Amazon AppStream 2.0 をデプロイするためのベストプラクティス

Amazon AppStream 2.0 をデプロイするためのベストプラクティス:

Table of Contents

Amazon AppStream 2.0 のデプロイのベストプラクティス

出版日:2022 年 1 月 19 日 (ドキュメントの改訂)

要約

このホワイトペーパーでは、Amazon AppStream 2.0 のデプロイに関する一連のベストプラクティスについて概説しています。このホワイトペーパーでは、Amazon 仮想プライベートクラウド (VPC) の設計、イメージの作成と管理、フリートのカスタマイズ、フリートの自動スケーリングの戦略について説明しています。これには、ユーザー接続方法、認証、および Microsoft Active Directory との統合が含まれます。このホワイトペーパーには、AppStream 2.0 のセキュリティ、監視、コスト最適化の設計に関する推奨事項も含まれています。

このホワイトペーパーは、関連情報にすばやくアクセスできるようにするために作成されています。ネットワークエンジニア、アプリケーションデリバリースペシャリスト、ディレクトリエンジニア、またはセキュリティエンジニアを対象としています。

序章

Amazon AppStream 2.0 は、デスクトップアプリケーションに即座にアクセスできるようにする、完全マネージド型のアプリケーションストリーミングサービスです。AppStream 2.0 は、アプリケーションをホストして実行するために必要な AWS リソースを管理します。自動的にスケールし、ユーザーにオンデマンドでアクセスを提供します。AppStream 2.0 を使用すると、エンドユーザーは選択したデバイスで必要なアプリケーションにアクセスできます。これは、ネイティブにインストールされたアプリケーションと区別がつかず、反応に優れたユーザーエクスペリエンスを提供します。

以下のセクションでは、Amazon AppStream 2.0 の詳細、サービスの仕組み、サービスの起動に必要なもの、使用できるオプションと機能について説明します。AppStream 2.0 をエンドユーザーにデプロイする場合、優れたユーザーエクスペリエンスを提供するためのベストプラクティスを実装することが重要です。さらに、あらゆる規模の企業が、毎月の運用コストを削減し、コスト最適化によるメリットを享受できます。

主要なコンセプト

AppStream 2.0 を最大限に活用するには、以下のコンセプトを理解しておく必要があります。

- イメージ — イメージとは、事前に設定されたインスタンスのテンプレートです。イメージには、ユーザーにストリーミングできるアプリケーションと、ユーザーがアプリケーションの使用をすばやく開始できるようにするためのデフォルトの Windows 設定とアプリケーション設定が含まれています。AWS は、Image Builder を作成するために使用できるベースイメージを提供しています。その後、その Image Builder で独自のアプリケーションが含まれるイメージを作成します。イメージの作成後にイメージを変更することはできません。他のアプリケーションの追加、既存のアプリケーションの更新、またはイメージ設定の変更を行うには、新しいイメージを作成する必要があります。イメージは、他の AWS リージョン にコピーしたり、同じリージョン内の他の AWS アカウント と共有したりすることができます。
- Image Builder は、Image Builder は、イメージの作成に使用する仮想マシンです。Image Builder を起動して接続するには、AppStream 2.0 コンソールを使用します。Image Builder に接続すると、アプリケーションをインストール、追加、テストできます。さらに Image Builder を使用してイメージを作成できます。自己所有のプライベートイメージを使用して新しい Image Builder を起動できます。
- フリート - フリートは、指定したイメージを実行するフリートインスタンス (ストリーミングインスタンスとも呼ばれる) で構成されます。フリートに必要なストリーミングインスタンスの数を設定し、要求に基づいてフリートを自動的に拡張するようにポリシーを設定できます。各ユーザーに 1 つのインスタンスが必要です。
- スタック - スタックは、関連付けられたフリート、ユーザーアクセスポリシー、ストレージ構成で構成されます。ユーザーに対してストリーミングアプリケーションを開始するためにスタックを設定します。
- ストリーミングインスタンス — ストリーミングインスタンス (フリートインスタンスとも呼ばれます) は、1 人のユーザーがアプリケーションストリーミングに使用できる Amazon Elastic Compute Cloud (Amazon EC2) インスタンスです。ユーザーのセッションが完了すると、インスタンスは Amazon EC2 によって終了します。

VPC の設計

設計のガイドライン

AppStream 2.0 を専用 VPC にデプロイします。AppStream 2.0 VPC を設計する際は、予測される成長に合わせてサイズを設定します。IP アドレス容量を新しいユースケース用に、また後で追加される可能性のあるアベイラビリティーゾーン (AZ) 用に確保します。AppStream 2.0 の基本的な設計のポイントは、1 人のユーザーだけが AppStream 2.0 インスタンスを利用できるようにすることです。IP スペースを割り当てるときは、1 人のユーザーを AppStream 2.0 インスタンスごとに 1 つの IP アドレスと考えます。AppStream 2.0 では、1 人のユーザーが複数の AppStream 2.0 インスタンスを利用することができます。そのため、IP スペースを計画する際には、AppStream 2.0 インスタンスの追加を必要とするユースケースも考慮する必要があります。

VPC Classless Inter-Domain Routing (CIDR) の最大サイズは /16 ですが、AWS では、プライベート IP アドレスを過剰に割り当てないことをお勧めします。CIDR を追加して VPC のサイズを拡張することは可能ですが、これには制限があります。そのため、最初から必要なものを割り当てます。

AppStream 2.0 デプロイが Active Directory ドメインに参加している場合、VPC に設定されている DHCP オプションにはドメイン DNS が設定されている必要があります。ドメインネームサーバーは、Active Directory ドメインに対して権限のある DNS IP アドレスを指定するか、DNS が Active Directory ドメインの権限のある DNS インスタンスに DNS リクエストを転送する必要があります。また、VPC には `enableDnsHostnames` と `EnableDnsSupport` が設定されている必要があります。

アベイラビリティーゾーン

アベイラビリティーゾーンは、AWS リージョン の冗長電源、ネットワーク、および接続を備えた 1 つ以上の個別のデータセンターです。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性、耐障害性、および拡張性が優れています。

Amazon AppStream 2.0 では、フリートを起動するために必要なサブネットは 1 つだけです。ベストプラクティスは、少なくとも 2 つのアベイラビリティーゾーン (固有のアベイラビリティーゾーンごとに 1 つのサブネット) を設定することです。フリートの自動スケーリングを最適化するには、3 つ以上のアベイラビリティーゾーンを使用してください。水平スケーリングには、サブネットに IP スペースを追加して拡張できるという利点があります。これについては、このドキュメントの次の「サブネットのサイズ設定」セクションで説明します。AWS マネジメントコンソールでは、フリートの

作成時に指定できるサブネットは 2 つだけです。AWS Command Line Interface (AWS CLI) または AWS CloudFormation を使用して、3 つ以上のサブネット ID を許可します。

サブネットのサイズ設定

ルーティングポリシーとネットワークアクセスコントロールリストを柔軟に設定できるように、サブネットを AppStream 2.0 フリート専用にします。多くの場合、スタックには個別のリソース要件があります。例えば、AppStream 2.0 スタックには分離要件があり、ルールセットを区分する方法を提供します。複数の Amazon AppStream 2.0 フリートが同じサブネットを使用する場合は、すべてのフリートの最大キャパシティの合計が、使用可能な IP アドレスの総数を超えないようにします。

同じサブネット内のすべてのフリートの最大キャパシティが、使用可能な IP アドレスの総数を超える可能性がある、または超えた場合は、フリートを専用のサブネットに移行します。これにより、自動スケーリングイベントによって割り当てられた IP スペースが枯渇するのを防ぐことができます。フリートの合計キャパシティが、割り当てられたサブネットの割り当てられた IP スペースを超える場合は、API または AWS CLI の「フリートの更新」を使用してさらにサブネットを割り当てます。詳細については、「Amazon VPC クォータとその拡大方法」を参照してください。

