスエンジニアリングに関わるテクニカルコンサルタント

各プロバイダには、以下のような情報の提供を依頼しましょう。

- イベント中の連絡のライブポイント
- 重要なサポート連絡とエスカレーションプロセス
- 名前、電話番号、メールアドレス
- リアルタイムでの技術的な連絡が可能となる認証

エンタープライズサポートにサブスクライブした AWS のお客様には、担当の AWS サポートスタッフがイベントのサポートに準備するように調整および検証することのできる、テクニカルアカウントマネージャー (TAM) を配属します。TAM は、イベント中に連絡することができ、ウォールームに参加し、必要に応じてサポートエスカレーションの実行を助けします。

NOC の準備

イベントの前に、製作中のウェブサービスで重要な各コンポーネントをイベントが起きたときにモニタリングするための、ライブメトリクスダッシュボードを作成する指示をオペレーションチームや開発者チームに出しましょう。ダッシュボードは、1 分ごとに、またはイベント中に適切で効率的な間隔で、自動的にメトリクスを更新するのが理想的です。

以下のコンポーネントのモニタリングを考慮しましょう。

- 各サーバーのリソース利用 (CPU、ディスク、メモリ使用量)
- ウェブサービスの応答時間
- ウェブトラフィックメトリクス (ユーザー、ページ表示、セッション)
- ビジターの地域ごとのウェブトラフィック (グローバルカスタマーセグメント)
- データベースサーバー利用
- コンバージョンレートとフォールアウト率のような、マーケティングフローコンバージョンファンネル
- アプリケーションエラーのログ
- カナリアモニタリング

Amazon CloudWatch は、CloudWatch のカスタムダッシュボードを用いて、AWS のリソースから大半のメトリクスを収集する手段を提供します。さらに、CloudWatch は、AWS が自動的にメトリクスを提供していない CloudWatch にカスタムメトリクスをインポートする機能も提供します。AWS のモニタリングツールと機能の詳細については、このペーパーのモニタリングのセクションをご覧ください。

運用指示書の準備

インフラストラクチャイベント用の運用指示書を作成しましょう。運用指示書とは、お客様のオペレーターがイベント中に実施する手順やオペレーションを記した、オペレーションマニュアルです。イベントの運用指示書は、ルーチンオペレーションや例外処理に使われている既存の運用指示書を参考にして作ることができます。通例、運用指示書には、システムを開始し、停止し、監査し、デバッグするための手順が記されています。予期せざるイベントや偶発的な事故に対処するための手順も含むとよいでしょう。

運用指示書には以下のセクションを含むようにしましょう。

- **イベントの詳細**: イベントや、成功の基準、メディアカバレッジ、イベントの日時、顧客側および AWS などの主要なステークホルダーからの連絡の詳細を、簡潔に描写します。
- **AWS のサービスのリスト**: イベント中に用いられる AWS のサービスを列挙します。また、これらのサービスや、影響を受けるリージョン、アカウント ID が被ると想定される負荷も示します。
- **アーキテクチャおよびアプリケーションのレビュー**: ドキュメント負荷テストの結果、インフラストラクチャおよびアプリケーションデザインのストレスポイント、ワークロードに対する弾力性の測定、単一障害点、潜在的なボトルネック。
- **オペレーションレビュー**: ハイライトモニタリングセットアップ、ヘルス基準、通知メカニズム、およびサービス復旧手順。
- **準備度合いのチェックリスト**: 例えば、サービス上限のチェックや、ロードバランサーのようなアプリケーションスタックコンポーネントのプレウォーミング、ストリームシャードや DynamoDB 分割、S3 分割のようなリソースの事前準備などを検討することです。詳細については、このホワイトペーパーの添付書類にある、アーキテクチャレビューの詳細なチェックリストをご覧ください。

