単語	概要	重要度
EC2 Auto Scaling ウォームプール	事前に初期化されたEC2インスタンスのプール。起動時間の短縮。 停止状態のウォームプールインスタンスはコスト低。停止状態から素早く起動できるため、初期化時間を短縮できます。	
EC2の詳細モニタリング	標準モニタリングは5分間隔ですが、詳細モニタリングを有効にすると1分間隔でメトリクスが収集されます	
CloudWatchエージェント	メモリ使用率、ディスク使用率、ディスクI/O、スワップ使用率 EC2の標準メトリクスにはメモリ使用率が含まれていません。 ・AMI（Amazon Machine Image）にエージェントを含めてもOK.	
ALBと複数のターゲットグループ	異なる特性を持つワークロードを分離できる。 各ワークロードに最適なインスタンスタイプを選択できる	
サービスコントロールポリシー（SCP）	IAMポリシーで許可されているアクションでも、SCPで許可されていなければ拒否される	
EKSの主要コンポーネント	**コントロールプレーン**：Kubernetesマスターノードを管理し、APIサーバー、スケジューラー、コントローラーマネージャーなどを含む **ワーカーノード**：実際にアプリケーションコンテナを実行するEC2インスタンス **ポッド**：1つ以上のコンテナとその共有リソースのグループ。Kubernetesの最小デプロイ単位 **サービス**：ポッドのセットとそれらにアクセスするためのポリシーを定義	
EKS	アプリケーション固有の問題を特定するためには、ポッドレベルのメトリクスが最も適している	
CloudWatch **ContainerInsights**	EKSクラスタやKubernetesクラスタのパフォーマンスメトリクスを収集するためのCloudWatchの機能。pod_memory_utilizationメトリクスで、個々の**Kubernetesポッドのメモリ使用率**が分かる。	1
CloudWatchAgentServerPolicy	CloudWatchエージェントがメトリクスデータをCloudWatchに送信するために必要な権限を提供する管理ポリシーです	
S3バケットポリシー	条件キー： aws:**SourceAccount**：リクエストの送信元のアカウントを指定 aws:**PrincipalOrgID**：プリンシパルが属する組織を指定 aws:**PrincipalTag**：プリンシパルに付けられたタグに基づいてアクセスを制御	
アクセス制御階層	SCP → **IAM権限境界** → IAMポリシー → リソースポリシー	
Lambda関数	**大規模データに対して非効率的**です。Lambda関数には15分の実行時間制限と一時ストレージの制限（/tmpディレクトリで10GB）があるため、大きなオブジェクトの処理に適していません。	
S3バッチオペレーション	S3バッチオペレーションは、**大量のS3オブジェクト**に対する一括操作を提供するサービスです： サポートされる操作: コピー、タグ付け、ACL変更、アーカイブ、Lambda関数実行、レプリケーション	1
S3レプリケーション	クロスリージョンレプリケーション: 異なるリージョン間でのレプリケーション 同一リージョンレプリケーション: 同一リージョン内でのレプリケーション 双方向レプリケーション: 2つのバケット間で相互にレプリケーションを設定 レプリケーション失敗の検出は、EventBridge	
S3イベント通知	バケット内でのオブジェクト操作を検出して通知する機能です： イベントタイプ: オブジェクトの作成、削除、レプリケーション状態変更など 通知先: SNS、SQS、Lambda、EventBridge	
S3バッチオペレーション	S3レプリケーション失敗をEventBridgeで検知し、Lambdaへ送信。LambdaでS3バッチオペレーションジョブを作成。	
CloudFormationテンプレート	**ブルー/グリーンデプロイメント**は、デプロイ中の高可用性の維持 **Canary**: 2ステップの移行（最初に10%、その後残りの90%など） **Linear**: 等間隔のステップでトラフィックを移行(1分ごとに10%ずつ)	1

ECS (Elastic Container Service)	主要コンポーネント: タスク: 一緒にデプロイされる1つ以上のコンテナの集まり。最小のデプロイ単位。 クラスター: タスクが実行されるインフラのグループ。EC2起動タイプまたはFargate起動タイプで実行可能。
CloudWatch Logs Insights:	ログデータを効率的にクエリするための専用クエリ言語 複雑なフィルターとパターンマッチング クエリ結果の視覚化
ECRのイメージスキャン 拡張スキャン	Amazon Inspectorを使用 OSだけでなく、プログラミング言語(Python、JavaScriptなど）もスキャン AWS Security Hubと統合可能 重大度評価（CRITICAL、HIGH、MEDIUM、LOW）
ECRのスキャン結果	**EventBridge + Lambda関数**は、Amazon ECRのスキャン結果を評価し、必要なアクションを実行するために使用できます：
EC2 Image Builder	**コンテナイメージ**の作成に特化したサービス **複数のリージョンへの配布をサポート** セキュリティ標準化、**定期的なスケジュール実行**、**複数リージョン**への直接配布