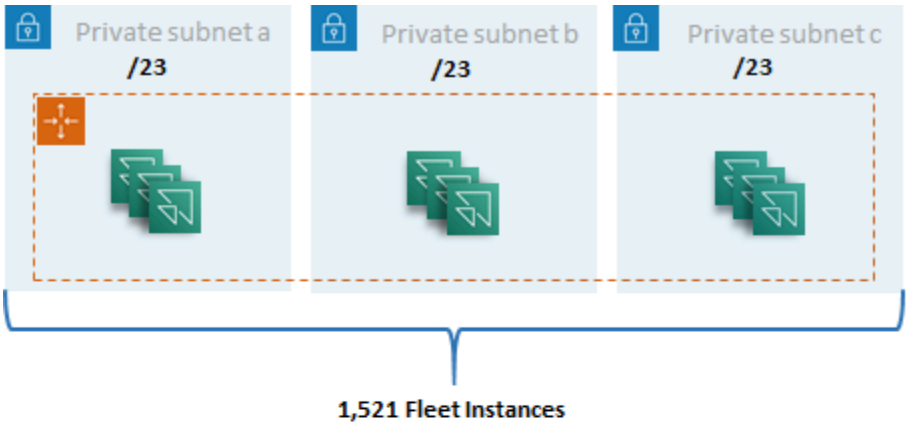
ベストプラクティスとしては、サブネット数をスケールアウトし、それに応じてサブネットのサイズを設定して、VPC のキャパシティを拡大することです。また、AppStream 2.0 フリートの最大数が、サブネットによって割り当てられた合計 IP スペースを超えないようにします。IP スペースの合計量を計算する際には、AWS のサブネットごとに 5 つの IP アドレスが予約されます。3 つ以上のサブネットを使用して水平方向にスケーリングする場合、次のようないくつかの利点があります。

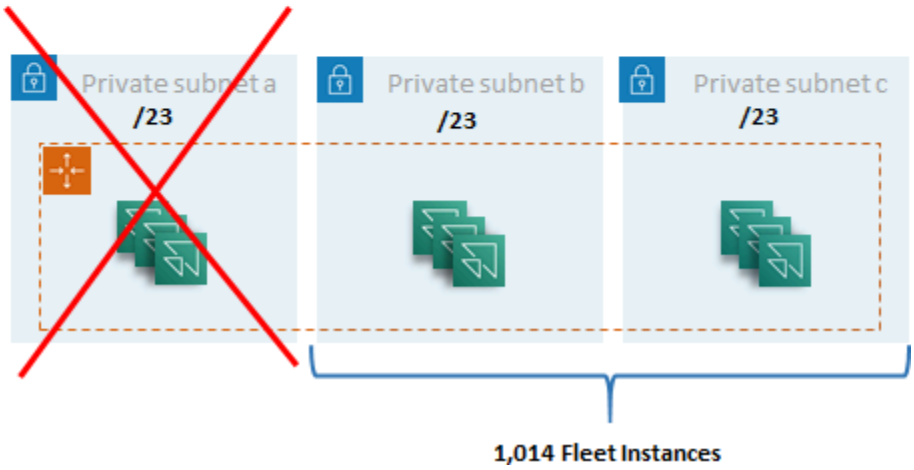
- アベイラビリティーゾーンの障害に対する耐性の向上
- フリートインスタンスを自動スケーリングする場合のスループットの向上
- プライベート IP アドレスをより効率的に使用して IP バーンを回避

Amazon AppStream 2.0 のサブネットのサイズを設定するときは、サブネットの総数と、使用率のピーク時に予想されるピーク同時実行数を考慮してください。これは、フリートに対して (`InUseCapacity`) とリザーブドキャパシティ (`AvailableCapacity`) を使用することでモニタリングできます。Amazon AppStream 2.0 では、使用済みおよび使用可能な AppStream 2.0 フリートインスタンスの合計に `ActualCapacity` のラベルが付けられます。合計 IP スペースを適切なサイズに設定するには、必要な `ActualCapacity` を予測し、フリートに割り当てられたサブネットの数から、耐障害性のための 1 つのサブネットを引いた数で割ります。

例えば、ピーク時に予測されるフリートインスタンスの最大数が 1000 で、1 つのアベイラビリティーゾーンに障害が発生しても回復力を維持することがビジネス要件である場合、3 x/23 サブネットで技術要件とビジネス要件を満たすことができます。

- /23 = 512 ホスト — 5 リザーブド = サブネットあたり 507 フリートインスタンス
- 3 サブネット — 1 サブネット = 2 サブネット
- 2 サブネット x サブネットあたり 507 フリートインスタンス = ピーク時 1,014 フリートインスタンス





サブネットのサイズ設定の例

2 x /22 サブネットでも耐障害性は十分ですが、次の点を考慮してください。

- 1,536 の IP アドレスを予約する代わりに、2 つの AZ を使用すると 2,048 の IP アドレスが予約され、他の機能に使用できる IP アドレスが無駄になります。

- 1 つの AZ にアクセスできなくなった場合、フリートインスタンスをスケールアウトする機能は AZ のスループットによって制限されます。これにより、PendingCapacity の期間が延長される可能性があります。

サブネットのルーティング

AppStream 2.0 インスタンス用のプライベートサブネットを作成し、アウトバウンドトラフィック用に一元管理された VPC を介してパブリックインターネットにルーティングするのがベストプラクティスです。AppStream 2.0 セッションストリーミングのインバウンドトラフィックは、ストリーミングゲートウェイ経由で Amazon AppStream 2.0 サービスを通じて処理されます。このためにパブリックサブネットを設定する必要はありません。

リージョン内の接続

Active Directory ドメインに参加している AppStream 2.0 フリートインスタンスでは、各 AWS リージョン の共有サービス VPC で Active Directory ドメインコントローラを設定します。Active Directory のソースは、Amazon EC2 ベースのドメインコントローラでも AWSMicrosoft マネージド AD でもかまいません。共有サービスと AppStream 2.0 VPC 間のルーティングは、VPC ピアリング接続またはトランジットゲートウェイのいずれかを介して行うことができます。トランジットゲートウェイは大規模なルーティングの複雑さを解決しますが、VPC ピアリングがほとんどの設定で望ましい理由はいくつかあります。

- VPC ピアリングは 2 つの VPC 間の直接接続です (余分なホップはありません)。
- 時間単位の課金はなく、アベイラビリティーゾーン間の標準データ転送料金のみです。
- 帯域幅に制限はありません。
- VPC 間のセキュリティグループへのアクセスのサポート。

これは、AppStream 2.0 インスタンスが、共有サービス VPC 内の大きなデータセットを持つアプリケーションインフラストラクチャやファイルサーバーに接続する場合に特に当てはまります。これらの一般的にアクセスされるリソースへのパスを最適化することで、他のすべての VPC およびインターネットルーティングがトランジットゲートウェイを介して実行される設計でも、VPC ピアリング接続が優先されます。

アウトバウンドインターネットトラフィック

共有サービスへの直接ルーティングはほとんどピアリング接続を通じて最適化されますが、AppStream 2.0 のアウトバウンドトラフィックは、AWS トランジットゲートウェイを使用して

複数の VPC から単一のインターネットの出口を作成することで設計できます。マルチ VPC 設計では、すべての送信インターネットトラフィックを制御する専用 VPC を用意するのが標準的な方法です。この設定では、トランジットゲートウェイがより柔軟になり、サブネットにアタッチされた標準ルーティングテーブルでのルーティングを制御できるようになります。この設計では、複雑さを増すことなく推移的なルーティングもサポートされ、冗長なネットワークアドレス変換 (NAT) ゲートウェイ、つまり各 VPC 内の NAT インスタンスが不要になります。