モニター

プランのモニタリング

データベース、アプリケーション、およびオペレーションシステムのモニタリングはイベントで成功を収めるために必要です。包括的なモニタリングシステムをセットアップして、インフラストラクチャイベント中の深刻なインシデントを効率的に発見し、迅速な措置を講じることができるようにしておきましょう。高水準に達すると、効率的なモニタリング戦略により、ビジネスの重要性に応じて適切な度合でモニタリングツールをアプリケーションに実装することができます。効率的なインシデント管理戦略を実現するには、AWS およびカスタマーモニタリングデータの両方を、イベントとインシデントの管理ツールおよびプロセスと組み合わせましょう。お客様の AWS のソリューションセグメント全体からモニタリングデータを集合的に集めるモニタリングプランを実施することで、複雑な不具合が発生したときにデバッグするのが非常に容易になります。

モニタリングプランでは、以下のような問題を扱うようにしましょう。

- イベントのためにモニタリングツールとダッシュボードがセットアップするべきは何か？
- モニタリングの目的と許可されたスレショルドは何か？どのようなイベントがアクションをトリガーするか？
- どのようなリソースや、リソースのメトリクスをモニタリングして、適宜調節しなければならないか？
- 誰がモニタリングタスクを実行するか？モニタリングアラートはどのように設定するべきか？誰がアラートを受信するか？
- 一般的不具合または想定される不具合のために、どのような修正プランをセットアップするべきか？予期せざるイベントにはどのように対処するか？
- 不具合発生時のエスカレーションをどのようなプロセスで実施するか？

以下に記した AWS のモニタリングツールは、こうした戦略の一部として活用することができます。

- **Amazon CloudWatch**: AWS のダッシュボードメトリクスや、モニタリング、アラート、および自動プロビジョニングのための、独創的なソリューションです。

- **Amazon CloudWatch カスタムメトリクス**: オペレーティングシステムや、アプリケーション、ビジネスメトリクス向けに用いられます。Amazon CloudWatch の API は実質的にあらゆる種類のカスタムメトリクスにも対応可能です。
- **Amazon EC2 インスタンスヘルス**: ステータスチェックを確認したり、自動リブーティングやインスタンスの再開といったお客様のインスタンス用のイベントを、インスタンスの状態に基づいて、スケジュールするために用いられます。
- **Amazon SNS**: イベントによる通知のセットアップ、オペレーション、および送信のために用いられます。
- **AWS X-Ray**: システムコンポーネントのデータフローを分析することで、分散されたアプリケーションとマイクロサービスアーキテクチャをデバッグしたり、分析したりするのに役立ちます。
- **Amazon Elasticsearch Service**: 一元的なログ収集とリアルタイムのログ分析のために用いられます。迅速かつヒューリスティックに問題を発見します。
- **サードパーティーのツール:** リアルタイム分析や、フルスタックモニタリングおよび視覚化のために用いられます。
- **スタンダードなオペレーティングシステムのモニタリングツール** OS レベルでのモニタリングのために用いられます。

AWS のモニタリングツールの詳細については、自動モニタリングと手動モニタリングをご覧ください。[21]また、Amazon CloudWatch ダッシュボードの使い方 [22]と Custom Metrics の公開もご覧ください。[23]

通知

インフラストラクチャイベントのためのデザインで重要なオペレーション要素は、アラームや通知がお客様のモニタリングソリューションと組み合わさるように設定することです。これらのアラームの通知を AWS Lambda のようなサービスと一緒に用いると、アラートに基づくアクショントリガーを設定することができます。オペレーションイベントに対する自動応答は、最大限の応答性をもって、ミティゲーションやロールバック、復旧を可能にするためのキーとなる要素です。

また、ツールは、一元的にワークロードをモニタリングし、キーとなるオペレーション指標に関係するログとメトリクスの中でも利用可能なものに基づいて、適切なアラートと通知を作成するように設定しましょう。例えば、異常がコン

トロールできなくなったときや、サービスまたはコンポーネントに不具合が発生したときのためのアラートと通知です。ローパフォーマンスのスレショルドや不具合が発生したときには、システムが、そのような通知とアラートに応じて、自動的に自己修復またはスケーリングするようにアーキテクトしておくことが理想的です。