コンテナレシピ	コンテナイメージを作成するための設計図です。以下の要素を含みます： ベースイメージ（Docker Hubや既存のECRイメージなど） 追加するコンポーネント（セキュリティ更新プログラム、カスタムソフトウェアなど） テスト設定 配布設定（ECRリポジトリ、リージョンなど）
ECR（Elastic Container Registry）	イメージの安全な保存と管理 IAMとの統合によるアクセス制御 イメージの脆弱性スキャン
ECRのリージョナル特性	ECRはリージョナルサービスであり、各リージョンに独立したリポジトリが存在。クロスリージョンアクセスは可能だが、レイテンシーやデータ転送コストが発生
ECRレプリケーション:	同一リージョン内の異なるリポジトリ間、または異なるリージョン間で設定可能
CodePipeline:	CI/CDパイプラインの自動化ツール ソース、ビルド、テスト、デプロイなどの段階を定義可能 スケジュールされたパイプライン実行をサポート
CodeBuild:	コンテナイメージのビルドに使用可能 ただし、EC2 Image Builderのようなコンテナ特化機能はない
CodeDeploy:	EC2インスタンス、Lambda関数、ECSサービスへのデプロイをサポート ECRへのコンテナイメージの登録や管理には最適ではない
CodePipeline	**ステージとアクション**：パイプラインは複数のステージ（例：ソース、ビルド、テスト、デプロイ）で構成。**各ステージ**には1つ以上のアクションが含まれます。 **アクションプロバイダー**：各アクションは特定のプロバイダー（AWS CodeBuild、AWS Lambda、Amazon ECSなど）によって実行されます。 **アーティファクト**：アクションの入出力はアーティファクトとして管理され、ステージ間で受け渡しされます。
ECS（Elastic Container Service）	Dockerコンテナを実行するためのフルマネージドコンテナオーケストレーションサービスです。タスク定義：コンテナの構成 **クラスター：**コンテナが実行されるEC2インスタンスまたはFargateタスクのグループです。
CodePipelineからECSにデプロイ	**imagedefinitions.json**：ECSデプロイアクションに必要なファイルで、コンテナ名とイメージリポジトリのURI（タグを含む）を指定します **ECSデプロイアクション**：CodePipelineのデプロイステージにECSアクションを追加し、タスク定義やサービス名を指定することで、新しいイメージでサービスを更新します。

1

1

CI/CDパイプラインに統合テストを追加	**AWS Lambda：**API呼び出しやエンドポイントチェックなどの軽量テストを実行するのに適しています。 **AWS CodeBuild**：より複雑なテストスイートを実行するためのビルド環境を提供します。 **AWS Step Functions**：複雑なテストワークフローのオーケストレーションに使用できます。	
ECSアクションプロバイダ	既存のパイプラインにデプロイステージを追加し、アクションプロバイダーとしてAmazon ECSを設定します。これにより、新しいコンテナイメージを既存のECSクラスターに簡単にデプロイできます。	
Kinesis 拡張ファンアウト（Enhanced Fan-out）	**各コンシューマが独立**した2MB/秒のスループットを獲得 通常200ミリ秒以下の低レイテンシーを実現	1
Lambda EventSourceMapping	ストリーミングサービスとLambda関数を連携させる設定です： **ParallelizationFactor**を増加させることで、Lambda関数が同時に複数のバッチを処理できるようになり、処理のスループットが向上	
Lambda同時実行制御	Lambdaには複数の同時実行制御メカニズムがあります： **プロビジョンドコンカレンシー**（Provisioned Concurrency）：事前にウォームアップされた実行環境を予約	1
CloudWatch LogsとKinesisの統合機能	**サブスクリプションフィルター**: ログデータを自動的にKinesisに送信 **リアルタイムストリーミング**: ログ生成と同時にストリーミング開始	
SQS デッドレターキュー	処理に複数回失敗したメッセージ（無効なデータを含むメッセージ）をメインキューから分離する	
Aurora**クロスリージョン**フェイルオーバーはアプリの参照変更が必要。	AuroraエンドポイントをSSM **Parameter Store**に保存し、障害検出時にEventBridgeイベントを使用してLambda関数をトリガーし、レプリカを昇格させ、Parameter Store内のエンドポイントを更新する。**各アプリはParameter Storeを参照する。**	1
SAM(Serverless Application Model)	AWS SAMは、サーバーレスアプリケーションを開発・デプロイするためのオープンソースフレームワークです。CloudFormationを拡張し、Lambda関数、API Gateway、DynamoDBなどのサーバーレスリソースを簡素化された構文で定義できるようにします。デプロイ時にCloudFormationテンプレートに変換されます。	