すべてのアウトバウンドインターネットトラフィックが単一の VPC に集中化されると、NAT ゲートウェイまたは NAT インスタンスが一般的な設計上の選択肢になります。どちらが組織にとって最適かを判断するには、管理ガイドの NAT ゲートウェイと NAT インスタンスの比較を参照してください。AWSNetwork Firewall は、ルートレベルで保護し、OSI モデルのレイヤー 3 から 7 までのステートレスルールとステートフルルールを提供することで、セキュリティグループやネットワークのアクセス制御レベルを超えて保護を拡張できます。詳細については、「AWS Network Firewall のデプロイモデル」を参照してください。組織が URL フィルタリングなどの高度な機能を実行するサードパーティ製品を選択した場合は、そのサービスをアウトバウンドインターネット VPC にデプロイします。これは NAT ゲートウェイや NAT インスタンスの代替となります。サードパーティベンダーが提供するガイドラインに従ってください。

オンプレミス

オンプレミスのリソースへの接続が必要な場合、特に Active Directory に参加している AppStream 2.0 インスタンスへの接続が必要な場合は、AWS Direct Connect を介して耐障害性の高い接続を確立してください。

VPC エンドポイント

Amazon S3 VPC エンドポイント

多くの AppStream 2.0 デプロイでは、ホームフォルダとアプリケーション設定を通じてユーザーの状態を永続化します。これらの Amazon Simple Storage Service (Amazon S3) の場所に対してプライベート通信を可能にします。これにより、パブリックインターネットを使用する必要がなくなります。これは VPC エンドポイントゲートウェイを通じて実現できます。VPC エンドポイントゲートウェイは、次の理由から Amazon S3 の AWS PrivateLink よりも優先されます。

- AppStream 2.0 ネットワークアクセス要件に合わせてコスト最適化されている
- オンプレミスリソースから Amazon S3 バケットにアクセスする必要がない
- カスタムポリシードキュメントを使用して、アクセスを AppStream 2.0 インスタンスからのみに制限できる

VPC エンドポイントゲートウェイを作成したら、カスタムポリシーを作成してプライベート化された接続を保護するのがベストプラクティスです。カスタムポリシーは AppStream 2.0 サービスの Identity and Access Management ロールの Amazon リソースネーム (ARN) から開始します。ユーザー状態の保持に必要な S3 アクションを明示的に指定します。

> **Note**
>
> Resources セクションの以下の例では、ステートホームフォルダのパスを最初に指定し、次にアプリケーション設定パスを指定しています。