既に述べたように、AWS は、予期せざるオペレーションイベントに応じて適切なアラートと通知を送信し、自動的に対応できるようにするためのサービス (Amazon SQS および Amazon SNS) を提供しています。

オペレーションレディネス (イベント当日)

プランの実行

イベント当日には、インフラストラクチャに関わるコアチームが、リアルタイムでダッシュボードをモニタリングするライブカンファレンスコールを受けることができるようにしておく必要があります。運用指示書の真価を発揮させ、利用しましょう。明確に定義したコミュニケーションプランをすべてのサポートスタッフやステークホルダーに伝え、コンティンジェンシープランをしっかりと準備しておきましょう。

ウォールーム

イベント中は、下記の関係者とのあいだにライブカンファレンスブリッジを開いておきましょう:

- 主に責任を負っているアプリケーションチームおよびオペレーションチーム
- オペレーションチームのリーダー
- テクニカルデリバリーに直接関係している、外部パートナーのテクニカルリソース
- ビジネスステークホルダー

イベント中、基本的には、このカンファレンスブリッジにおける会話は最小限にしましょう。不都合なオペレーションイベントが発生した場合、イベントに対応することのできる重要な関係者は、あらかじめブリッジに出席し、相談と行動の準備ができているようにしましょう。

リーダーへの報告

イベント中は、重要なリーダーとなるステークホルダーに、頻繁にメールを送信しましょう。メールでの更新情報として以下の項目をお勧めします。

- ステータスサマリー: 緑色 (順調)、黄色 (問題発生)、赤色 (重大な問題発生)
- キーメトリクスの更新情報
- 発生した問題、復旧プランのステータス、復旧のための ETA
- ウォールームカンファレンスブリッジの電話番号 (参加希望者がいる場合)

イベント終了時には、同様の形式を用いて、最終的なサマリーをメールで送信するとよいでしょう。

コンティンジェンシープラン

イベントのための準備プロセスの各段階に関して、テスト環境で検証されたロールバックプランに相当するプランを用意しておく必要があります。

ロールバックプランの内容として以下の問題点を参考にしてください。

- イベント中に起り得る最悪のシナリオは何か？
- どのような種類のイベントが広報活動にネガティブな影響を及ぼすか？
- どのサードパーティーコンポーネントまたはサービスがイベント中に不具合を起こし得るか？
- バッドシナリオの発生の指標としてモニタリングするべきメトリクスはどれか？
- 起り得ると想定される各シナリオ用のロールバックプランはどのようなものか？
- 各ロールバックプロセスにはどのくらいの時間がかかるか？許容できるリカバリーポイント目標 (RPO) とリカバリータイム目標 (RTO) は何か？(これらのコンセプトの詳細については、ディザスターリカバリーのために AWS を用いる方法 [24]をご覧ください。)

以下の種類のロールバックについて検討してください。

- **ブルー / グリーンディプロイメント**: 新製品のアプリや環境をロールアウトする場合、過去の製品バージョンをオンラインで保持し、いつでも迅速にロールバックすることができるようにしましょう。
- **ウォームパイロット**: 必要に応じて速やかにスケールアップすることのできる別のリージョンで最小限の環境をローンチしましょう。主要リージョンで不具合が発生した場合、速やかにバックアップリージョンでスケールアップし、トラフィックをスイッチさせましょう。
- **メンテナンスモードエラーページ**: ウェブサービスの各レイヤーにおけるエラーページのファシリティーやトリガーをチェックしましょう。必要に応じて、これらのエラーページにできるだけ具体的なメッセージを挿入できるように準備しておきましょう。