SAM(Serverless Application Model)	CodeUriに**S3パスが指定されている場合**、SAMは単にそのパスを参照するだけで、コードの内容や**変更を検出しません**。CodeUriを**ローカルディレクトリ**に変更すると、SAMはデプロイ時にコードを読み取り、パッケージ化してS3にアップロードします	1
S3クロスリージョンレプリケーション	異なるAWSリージョン間でS3オブジェクトを自動的に複製する機能です： **非同期レプリケーション**: オブジェクトがソースバケットにアップロードされた後、通常数分以内にターゲットバケットに複製されます	
S3双方向レプリケーション	2つのS3バケット間で相互にレプリケーションを設定する手法です：	
S3 Batch Operations	大量のS3オブジェクトに対して一括操作を実行するためのサービスです：	
S3クロスリージョンレプリケーションの手順	１．S3およびS3 Batch Operationsサービスプリンシパルに必要なS3レプリケーションの権限を付与するIAMロールを作成します。 ２．S3 **Batch Operations**を実行して、**既存データコピー** ３．**双方向レプリケーションルール**を設定	1
Step Functions	ワークフローを視覚的に構築し管理できるサービスです： **ステートマシン**: JSON形式で定義されるワークフロー **ステート（状態）**：ワークフローの各ステップ。Task、Choice、Parallel、Mapなど様々なタイプがある **Step Functions + Lambda**：ワークフロー管理、状態追跡、エラーハンドリング、再処理の機能が組み込まれており、サーバーレスで効率的	

API Gateway	主な認証方法: IAM認証: AWS IAMユーザー/ロールに基づく認証 Lambda オーソライザー: カスタムロジックによる認証 Cognito ユーザープール: ユーザー管理と認証 APIキー: シンプルなキーベースの認証 OAuth/JWT トークン: トークンベースの認証	
API Gateway	特定のOrganizational Unit（OU）内のエンティティのみがAPI Gatewayにアクセスできるようにする場合： ・**すべてのAPIメソッドにIAM認証**を有効化し、認証方法としてIAMを設定。 ・IAM認証を有効にした場合、クライアントは**Signature Version 4（SigV4）**使用してリクエストを署名する必要あり。 →API Gatewayで呼び出し元のIAMアイデンティティを識別可能となる。	1
ECRのスキャン機能	**高度スキャン**: Amazon **Inspector**を使用したより包括的なスキャン スキャン結果は重要度（CRITICAL、HIGH、MEDIUM、LOW、INFORMATIONALなど）でラベル付け	
Dockerfile	Docker imageの設計図（レシピ）	
Docker image	Dockerfile から作られる実体（完成した環境）	
Docker container	Docker imageを実際に動かした実行中の環境	
Kubernetesマニフェスト	コンテナをどう動かすかの設計図	
Control Tower	マルチアカウントAWS環境を設定および管理するためのサービスです **ランディングゾーン**：複数のAWSアカウントを管理するための基盤となる環境 **プロアクティブ（予防的）**コントロールと**ディテクティブ（検出的）**コントロールの2種類	1
Control Tower プロアクティブコントロール	**非準拠なリソースの作成を事前に防ぐ**機能です。これは**CloudFormationフック**を使用して実装され、リソースがCloudFormationテンプレートによって作成される際に、定義されたポリシーに従って検証を行います。検証に失敗した場合、**リソースの作成自体が阻止**されます。	1
Control Tower ディテクティブコントロール	AWS **Config**とAWS **CloudTrail**を基盤として動作し、コンプライアンス状況の可視化を提供	
CloudFormationフック	CloudFormationスタックのライフサイクルの特定のポイントで実行されるカスタム検証やアクションを定義する仕組みです。**リソース作成前（Pre-create）**、作成後（Post-create）、**更新前（Pre-update）**、更新後（Post-update）、削除前（Pre-delete）などのタイミングでフックを実行できます。	1
CloudFormation StackSets	単一の操作で複数のアカウントとリージョンにわたってスタックを作成、更新、または削除できます	
予防的および検出的制御	**予防的制御**：SCPs、IAMポリシー、セキュリティグループ、**CloudFormationフック** **検出的制御**：AWS Config Rules、監査ログ、コンプライアンスチェックなど	
Kinesis Data Firehose	ストリーミングデータを収集し、**Lambda関数を使用して変換し**、様々な宛先に配信する	1
IAMロール	認証情報は自動的にローテーションされ、**短期間のみ有効**	
IAMインスタンスプロファイル	AWS API呼び出し時に自動的に**一時的な認証情報**を提供	1