Example

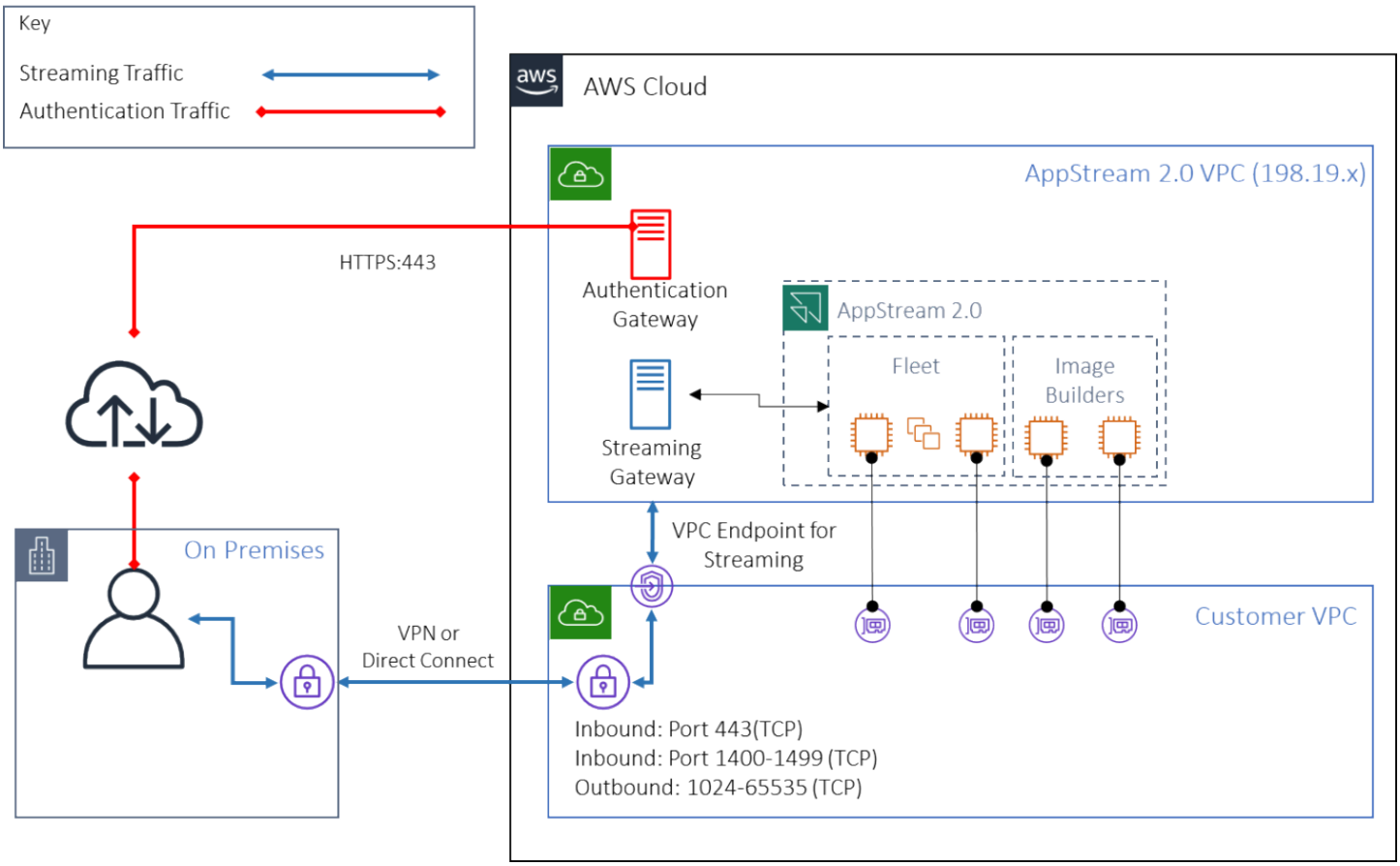
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow-AppStream-to-access-home-folder-and-
application-settings",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:sts::account-id-without-hyphens:assumed-
role/AmazonAppStreamServiceAccess/AppStream2.0"
      },
      "Action": [
        "s3:ListBucket",
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject",
        "s3:GetObjectVersion",
        "s3:DeleteObjectVersion"
      ],
      "Resource": [
        "arn:aws:s3:::appstream2-36fb080bb8-*",
        "arn:aws:s3:::appstream-app-settings-*"
      ]
    }
  ]
}
```

Amazon AppStream 2.0 API インターフェイス VPC エンドポイント

Amazon AppStream 2.0 への API コマンドと CLI コマンドが VPC から発信される設計シナリオでは、インターフェイス VPC エンドポイントを介してこれらのプログラム呼び出しをプライベート化します。

Amazon AppStream 2.0 ストリーミングインターフェイス VPC エンドポイント

Amazon AppStream 2.0 のストリーミングトラフィックをインターフェイス VPC エンドポイント経由でルーティングすることは可能ですが、この設定は注意して使用してください。パブリックインターネット経由のデフォルトのストリーミング動作は、Amazon AppStream 2.0 ストリーミングトラフィックの最も効率的でパフォーマンスの高い配信方法です。



Amazon AppStream 2.0 ストリーミングインターフェイス VPC エンドポイント

前の図に示したように、パブリックインターネットは Amazon AppStream 2.0 ストリーミングゲートウェイへの最も効率的なパスです。カスタマーマネージド VPC とネットワークを介したルーティ

ングは、複雑になり、レイテンシーが増加します。また、Direct Connect を介したデータ転送料金も加算されます。

> **Note**
>
> VPC エンドポイントではストリーミングのみがサポートされており、認証は引き続きパブリックインターネット上で行う必要があります。SAML シングルサインオン (SSO) ID プロバイダー (IdP) などの前提条件のアクセスは、依然としてパブリックインターネット経由でのみアクセスできる必要があります。

イメージの作成と管理

AppStream 2.0 でフリートまたは Image Builder を起動するときは、AppStream 2.0 ベースイメージのいずれかを選択する必要があります。その後、管理者はベースイメージに基づいて独自のアプリケーションや構成設定を追加できます。

イメージを構築する際には、アプリケーションが正しく安全に動作するようにするための重要な考慮事項があります。さらに、そのイメージをどのように管理するかについて、設計上の考慮事項があります。

AppStream 2.0 イメージの構築

新しいイメージを構築するときは、次の点を考慮することが重要です。

- オペレーティングシステム
- アプリケーション
- ユーザープロファイル
- セキュリティ
- パフォーマンス
- エージェントのバージョン
- Image Assistant CLI

AppStream 2.0 イメージの構築

2021 年 11 月、AppStream 2.0 は Amazon Linux 2 のサポートを開始しました。今回の発表により、AppStream 2.0 は次の 4 つのプラットフォームタイプをサポートするようになりました。

- Windows Server 2012 R2
- Windows Server 2016
- Windows Server 2019
- Amazon Linux 2

アプリケーションの要件に基づいて特定のプラットフォームを選択しなければならない場合があります (例えば、アプリケーションに Windows が必要な場合、Amazon Linux 2 はオプションを選択する

ことはできません)。アプリケーションの要件以外にも、以下の比較マトリクスを参照して、ユースケースと環境に最適なプラットフォームタイプを選択してください。

表 1 — プラットフォームの種類、どのようなときに使うか、料金

プラットフォームの種類	どのようなときに使うか	フリートの料金*
Windows Server (2012 R2、2016 または 2019)	アプリケーションは Windows でのみ実行できる (Amazon Linux 2 はサポートされていません)。ドメインをストリーミングインスタンスに参加させたい。AppStream 2.0 ストリーミングインスタンスで既存のグループポリシーを使用したい (Linux はグループポリシーに準拠していませんが、セッション開始時にセッションスクリプトを使用して設定を自動化できます)。デスクトップビューを使用し、ユーザーは Windows デスクトップエクスペリエンスを求めている。ステップバイステップのウィザードを備えた Image Assistant アプリケーションを使用して、アプリケーションカタログとイメージを作成すること希望している。現時点では、ターミナルコマンドを使用して Amazon Linux 2 イメージを作成する必要があります (詳細については、このチュートリアルを参照してください)。アプリケーション設定の永続化を使用したい。アプリケーション設定の永続化	RDS SAL (Microsoft リモートデスクトップサービスサブスクライバーアクセスライセンス) 料金 (固有のユーザー 1 人につき 1 か月あたり 4.19 USD) ** に加えて、以下が加算されます。 1. 常時オン、オンデマンドのフリートでは 1 時間あたり 0.10 USD 2. Elastic フリートは 1 時間あたり 0.15 USD

プラットフォームの種類	どのようなときに使うか	フリートの料金*
	の有効化は、現在 Linux ベースのスタックではサポートされていません。	
Amazon Linux 2	低コストのストリーミングインスタンスを使用して、RDS SAL ライセンス料を回避したい。アプリケーションは Amazon Linux 2 と互換性がある	Linux インスタンスは Windows インスタンスに比べてコストが低くなります。Linux では、RDS SAL 料金はなく、以下の時間単位の料金を支払います。 1. 常時オン、オンデマンドのフリートでは 1 時間あたり 0.084 USD 2. Elastic フリートは 1 時間あたり 0.112 USD

* バージニア北部リージョンの stream.standard.medium に基づく

** 対象となるカスタマーが自身のライセンスを使う場合、AWS RDS SAL 料金は発生しません。詳細については、「AppStream 2.0 の料金表」ページを参照してください。教育機関のカスタマーも特典の対象となる場合があります。学校、大学、および特定の公共機関によっては、Microsoft RDS SAL ユーザー料金の対象となる場合があります。

アプリケーション

アプリケーションをインストールする前に、アプリケーションの依存関係やハードウェア要件などのアプリケーション要件を確認することが重要です。Image Builder インスタンスにアプリケーションを正常にインストールしたら、必ずユーザーを切り替え、テストユーザーのコンテキストでアプリケーションをテストしてください。

アプリケーションのデプロイを計画するときは、サービスのエンドポイントとクォータに注意してください。さらに、イメージを作成する前に、インストーラファイルとヘルパーファイルをクリーンアップして C ドライブの総容量を最適化してください。AppStream 2.0 インスタンスには 200 GB の固定サイズボリュームが 1 つあることに注意してください。固定サイズのボリュームを超えないようにするには、インストール後にディスク容量を最適化することがベストプラクティスです。

ユーザーがリアルタイムでアクセスできるアプリケーションのカタログを変更したい場合、動的アプリケーションフレームワークにより API オペレーションが提供されます。動的アプリケーションプロバイダーによって管理されるアプリケーションは、イメージ内に存在することも、Windows ファイル共有やアプリケーション仮想化テクノロジーなどからインスタンス外に存在することもあります。この機能では、Microsoft Active Directory ドメインに参加している AppStream 2.0 フリートが必要です。詳細については、「AppStream 2.0 での Active Directory の使用」を参照してください。

App Block