発生し得る各不具合のシナリオに応じたロールバックプランをテストし、文書化しておきましょう。

イベント後のアクティビティ

事後分析

お客様は通例、普段のオペレーションに戻すことを気にかけているため、事後分析は見落とされやすい項目です。それでも、インフラストラクチャイベントの管理ライフサイクル一環として、事後分析を行うことをお勧めいたします。事後分析により、関係のある各チームと協力し、オペレーション手順や実行の詳細、フェイルオーバーおよびリカバリの手順などのような、更なる最適化を必要とする部分を特定することができます。事後分析は、イベント中にアプリケーションスタックが中断したときに効果を発揮します。イベントの事後分析は、ルートコース分析 (RCA) ドキュメントを作成する必要がある場合に、文書化するのにも役立ちます。

ウィンドダウンプロセス

インフラストラクチャイベントの終了直後に、ウィンドダウンプロセスを開始しましょう。このとき、関連のあるアプリケーションとサービスのモニタリングを継続し、トラフィックが通常のプロダクションレベルに戻っているかどうか確認することをお勧めします。準備段階で作成されたあらゆるヘルスダッシュボードを用いて、トラフィックとトランザクション率が通常化しているかどうか検証しましょう。イベントのウィンドダウン期間は、直線的な時もあれば、不均等または漸次的にボリュームが減少する場合もあります。何らかのトラフ

ィックパターンが見られるでしょう。例えば、トラフィックの急増からの復旧には直線的なウィンドダウン手順が必要とされることが一般的ですが、新しい地理的なリージョンへのアプリケーションのディプロイメントや拡張の場合は、長期的な影響力を有するため、新しいトラフィックパターンを注意深くモニタリングし、永続的なアプリケーションスタックに追加でモニタリングする部分を付け加える必要があります。

イベント終了後のどこかの段階で、イベント管理オペレーションを安全に終了することのできるタイミングを判断しなければなりません。過去に文書化された、キーメトリクスの「通常」値が、イベントの完了または終了を宣言するタイミングを判断する参考になるでしょう。ウィンドダウンアクティビティは 2 つの部門に分けて、各部門に異なるタイムラインを設けることをお勧めします。1 つ目の部門は、イベントのオペレーション管理に注力します。例えば、内部および外部のステークホルダーとパートナーに向けて連絡し、サービスリミットをリセットすることです。2 つ目の部門は、ウィンドダウンのテクニカルな面に注力します。例えば、スケールダウン手順や、環境のヘルスの検証、アーキテクチャの変更を元に戻すか、それともコミットするかの判断基準などです。

各部門のタイムラインは、イベントの性質やキーメトリクス、およびお客様の都合によって決定します。以下の表に、各部門で扱われるべき一般的なタスクの概要を示します。イベント管理を終了する適切なタイミングを判断する際の参考にしてください。

表 2: オペレーションのウィンドダウンタスク

タスク	説明
コミュニケーション	内部および外部のステークホルダーにイベントの終了を通知。終了タイミングの連絡は、イベントの終了の定義と連動させる必要があります。[正常な状態への復旧] メトリクスを用いて、コミュニケーションを終了する適切なタイミングを判断しましょう。もしくは、コミュニケーションを段階的に終了させることもできます。例えば、ウォールームブリッジを閉鎖する一方で、イベント後の不具合に備えて、イベントのエスカレーション手順をそのまま残しておくこともできます。
サービス上限 / コスト抑制	イベントの後でもサービス上限を上昇させたままにしておくのは魅力的な選択肢ですが、サービス上限がセーフティーネットの役割をも果たしていることを肝に銘じておいてください。サービス上限は、サービスの過剰な使用や、アカウント乗っ取り、オートメーションの設定ミスを防ぐことで、お客様ご自身とお客様のコストを保護しています。

タスク	説明
報告と分析	データ収集とイベントメトリクスの対照調査は、パターンやトレンド、問題領域、成功した手順、アドホックな手順、イベントのタイムライン、そして成功の基準を満たすことができたか否かの判断を示す分析叙述と一緒に、コミュニケーションプランによって特定された内部の関係者向けに作成し、配布しましょう。イベントのサポートに要したオペレーションの費用を示すために、詳細なコスト分析も作成する必要があります。
最適化タスク	企業組織は、自身のオペレーションを改善するにつれて、時と共に進化します。オペレーションを最適化するためには、メトリクスやオペレーションのトレンド、イベントから得られた教訓をコンスタントに集め、改善の機会を見つけ出すことが不可欠です。最適化と準備を組み合わせることで、オペレーションの問題に対処し、問題の再発を防ぐためのフィードバックループを作り出すことができます。