EC2起動テンプレート	AMI、インスタンスタイプ、ネットワーク設定、**IAMロール**などを指定可能。 **IAMインスタンスプロファイルを統合可能。**	1
CloudFormation	S3バケットが空でない場合、CloudFormationはデフォルトでバケットの削除に失敗する。→ **Lambdaでバケットを空にする処理を行う。**	1
CodeArtifact	ソフトウェア開発のためのパッケージ管理サービス ・ソフトウェアパッケージを保存および共有するためのリポジトリを提供 ・パッケージバージョンのステータス管理。公開、非表示、**アーカイブ**、削除済み。 ・アップストリームリポジトリ	

CodeArtifact アーカイブ	アーカイブされたパッケージバージョンは**ダウンロードや使用が制限**されるため、脆弱なバージョンのダウンロードを効果的にブロックできます。	1
CodeArtifactパッケージのオリジンコントロール設定	**アップストリーム操作を防止**しながら直接公開を有効にする。パブリックなアップストリームリポジトリから脆弱なパッケージバージョンが自動的に取り込まれることを防止できます。	
SCP	組織内のプリンシパル（**ユーザー、ロール、ルートユーザー**）が実行できるアクションの最大範囲を定義します。SCPでは条件キー「**aws:SourceIp**」を使用して、特定のIPアドレス範囲外からのAPI呼び出しを明示的に拒否できます。	
Glueクローラー	S3に保存されたデータを自動的にカタログ化し、スキーマを検出	
S3の双方向バケットレプリケーション	プライマリとセカンダリの両リージョンで行われた変更を自動的に反映します。管理オーバーヘッドがほとんどありません。	
DynamoDBグローバルテーブル	任意のリージョンで行われた変更は**通常1秒以内に他のすべてのリージョンに自動的に伝播**されます。これにより、リアルタイムの要件が満たされます。	
S3バッチオペレーション	大量のS3オブジェクトに対してバッチ処理を実行するためのツール	
ALBヘルスチェックの猶予期間	新しいタスクが起動してからヘルスチェックが成功するまでに許容される時間です。**新しいコンテナイメージは、以前のイメージよりも初期化に時間がかかる可能性があります**	1
EventBridge	CloudWatch LogsをEventBridgeルールの追加ターゲットとして設定することで、ルールに一致するすべてのイベントをログとして記録することができます。**イベントフローの問題を特定**するのに役立ちます。	
CloudTrailログ	CloudTrailログを分析することで、**イベントの整合性も確認**できます。特に、使用パターンの不一致や権限の問題などを特定するのに役立ちます。	
EventBridge	EventBridgeルールのターゲットに**SQSスタンダードキューをデッドレターキュー**として設定することで、イベントが指定されたターゲットに**正常に配信されなかった場合に、そのイベントをキャプチャ**することができます。	
CloudFormation スタックエクスポート/インポート機能	**スタックエクスポート/インポート機能**は、**あるスタックで作成されたリソースを別のスタックで参照する**ための仕組みです。 この方法なら**両チームが独立**して管理プロセスを維持	1
CloudFormation StackSets	複数のAWSアカウントやリージョンにわたって同じスタックを一度に作成、更新、または削除するための機能です。	
NetApp ONTAP	エンタープライズグレードの**ストレージオペレーティングシステム** 高度なストレージ効率性（**デデュプリケーション**、圧縮、コンパクションなど） 複数のプロトコルサポート（NFS、SMB、iSCSI、FC） スナップショット技術によるデータ保護	
NetApp ONTAP SnapMirror	強力なレプリケーション技術	
FSx for NetApp ONTAP	NetApp ONTAPのファイルシステムをAWS上で提供するフルマネージドサービスです。高いパフォーマンスとスケーラビリティ	1
Config	AWSリソースの設定を評価し、指定されたルールに対する準拠状態を継続的に監視する。**組織ルール（Organization Rules）**を作成することで、AWS Organizations内のすべてのアカウントに一貫したルールを適用できます。**SCP**を実装してAWS Configの無効化や削除を制限可能。	
Route 53のフェイルオーバールーティング	プライマリリージョンで障害が発生した場合、トラフィックが自動的にセカンダリリージョンにリダイレクトされます。ヘルスチェックにより、リージョンの状態が継続的に監視され、問題が検出されると自動的にフェイルオーバーがトリガーされます。Route 53のDNS変更は**通常数秒から数分**で反映される	
Auroraグローバルデータベース	複数のAWSリージョンにまたがるデプロイを可能。通常1秒未満の遅延でセカンダリリージョンに複製される。マネージドフェイルオーバーまたは手動プロセスを通じて、セカンダリリージョンを新しいプライマリリージョンに昇格。**通常数分。**	
IAM 権限境界	権限境界は主に、IAMユーザーやロールが**自分自身の**権限を昇格させることを防ぐために使用されます。	