App Block は、ユーザーが使用するアプリケーションを起動するのに必要なセットアップスクリプトとアプリケーションファイルを表します。仮想ハードディスク (VHD) は Amazon S3 のどのオブジェクトでも可能です。ユーザーがアプリケーションにアクセスするには完全にダウンロードする必要があるため、このオブジェクトは 1.5 GB 未満にすることをお勧めします。

App Block の最適化

Windows ベースのフリートでは、アプリケーションを格納する VHDX ファイルを作成することをお勧めします。Linux ベースのフリートでは、イメージ (IMG) を作成することをお勧めします。これらの仮想ディスクは、アプリケーションファイルをホストするために、できるだけ小さく作成する必要があります。仮想ディスクは圧縮してサイズをさらに削減できます。セットアップスクリプトでは、マウントする前にディスクを解凍する必要があります。Windows PowerShell セットアップスクリプトの例には、解凍機能が含まれています。アーカイブ (zip) の展開とダウンロード速度の間にはトレードオフがあります。アプリケーションの起動時間を最短にするバランスを見つけるには、何度かテストの実施が必要な場合があります。

アプリケーションの更新

アプリケーションには、軽微な変更と大規模な変更が発生する場合があります。軽微な更新の場合は、App Block ファイルをホストする Amazon S3 バケットでバージョニングを有効にします。この設定により、管理者は App Block の設定を変更せずに、該当するアプリケーションの VHD オブジェクトのバージョンを変更することで、特定のアプリケーションの以前のバージョンにロールバックできます。大規模な更新では、更新された VHD に対して新しい App Block を作成します。これにより、管理者はアプリケーションの主要な変更をバージョン管理レベルではなく、App Block レベルで区別できるようになります。そのため、管理アプリケーションをより整理して管理できます。

ユーザープロファイルのカスタマイズ

Amazon AppStream 2.0 は、設計上、非永続的なアプリケーションおよびデスクトップソリューションです。ユーザーセッションが終了すると、システム変更とユーザー変更の両方も終了します。アプ

リケーション設定の永続化は、必要な場合にのみ有効にしてください。ログオンプロセスにオーバーヘッドが加わり、必要な S3 ストレージのコストに関する考慮事項が増える可能性があります。

アプリケーション設定の永続化が必要な状況では、AWS は、カスタムポリシーと S3 VPC ゲートウェイエンドポイントを使用して接続を保護することをお勧めします。アプリケーション設定全体のサイズを評価し、アプリケーション設定の永続化に保存される設定を最小限に抑えて、コストとパフォーマンスを最適化します。

ユーザープロファイルのカスタマイズは AppStream 2.0 Image Builder インスタンスで設定できます。これには、レジストリキーの追加と変更、ファイルの追加、その他のユーザー固有の設定が含まれます。AppStream 2.0 Image Assistant には、ユーザープロファイルを作成するオプションがあります。これにより、テンプレートユーザープロファイルがデフォルトのユーザープロファイルにコピーされます。イメージがフリートにデプロイされると、そのフリートからセッションをストリーミングするエンドユーザーのユーザープロファイルは、デフォルトのユーザープロファイルから作成されます。特にアプリケーション設定の永続化が有効になっている場合、ユーザープロファイルサイズの最小化を検討することが重要です。デフォルトでは、ユーザープロファイルの VHDx の最大サイズは 1 GB です。ストリーミングセッションが開始されるたびに、ユーザープロファイルの VHDx ファイルが S3 バケットからダウンロードされます。これにより、ストリーミングセッションの準備時間が長くなり、制限を超過するリスクが発生し、VHDx ファイルを使用したユーザープロファイルのマウントが失敗します。

1 GB を超えるユーザープロファイルが必要なユースケースについては、AWS では、プロファイルの保存に代替方法を使用することを推奨しています。例えば、Amazon FSx for Windows File Server などの共有ストレージで、ローミングプロファイルや FSLogix プロファイルコンテナを使用します。詳しくは、「Amazon FSx for Windows File Server と FSLogix を使用して Amazon AppStream 2.0 のアプリケーション設定の永続化を最適化する」を参照してください。

セキュリティ

開発者にはさまざまな考慮すべきセキュリティ対策があります。AppStream 管理者には、Windows オペレーティングシステムの更新プログラム、カスタマーのアプリケーション、それらの依存関係のインストールとメンテナンスを実施する責任があります。ベースイメージを最新の状態に保つためのその他のガイダンスについては、「AppStream 2.0 イメージを最新の状態に保つ」を参照して、ベースイメージを最新の状態に保つための追加のガイダンスを参照してください。

AppStream 2.0 では、デフォルトで、ユーザーまたはアプリケーションは、イメージアプリケーションカタログで指定されているプログラム以外にも、インスタンス上の任意のプログラムを起動できます。これは、アプリケーションがワークフローの一部として別のアプリケーションに依存しており、その依存アプリケーションをユーザーが直接起動できないようにしたい場合に便利です。例えば、ア

プリケーションはブラウザを起動してアプリケーションベンダーのウェブサイトからヘルプの説明を表示しますが、ユーザーにはブラウザを直接起動させたくない場合があります。状況によっては、ストリーミングインスタンスで起動できるアプリケーションを制御したい場合もあります。Microsoft AppLocker は、明示的な制御ポリシーを使用して、ユーザーが実行できるアプリケーションを有効または無効にするアプリケーション制御ソフトウェアです。

ウイルス対策ソフトウェアは、ストリーミングセッションや Image Builder インスタンスに悪影響を及ぼす可能性があります。AWS では、ウイルス対策ソフトウェアに、自動更新機能を使用しないことをお勧めします。Windows Defender について詳しくは、「ウイルス対策ソフトウェア」を参照してください。

パフォーマンス

新しいイメージを作成する前に、テストユーザーとしてアプリケーションをテストすることが重要です。テストユーザーとしてテストすることで、管理者以外のユーザーコンテキストでもアプリケーションを実行できることを確認できます。さらに、タスクマネージャーやパフォーマンスモニターなどの組み込みツールを使用して、アプリケーションのパフォーマンスとユーザーエクスペリエンスを確認します。CPU、メモリ、GPU メモリなどのリソース使用率をモニタリングするのがベストプラクティスです。CPU、メモリ、または GPU メモリのリソースに制約がある場合は、インスタンスタイプのアップグレードを検討してください。パフォーマンスを向上させるには:

- ブラウザのポップアップウィンドウを無効にする
- 拡張 IE セキュリティを無効にする

AppStream 2.0 エージェントのバージョン選択

新しいイメージを作成する場合、最新の AppStream 2.0 エージェントソフトウェアを使用するか、または更新しないかを選択できます。AppStream 2.0 エージェントソフトウェアには、バグの修正と機能拡張が含まれています。最新のソフトウェアでイメージを維持しましょう。このメカニズムについては、このドキュメントの「イメージ更新」セクションで確認してください。

[最新のエージェントを使用する] オプションを選択できます。このオプションでは、起動時に最新の AppStream 2.0 エージェントが常にインストールされます。ただし、予期しない変更がユーザーエクスペリエンスに影響を及ぼす可能性があり、エージェントを更新するとインスタンスの起動時間が長くなる可能性があります。ベースイメージを更新するには、イメージを再作成する必要があります。また、起動時間を最小限に抑えるには、更新したイメージを本番環境にロールアウトする前にテストすることも重要です。

Image Assistant Command Line Interface (CLI)

AppStream 2.0 のイメージを自動化またはプログラムで作成したい開発者は、Image Assistant CLI を使用してください。こでは、2019 年 7 月 26 日以降にリリースされた AppStream 2.0 エージェントソフトウェアを搭載した Image Builder でご利用いただけます。次の概要では、プログラムで AppStream 2.0 イメージを作成するプロセスについて説明します。

1. アプリケーションインストールの自動化を使用して、イメージビルダーに必要なアプリケーションをインストールします。このインストールには、ユーザーが起動するアプリケーション、依存関係、およびバックグラウンドアプリケーションが含まれる場合があります。
2. 最適化するファイルとフォルダーを決定します。
3. 該当する場合は、Image Assistant `add-application` CLI オペレーションを使用して、AppStream 2.0 イメージのアプリケーションメタデータと最適化マニフェストを指定します。
4. AppStream 2.0 イメージに追加のアプリケーションを指定するには、必要に応じてアプリケーションごとに手順 1 ~ 3 を繰り返します。