表 3: テクニカルなウィンドダウンタスク

タスク	説明
サービス上限 / コスト抑制	イベントの後でもサービス上限を上昇させたままにしておくのは魅力的な選択肢ですが、サービス上限がセーフティーネットの役に立っていることも肝に銘じておいてください。サービス上限は、乗っ取られたアカウントからの悪意によるアクティビティストリーミングまたはオートメーションの設定ミスに由来する、サービスの過剰な使用を防ぐことで、お客様のオペレーションとオペレーションコストを保護しています。
スケールダウン手順	準備段階でスケールアップさせたリソースを元に戻しましょう。対象となるリソースはお客様のアーキテクチャごとに異なりますが、以下に一般的な例を示します: EC2 / RDS インスタンスのサイズ Auto Scaling 設定 リザーブドキャパシティー プロビジョンド IOPS
環境の正常さの検証	ベースラインメトリクスと比較検討し、プロダクションの正常さをレビューして、イベント後とスケールダウン後の手順が完了しているかということと、影響を受けたシステムが通常の挙動を報告しているかということを検証しましょう。
アーキテクチャの変更のデ	イベントに備えるための変更は、イベントの性質とオペレーションメトリクスの様子次第では、維持する方が良い場合もあります。例えば、新しい地理的なリージョンへ拡張

タスク	説明
ポジション	すると、当該リージョンにおける恒久的なリソースの増加が必要になる場合があります。また、特定のサービス上限や、DB の分割数または、ボリューム内の PIOPS のストリーミングにおけるシャードの数のような設定パラメーターの上昇は、継続するべきパフォーマンスチューニングと見なすことができる場合もあります。

最適化

事後分析を行って、確認されたオペレーションおよびアーキテクチャの問題を特定し、改善の機会を得ることは、インフラストラクチャイベント管理にとって、最も重要であると言えます。インフラストラクチャイベントが一度限りで終わることはほとんどありません。インフラストラクチャイベントは定期的に発生したり、アプリケーションの新リリースと重なったりすることがあります。また、企業が新しい市場や地域に参入する際の成長の一環となることもあります。そのため、どんなインフラストラクチャイベントも、次のイベントをいっそう効率的に処理するために、観察し、改善し、準備する機会になるのです。

まとめ

AWS は、どんなスケールのワークロードでもサポートすることのできる、弾力性のあるプログラム可能な製品とサービスという形で、素材を提供します。AWS のインフラストラクチャイベントのガイドラインとベストプラクティスを、私たちが提供する利便性の高いサービス群と組み合わせて用いることで、お客様は迅速な対応とグローバルな射程を確保できます。それと同時に、メジャーなビジネスイベントのためのデザインと準備を実施し、スケーリングデマンドをスムーズかつダイナミックに満たすことができるようになるのです。

寄稿者

本書の執筆に当たり、次の人物および組織が寄稿しました。

- Presley Acuna, AWS エンタープライズサポートマネージャー
- Kurt Gray, AWS グローバルソリューションアーキテクト
- Michael Bozek, AWS シニアテクニカルアカウントマネージャー

- Rovan Omar, AWS テクニカルアカウントマネージャー
- Will Badr, AWS テクニカルアカウントマネージャー
- Eric Blankenship, AWS シニアテクニカルアカウントマネージャー
- Greg Bur, AWS テクニカルアカウントマネージャー
- Bill Hesse, AWS シニアテクニカルアカウントマネージャー
- Hasan Khan, AWS シニアテクニカルアカウントマネージャー
- Varun Bakshi, AWS シニアテクニカルアカウントマネージャー

その他の資料

オペレーションとアーキテクチャのベストプラクティスについては、AWS 用のオペレーションチェックリストもご覧ください。[25]クラウドベースのアプリケーションデリバリースタックを評価するための構造的アプローチに対する読者のレビューをお勧めしています。AWS Well Architected Framework[26]AWS は、AWS のテクニカルアカウントマネージャーおよびサポートエンジニアと直接かかわりながら、デザイン、プランニング、イベント当日のオペレーションに対処することを希望するお客様のためのプレミアムサポートとして、インフラストラクチャイベント管理 (IEM) を提供しています。AWS IEM プレミアムサポートの詳細については、インフラストラクチャイベント管理をご覧ください。[27]

添付資料

アーキテクチャレビューの詳細なチェックリスト

はい-いいえ-N/A	セキュリティ
□-□-□	AWS のセキュリティベストプラクティスにしたがって、3 ヶ月ごとに、AWS Identity and Access Management (IAM) のアクセスキーおよびユーザーパスワードと、自社のアプリケーションに関係するリソースの認証情報を変更している。すべてのアカウントにパスワードポリシーを適用し、ハードウェア認証またはバーチャルマルチファク

はい-いいえ-N/A	セキュリティ
	ター認証 (MFA) デバイスを使用している。
□-□-□	内部のセキュリティプロセスを設けており、また、IAM を活用して AWS API にアクセスする権限が役割ごとに固有かつ最小限のものになるように管理している。
□-□-□	あらゆるカスタマイズされた Amazon マシンイメージ (AMI) から、埋め込まれた公開 / 専用インスタンスキーペアを含む極秘情報または要注意情報を取り除き、SSH 認証を受けたすべてのキーファイルをレビューした。
□-□-□	AMI に認証情報を埋め込む代わりに、IAM ロールを EC2 インスタンスとして活用している。
□-□-□	IAM を管理する役割を作り、他に機能している役割には IAM のアクションを制限することで、IAM の管理権限を通常のユーザーの権限から切り離している。
□-□-□	Windows または Linux のインスタンスで最新のセキュリティパッチを EC2 インスタンスに適用している。Amazon EC2 セキュリティグループルールや、VPC ネットワークアクセスコントロールリスト、OS ハードニング、ホストベースのファイアウォール、侵入の発見および予防、ソフトウェア設定のモニタリング、ホストインベントリなどのオペレーティングシステムのアクセスコントロールを使用している。
□-□-□	企業の AWS および企業環境とのネットワーク接続が、確実に暗号化プロトコルでの転送を使用して行われるようにしている。
□-□-□	異常なアクセスパターンや、環境に対する悪意のある攻撃の特定と分析のために、一元的なログと監査管理のソリューションを採用している。
□-□-□	イベントおよびインシデントの管理、対比、時宜に応じた報告を安全に行うことができる。
□-□-□	自社のセキュリティグループ内で、AWS のリソースに対する無制限のアクセス権限は存在しない。
□-□-□	フロントエンドコネクション (クライアントからロードバランサーへのコネクション) のために、安全なプロトコル (HTTPS または SSL) や、最新のセキュリティポリシー、暗号プロトコルを使用している。リクエストはクライアントとロードバランサーの間で暗号化されており、よりいっそうの安全が確保されている。
□-□-□	自社のドメインでメールを送信することが認証されたサーバーを特定するための、対応するセンダーポリシーフレームワーク (SPF) の値を含む TXT リソースレコードセットを