5. 該当する場合は、Image Assistant `update-default-profile` CLI オペレーションを使用して、デフォルトの Windows プロファイルを上書きし、ユーザーのデフォルトのアプリケーションと Windows 設定を作成します。
6. Image Assistant `create-image` CLI オペレーションを使用してイメージを作成します。

詳しくは、「Image Assistant CLI オペレーションを使用してプログラムで AppStream 2.0 イメージを作成する」を参照してください。

ユーザーのストリーミングエクスペリエンスの管理

セッションスクリプトを使ったカスタマイズ

AppStream 2.0 には、インスタンスセッションスクリプトが用意されています。ユーザーのストリーミングセッションで特定のイベントが発生したときに、これらのスクリプトを使用して独自のカスタムスクリプトを実行できます。たとえば、ユーザーのストリーミングセッションが開始される前に、カスタムスクリプトを使用して AppStream 2.0 環境を準備できます。ユーザーがストリーミングセッションを完了した後に、カスタムスクリプトを使用してストリーミングインスタンスをクリーンアップすることもできます。

AppStream 2.0 イメージ内にセッションスクリプトを指定します。セッションスクリプトの設定について詳しくは、管理ガイドの「セッションスクリプトの使用によるユーザーエクスペリエンスの

管理に関する」セクションを参照してください。ネットワーク共有または AWS Identity and Access Management(IAM) プロファイルと併用すると、セッションスクリプトを使用してストレージロケーションから追加のスクリプトを取得できます。この追加のスクリプトにより、ユーザーエクスペリエンスをさらに最適化できます。これにより、アプリケーション環境をユーザーに配信するのに必要なイメージとフリートの数を最小限に抑えることができます。

Active Directory グループポリシーの使用

Active Directory ドメインで AppStream 2.0 フリートを使用する予定の場合は、グループポリシーオブジェクト (GPO) を使用してユーザーエクスペリエンスを管理できます。GPO は AppStream 2.0 インスタンスが作成される組織単位 (OU) に割り当てることができます。イメージの作成を簡素化するには、継承をブロックする OU でベース AppStream 2.0 イメージを起動します。これにより、他のドメインポリシーが AppStream 2.0 のユーザーエクスペリエンスに影響を与えるのを防止します。各フリートを専用の OU にデプロイし、独自の GPO を使用して環境を確立することで、AppStream 2.0 のイメージ管理のメリットを 1 対多で統合できます。

グループポリシーの使用例としては、AppStream 2.0 フリートごとに異なる Internet Explorer ホームページをイメージセットに指定することが挙げられます。

イメージの更新

ソフトウェアパッチは、コンピューティングリソースのセキュリティとパフォーマンスにとって重要です。Well-Architected フレームワークのセキュリティの柱には、ベストプラクティスとして頻繁なパッチの適用が挙げられています。

イメージを構築してデプロイするとき、AppStream 2.0 イメージにパッチを適用する必要のあるソフトウェアには 4 つのカテゴリがあります。

- アプリケーションと依存関係 — イメージ内のアプリケーションと依存関係にパッチを適用する必要があります。
- Microsoft Windows オペレーティングシステム — Windows の更新をインストールおよび維持する必要があります。
- ソフトウェアコンポーネント — AppStream 2.0 の運用に必要なドライバー、エージェント、その他のソフトウェア (Amazon CloudWatch エージェントなど) です。AppStream 2.0 では、新しいエージェントとドライバーを含む新しいベースイメージが定期的にリリースされます。最新のベースを使用してイメージを再構築し、イメージに含まれるソフトウェアコンポーネントを最新のベースラインに合わせることができます。アプリケーションが多数ある場合や、アプリケーションのインストールが複雑な場合は、最新のベースでイメージを再構築するプロセスには時間がかかり、複雑な作業になる可能性があります。

- AppStream 2.0 エージェント — Image Assistant で [常に最新のエージェントバージョンを使用する] を選択できます。このオプションでは、イメージから起動されるストリーミングインスタンスは、最新バージョンのエージェントを自動的に使用します。

AppStream 2.0 イメージを最新の状態に保つには、次のいずれかの操作を行います。

- マネージド AppStream 2.0 イメージ更新を使用してイメージを更新する - この更新方法では、最新の Windows オペレーティングシステムの更新とドライバーの更新、および最新の AppStream 2.0 エージェントソフトウェアが提供されます。このマネージド方法では、サービスコンポーネントと Microsoft オペレーティングシステムコンポーネントは更新されますが、アプリケーションコンポーネントは更新されません。アプリケーションのインストールが複雑な場合や手動での設定が必要な場合は、この方法を使用するのがベストプラクティスです。
- マネージド AppStream 2.0 イメージバージョンを使用して AppStream 2.0 を更新する - この更新方法では、最新の AppStream 2.0 エージェントソフトウェアが提供されます。この方法では、アプリケーションコンポーネントを更新できます。

フリートのカスタマイズ

フリートタイプ

フリートを作成する場合、カスタマーはフリートタイプを選択する必要があります。フリートタイプごとに、ユーザーエクスペリエンス、コスト、メンテナンスの諸経費の面で異なるメリットがあります。選択したフリートタイプにかかわらず、各オプションは Windows と Linux の両方のプラットフォームタイプ、およびデスクトップビューまたはアプリケーションビューをサポートします。

カスタマーは次のフリートタイプから選択できるようになりました。

- 常時オン — ユーザーがアプリケーションに瞬時にアクセスできるようにします。アプリケーションをストリーミング中のユーザーがいない場合でも、フリート内の実行中のすべてのインスタンスに対して料金が発生します。
- オンデマンド — ストリーミングコストを最適化するには、このフリートタイプを選択します。オンデマンドフリートでは、ユーザーのセッションの開始時間は約 1 ~ 2 分です。ただし、ストリーミングインスタンス料金は、ユーザーが接続しているときにのみ請求され、アプリをストリーミングしていないフリート内の各インスタンスには少額の時間単位の料金がかかります。
- Elastic — Elastic フリートはインストールが不要なアプリケーションに使用でき、仮想ハードディスク (VHD) から実行できます。Elastic フリートは AppStream 2.0 イメージをサポートしておらず、スケーリングポリシーも必要ありません。課金されるのは、ストリーミングセッションの期間中のみです。

表 2 - Amazon AppStream 2.0 のフリートタイプ

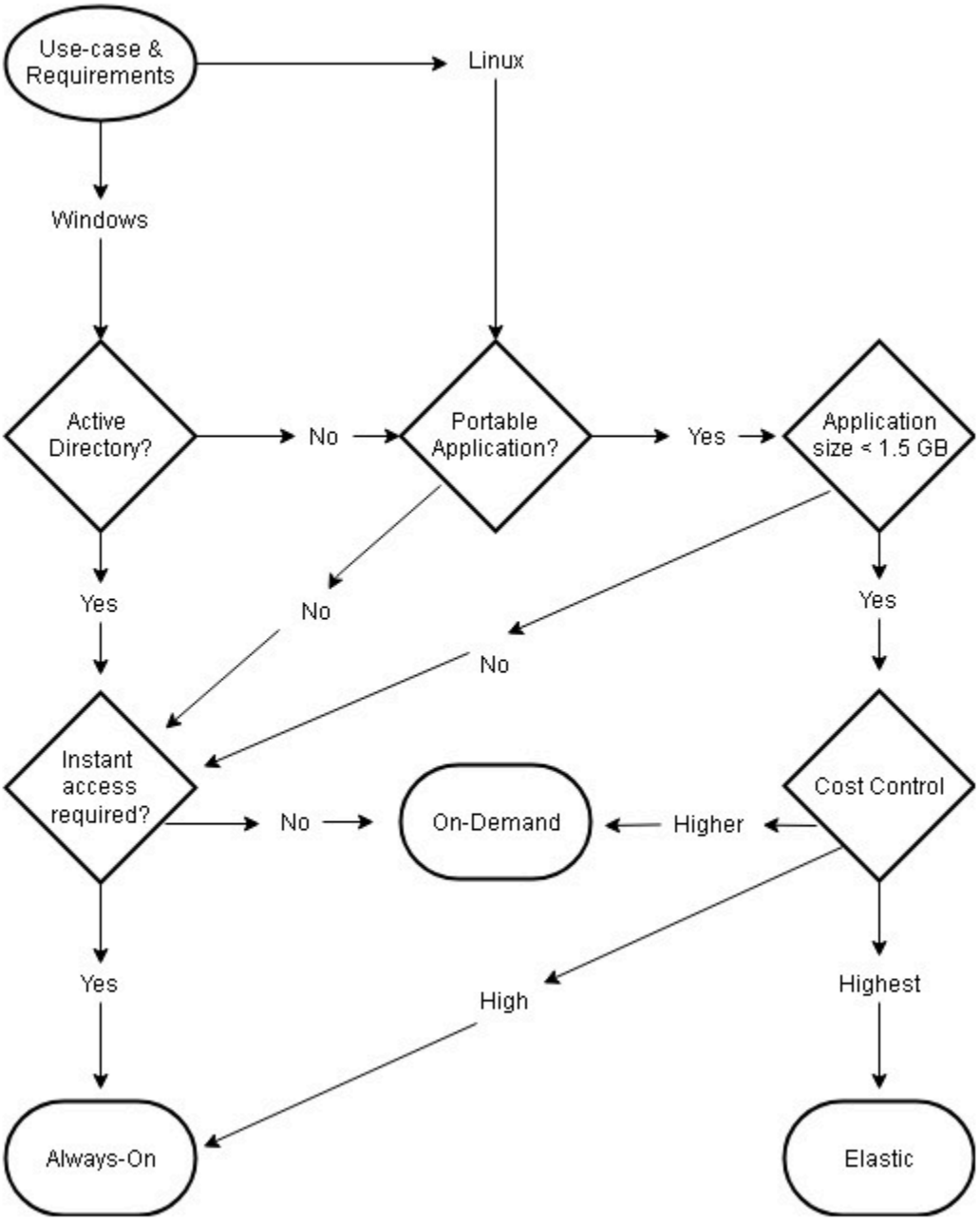
フリートタイプ	どのようなときに使うか	ユーザーエクスペリエンス	料金モデル	メモ
常時オン	ユーザーはセッションを開始するとすぐにアプリケーションにアクセスする必要がある。使用パターンが予	アプリケーションへの即時アクセス	(セッションに使用されているかどうかに関係なく) フリートで使用可能なすべてのインスタンス	カスタムイメージポリシーとスケーリングポリシーをサポートしている。

フリートタイプ	どのようなときに使うか	ユーザーエクスペリエンス	料金モデル	メモ
	測可能で、スケーリングポリシーによってコストを確実に管理できるため、フリートのキャパシティが過剰になることがない。		に対して全額を支払う。	

フリートタイプ	どのようなときに使うか	ユーザーエクスペリエンス	料金モデル	メモ
オンデマンド	フリートには相当な超過キャパシティを維持する必要がある。最もコストが最適化された環境が必要で、未使用のキャパシティに対して全額を支払うのは避けたい。ユーザーは、セッションの開始後、アプリケーションにアクセスできるようになるまで 1 ～ 2 分待つことができる。サイズの大きいインスタンスタイプを使用している。実行中のインスタンスの 1 時間あたりのコストは、停止したインスタンス料金よりもはるかに高くなる。	ユーザーは、セッションを開始してからアプリケーションにアクセスするまで 1 ～ 2 分待機する。	セッションがアクティブなストリーミングインスタンスに対してのみ全額を支払い、アイドル状態のインスタンスには少額の 1 時間あたりの料金を支払う。	カスタムイメージポリシーとスケーリングポリシーをサポートしている。

フリートタイプ	どのようなときに使うか	ユーザーエクスペリエンス	料金モデル	メモ
Elastic	アプリケーションとその依存関係は最大 1.5 GB 未満である。ユーザーが Elastic フリートでセッションを開始するたびに、仮想ハードディスク (VHD) ファイルを Amazon S3 からセッションにダウンロードする必要がある。その結果、VHD ファイルが大きくなる (つまり、サイズが 1.5 GB を超える) と、エンドユーザーエクスペリエンスが低下する。アプリケーションはポータブルである。そのため、アプリケーションとそのすべての依存関係を VHD に配置し、VHD から起動できる。ドメインに参加したストリー	セッションの開始後、ユーザーはアプリケーションにアクセスするまで 45 秒から 3 分待機する (待機時間は仮想ハードディスクのサイズによって異なる)。	課金されるのは、ストリーミングセッションの期間中のみ。Elastic フリートにはアイドル状態のインスタンスという概念がないため、未使用のインスタンスに対しては課金されない。	カスタムイメージ (カスタマーがアプリケーションと共に VHD を提供) やスケーリングポリシーはサポートしていません。現在 `stream.standard.small` 、`stream.standard.medium` インスタンスをサポートしています。ユースケースで別のインスタンスタイプが必要な場合は、AWS アカウントチームにお問い合わせください。

フリートタイプ	どのようなときに使うか	ユーザーエクスペリエンス	料金モデル	メモ
	ミングインスタンスが必要ない (現在、Elastic フリートではドメイン参加はできません)。アクティブなセッションに対してのみ料金を支払いたい (つまり、フリートの未使用のキャパシティについては支払いを行わない)。ユーザーは、セッション開始後、アプリケーションにアクセスするまで 45 秒以上待つことができる。スケーリングは AWS に管理してもらいたい (管理対象のスケーリングポリシーがない)。			



フリートタイプののユースケースと要件

フリートのサイズ設定

最小キャパシティとスケジュールされたスケーリング

AppStream 2.0 フリートの規模を決定する際には、ユーザーエクスペリエンスとコストに直接影響する考慮事項がいくつかあります。[最小キャパシティ] に値を入力することで、AppStream 2.0 インス

タンスの数は、この値を下回ることがほとんどなくなります。AppStream 2.0 セッションが終了した後、AppStream 2.0 インスタンスの合計が [最小キャパシティ] 値を下回ると、新しいフリートインスタンスが開始されます。ここでも重要なのは、1 つの AppStream 2.0 インスタンスは 1 つのユーザーセッションに直接マップされ、これは [最小キャパシティ] 値に直接影響することを覚えておくことです。

[最小キャパシティ] に予想される同時実行数を超える値を入力すると、コストは増加しますが、ユーザーエクスペリエンスには影響しません。値が低すぎる場合、コストは減少しますが、リクエストの合計が利用可能なキャパシティを超えるとユーザーエクスペリエンスに影響します。このような状況では、管理者は「容量の不足不足」エラーが発生することに気付くでしょう。例えば、1 日の開始時に予想される接続数が予測可能な一定値であれば、`PendingCapacity` が `AvailableCapacity` になるのを待つのは、ユーザーの時間を効率的に使用していません。

通常のオフピーク時間に対応できる最小キャパシティから開始し、スケジュールされたスケーリングポリシーを使用して、勤務日の開始前に最小キャパシティを効果的にリセットします。[最小キャパシティ] をオフピーク時間に戻すために、必ずスケジュールされたスケーリングポリシーをもう 1 つ作成してください。スケーリングポリシーとその実装方法の詳細については、このドキュメントの「フリートの自動スケーリング戦略」セクションを参照してください。

最大キャパシティとサービスクォータ

最大キャパシティには任意の値を設定すれば良いように思うかもしれませんが、適切に予測して設定すれば、リソースの総消費量とコストを最適化できます。入力された値が AWS アカウント の AppStream 2.0 フリートのサービスクォータよりも大きい値を入力しても有効であるように思えますが、自動スケーリングイベントがリソースを最大キャパシティにスケーリングしようと試みると、最大キャパシティの値が利用可能なサービスクォータを超えているため、起動に失敗します。組織の予想どおりに自動スケーリングが機能するようには、希望する最大キャパシティでサービスクォータをリクエストしてください。

[最大キャパシティ] の値を設定する際のもう 1 つの重要な考慮事項は、コストです。詳細については、本書の「フリートタイプの選択によるコストの最適化」セクションを参照してください。

デスクトップビューまたはアプリケーションビューの選択

アプリケーションビューとデスクトップビューのどちらを選択しても、パフォーマンスやコストには影響しません。AppStream 2.0 フリート 1 つにつき、一度にアクセスできるビューは 1 つだけです。ストリームビューオプションは変更できます。ストリームビューを変更するにはフリートの再起動が必要なため、この変更はピーク以外の営業時間帯に計画してください。

ストリームビューのベストプラクティスは 1 つではありません。ストリームビューオプションの影響は、次のようにまとめられています。

- 管理者向けの使用状況レポート機能によるアプリケーション使用状況の詳細レポート
- エンドユーザー向けの全体的なエクスペリエンスとワークフロー (例えば、全画面表示のデスクトップはユースケースのニーズに対応しているのか、それともアプリケーションを見るだけで十分なのか、など)。

デスクトップビュー

ユーザーのワークフローがすべてセッションで実行されるユースケースでは、デスクトップビューではすべてのアプリケーションを 1 つの環境に集中させることができるため、ユーザーエクスペリエンスが簡素化されます。デスクトップビューを使用すると、オペレーティングシステム (OS) との統合を必要とするアプリケーションを 3 ~ 5 個以上でプロイしても、より一貫したユーザーエクスペリエンスを実現できます。デスクトップビューは、2 つの異なる環境を管理する場合に効果的です。例えば、ユーザーは本番環境と本番前のデスクトップ環境の両方に同時にアクセスして、レイアウト、構成、およびアプリケーションアクセスの変更を検証できます。

AppStream 2.0 使用状況レポートは、デスクトップビューの毎日のアプリケーションレポートを作成します。生成されるアプリケーションの出力は単に「デスクトップ」で、AppStream 2.0 セッションに直接マッピングされます。詳細については、このドキュメントの「ユーザー使用状況のモニタリング」セクションを参照してください。

アプリケーション専用ビュー

アプリケーション専用ビューは、AppStream 2.0 スタックが断続的に必要とされるいくつかのアプリケーションを提供することを目的としている場合にも効果的です。キオスク環境では、安全にロックダウンされたアプリケーションのデリバリーがアプリケーションビューを通じて配信されます。アプリケーションビューでは、AppStream 2.0 はデフォルトの Windows シェルをカスタムシェルに置き

換えます。このカスタムシェルは実行中のアプリケーションのみを表示し、OS のアタックサーフェス領域を最小限に抑えます。

AppStream 2.0 を使用して既存の組織のデスクトップ環境を強化するユースケースでは、アプリケーション専用ビューが推奨されます。AppStream 2.0 Windows Client をネイティブアプリケーションモードでデプロイし、キーボードショートカットをフルに使用できるようにすることでユーザーの混乱を最小限に抑えます。

AppStream 2.0 使用状況レポートは、アプリケーションビューの毎日のアプリケーションレポートを作成します。アプリケーションと実行の使用状況をより詳細に報告するには、オペレーティングシステムレベルで報告するサードパーティのソリューションを検討してください。Microsoft AppLocker をレポートモードで使用するか、または Liquidware の Stratusphere UX など、AWS Marketplace で使用できるソリューションを検討することもできます。

AWS Identity and Access Management ロールの設定

ワークロードで AppStream 2.0 のエンドユーザーがセッション内から他の AWS サービスにアクセスする必要がある場合、AWS Identity and Access Management(IAM) ロールを使用してアクセスを委任するのがベストプラクティスです。IAM ロールは、フリートレベルでの割り当てを通じてエンドユーザーのセッションに直接アタッチできます。AppStream 2.0 で IAM ロールを使用する際のその他のベストプラクティスについては、管理者ガイドのこのセクションを参照してください。

静的認証情報の使用

ワークロードによっては、IAM アクセスキーをアタッチされたロールから継承するのではなく、静的な入力が必要になる場合があります。これらの認証情報を受け取るには 2 つの方法があります。1 つ目の方法は、アクセスキーを AWS サービス内に保存し、そのサービスから特定の値を取得するための明示的な IAM アクセスをエンドユーザーに付与します。アクセスキーの保存メカニズムの 2 つの例では、AWS Secrets Manager または AWS SSM パラメータストアを使用します。2 つ目の方法は、AppStream 2.0 認証情報プロバイダーを使用して、アタッチされたロールのアクセスキーにアクセスすることです。これを行うには、認証情報プロバイダーを呼び出し、アクセスキーとシークレットキーの出力を解析します。PowerShell 内でこのアクションを実行する方法の例を以下に示します。