はい-いいえ-N/A	セキュリティ
	有するように、Amazon Route 53 の MX リソースレコードセットを設定している。

はい-いいえ-N/A	信頼性
□-□-□	あらかじめ定義されたスケーリングプランに基づく自動的な水平方向のスケーリングを確保するために、Auto Scaling グループにデプロイされた EC2 インスタンス群にアプリケーションをデプロイしている。詳細。
□-□-□	Auto Scaling グループが基礎的な EC2 インスタンスの正常さに基づいて動作するように、Auto Scaling グループ内で、Elastic Load Balancing のヘルスチェックを使用している。(Auto Scaling グループのロードバランサーを使用している場合のみ該当)
□-□-□	複数のアベイラビリティーゾーンにわたって自社アプリケーションの重要コンポーネントをデプロイしており、ゾーン間で適切にデータを複製している。Elastic Load Balancing や、Amazon Route 53、またはサードパーティーツールを用いて、これらのコンポーネントにおける不具合が、アプリケーションのアベイラビリティーに及ぼす影響をテストしている。
□-□-□	データベースレイヤーにおいて、Amazon RDS インスタンスを複数のアベイラビリティーゾーンにデプロイし、異なるアベイラビリティーゾーンでスタンバイインスタンスを同時複製することで、データベースのアベイラビリティーを補強している。
□-□-□	停電やパフォーマンスの低下に備えて、自動または手動でのフェイルオーバーするための手順が定められている。
□-□-□	DNS を自社のサービスにマッピングするための CNAME レコードがある。A レコードを使用していない。
□-□-□	Amazon Route 53 のレコードセットに、短い有効生存期間 (TTL) の値を設定している。これは、トラフィックをリルート中の DNS リゾルバが最新の DNS レコードをリクエストしたときに遅延を避けるためのものです。(例えば、DNS フェイルオーバーがお客様のエンドポイントに不具合を発見し、それに対処しようとしているときに機能します。)
□-□-□	停電または AWS のエンドポイントにおけるデバイスの予定メンテナンスに備えて余剰を確保するために、設定済みの VPN トンネルを少なくとも 2 つ用意している。

はい-いいえ-N/A	信頼性
□-□-□	デバイスが利用できなくなったときに備えて余剰を確保するために、AWS Direct Connect を使用し、常に、設定済みの Direct Connect を 2 つ用意している。特定の場所が利用できなくなった場合に備えて余剰を確保するために、接続は、異なる場所の Direct Connect で準備されている。 複数の Direct Connect と場所で、複数の設定済み仮想インターフェイスを確保するために、自社の仮想プライベートゲートウェイに対する接続も設定している。
□-□-□	Windows のインスタンスを使用し、最新の PV ドライバを適用していることを確認している。PV ドライバは、ドライバのパフォーマンスを最大化し、ランタイムイシューとセキュリティリスクを最小化する役に立ちます。EC2Config エージェントが Windows のインスタンスで最新のバージョンを使用していることを確認している。
□-□-□	Amazon Elastic Block Store (EBS) ボリュームのスナップショットを取って、不具合時に備えたポイントインタイムリカバリを確保している。
□-□-□	必要に応じて、オペレーティングシステムおよびアプリケーションやデータベースごとに別の Amazon EBS ボリュームを使用している。
□-□-□	Linux インスタンスに、最新のカーネルやソフトウェア、ドライバパッチを適用している。

はい-いいえ-N/A	パフォーマンス効率
□-□-□	AWS がホストするアプリケーションのコンポーネントを、実際に使用する前に、パフォーマンスを含めてテストしている。EC2 インスタンスのサイズや、IOPS の数、RDS DB インスタンスのサイズなどが適切かどうか確認するために、負荷テストを実施している。
□-□-□	サービス上限に対する使用率のチェックレポートを実施し、AWS のサービスの現在の使用率が、サービス上限の 80% 以下であることを確認している。詳細
□-□-□	アプリケーション (Amazon CloudFront) のキャッシングや、ユーザーに最も近いエッジロケーションへのコンテンツの配信とコンテンツの自動ディストリビューションを最適化するために、コンテンツ配信 / ディストリビューションネットワーク (CDN) を用いている。
□-□-□	Amazon CloudFront が受信する 一部のダイナミックな HTTP リクエストヘッダーが、

はい-いいえ-N/A	パフォーマンス効率
	キャッシュヒット率を減少させ、オリジンに対する負荷を上昇させることで、パフォーマンスに影響を及ぼすことを理解している。詳細
□-□-□	EC2 インスタンスの最大スループットが、付加された EBS ボリュームの最大集合スループットよりも大きいことを確認している。EBS に最適化されたインスタンスを PIOPS EBS ボリュームとともに用いて、ボリュームから想定のパフォーマンスを得ている。
□-□-□	ソリューションデザインが、インフラストラクチャ内のボトルネックまたは、データベースやアプリケーションデザイン内のストレスポイントを伴わないようにしている。
□-□-□	Amazon CloudWatch またはサードパーティーパートナーのツールを用いて、アプリケーションリソースに対するモニタリングをデプロイし、パフォーマンス超過に基づくアラームを設定している。
□-□-□	セキュリティグループがアプリケーションインスタンスに付加された際に大量のルールを用いないようにデザインしている。セキュリティグループに大量のルールがあると、パフォーマンス低下の原因になります。

はい-いいえ-N/A	コストの最適化
□-□-□	インフラストラクチャイベントには、不必要なコストを避けるためにイベント後に除去する必要のある、余分に準備されたキャパシティーがあることを理解している。
□-□-□	EC2 インスタンスのサイズや、RDS DB インスタンスのサイズ、キャッシングクラスターノードのサイズと数、Redshift クラスターノートのサイズと数、EBS ボリュームのサイズを適切な規模にしている。
□-□-□	都合に応じてスポットインスタンスを使用している。スポットインスタンスは、フレキシブルな開始時間および終了時間のあるワークロードに最適です。以下は、スポットインスタンスに最適なユースケースの例です。バッチ処理、レポート作成、ハイパフォーマンスコンピューティング (HPC) ワークロード
□-□-□	想定可能なアプリケーションキャパシティーの最低要件を有し、リザーブドインスタンスを活用している。 . リザーブドインスタンスでは、オンデマンドインスタンスの料金に比べて、時間単位での費用の安さと引き換えに、Amazon EC2 のコンピューティングキャパシティーを確保することができます。

Notes

[1] https://aws.amazon.com/answers/account-management/aws-tagging-strategies/

[2] https://aws.amazon.com/blogs/aws/resource-groups-and-tagging/

[3] https://aws.amazon.com/sqs/

[4] http://docs.aws.amazon.com/general/latest/gr/rande.html

[5] https://aws.amazon.com/emr/

[6] https://aws.amazon.com/rds/

[7] https://aws.amazon.com/ecs/

[8] https://aws.amazon.com/sns/

[9] https://aws.amazon.com/blogs/compute/using-aws-lambda-with-auto-scaling-lifecycle-hooks/

[10] http://docs.aws.amazon.com/lambda/latest/dg/welcome.html

[11] https://aws.amazon.com/blogs/aws/new-auto-recovery-for-amazon-ec2/

[12] https://aws.amazon.com/answers/configuration-management/aws-infrastructure-configuration-management/

[13] https://d0.awsstatic.com/whitepapers/Big_Data_Analytics_Options_on_AWS%20.pdf

[14] http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html#routing-policy-latency

[15] https://aws.amazon.com/elasticache/

[16] https://aws.amazon.com/cloudfront/

[17] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts-on-demand-reserved-instances.html

[18] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html

[19] https://docs.aws.amazon.com/general/latest/gr/aws_service_limits.html

[20] https://aws.amazon.com/about-aws/whats-new/2014/07/31/aws-trusted-advisor-security-and-service-limits-checks-now-free/

[21] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring_automated_manual.html

[22] http://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch_Dashboards.html

[23] http://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/publishingMetrics.html

[24] https://aws.amazon.com/blogs/aws/new-whitepaper-use-aws-for-disaster-recovery/

[25] http://media.amazonwebservices.com/AWS_Operational_Checklists.pdf

[26] http://d0.awsstatic.com/whitepapers/architecture/AWS_Well-Architected_Framework.pdf

[27] https://aws.amazon.com/premiumsupport/iem/