```
$CMD = 'C:\Program Files\Amazon\Photon\PhotonRoleCredentialProvider
\PhotonRoleCredentialProvider.exe'
$role = 'Machine'

$output = & $CMD --role=$role
```

```
$parsed = $output | ConvertFrom-Json

$access_key = $parsed.AccessKeyId
$secret_key = $parsed.SecretAccessKey
$session_token = $parsed.SessionToken
```

AppStream 2.0 S3 バケットの保護

AppStream 2.0 ワークロードにホームフォルダまたはアプリケーション永続化が設定されている場合は、永続データが保存されている Amazon S3 バケットを不正アクセスや誤った削除から保護するのがベストプラクティスです。最初の保護レイヤーは、バケットが誤って削除されないように Amazon S3 バケットポリシーを追加することです。2 つ目の保護レイヤーは、最小権限の原則に沿ったバケットポリシーを追加することです。この原則に従うには、必要な関係者にのみバケットアクセスを許可する必要があります。

フリートの自動スケーリング戦略

AppStream 2.0 インスタンスを理解する

AppStream 2.0 フリートインスタンスは、ユーザーとフリートインスタンスの比率が 1:1 です。つまり、各ユーザーは独自のストリーミングインスタンスを持っているということです。同時に接続するユーザーの数によって、フリートのサイズが決まります。

スケーリングポリシー

AppStream 2.0 フリートは、Application Auto Scaling グループで起動されます。これにより、使用状況に基づいてフリートをスケーリングして需要を満たすことができます。使用量が増えるとフリートはスケールアウトし、ユーザーが接続を解除するとフリートはスケールインします。これはスケーリングポリシーを設定することで制御されます。スケジュールされたスケーリング、ステップスケーリングポリシー、およびターゲット追跡スケーリングポリシーを設定できます。これらのスケーリングポリシーの詳細については、「Amazon AppStream 2.0 の Fleet Auto Scaling」を参照してください。

ステップスケーリング

これらのポリシーは、フリートのキャパシティを現在のフリートサイズまたは特定のインスタンス数に対するパーセンテージで増減します。ステップスケーリングポリシーは、`Capacity Utilization`、`Available Capacity`、または `Insufficient Capacity Errors` の AppStream 2.0 CloudWatch メトリクスによってトリガーされます。

ステップスケーリングポリシーを使用する場合、AWS では、固定数のインスタンスではなく、一定の割合のキャパシティを追加することを推奨します。これにより、スケーリングアクションがフリートの規模に比例するようになります。これにより、(フリートのサイズに対して追加したインスタンスの数が少ないことで) スケールアウトが遅すぎたり、フリートが小さいときにインスタンス数が多すぎる状況を回避できます。

ターゲット追跡

このポリシーでは、フリートに対するキャパシティの使用レベルを指定します。Application Auto Scaling は、スケーリングポリシーをトリガーする CloudWatch アラームを作成および管理します。これにより、指定されたターゲット値、またはそれに近い値にフリートを維持するためにキャパシ

ティを追加または削除します。アプリケーションの可用性を高めるために、フリートのスケールアウトはメトリクスに比例して可能な限り高速に行われますが、スケールインはより緩やかです。ターゲット追跡を設定するときは、スケールアウトとスケールインが希望の間隔で行われるようにスケーリングのクールダウンを考慮してください。

ターゲット追跡は、チャーン率が高い状況に効果的です。チャーンは、多数のユーザーが短期間にセッションを開始し、終了したときに発生します。チャーンは、フリートの CloudWatch メトリクスを調べることで特定できます。フリートの保留中のキャパシティが 0 以外で、希望するキャパシティに変更がない (またはほとんど変更がない) 期間は、高いチャーンが発生している可能性が高いことを示しています。チャーン率が高い状況では、(100 — ターゲット使用率) が 15 分間のチャーン率を上回るターゲット追跡ポリシーを設定します。例えば、ユーザーのターンオーバーによって 15 分以内にフリートの 10% が終了する場合は、高いチャーンを相殺するためにキャパシティのターゲット使用率を 90% 以下に設定します。

スケジュールベースのスケーリング

これらのポリシーにより、時間ベースのスケジュールに基づいて希望するフリートキャパシティを設定できます。このポリシーは、ログインの動作を理解し、需要の変化を予測できる場合に有効です。

例えば、勤務日の最初に、100 名のユーザーが 午前 9 時にストリーミング接続をリクエストすることが予期されるとします。この場合、スケジュールベースのスケーリングポリシーを設定して、午前 8 時 40 分に最小フリートサイズを 100 に設定できます。これにより、フリートインスタンスを作成して勤務日の開始時に使用可能になることで、100 人のユーザーが同時に接続できます。その後、午後 5 時にフリートを少なくとも 10 人にスケーリングするようにスケジュールされた別のポリシーを設定できます。これにより、勤務時間外のセッションの需要が勤務時間中よりも少なくなるため、コストを節約できます。

本番環境におけるスケーリングポリシー

さまざまな種類のスケーリングポリシーを 1 つのフリートにまとめることで、ユーザーの行動に合わせた正確なスケーリングポリシーを定義できます。前の例では、スケジュールされたスケーリングポリシーをターゲット追跡またはステップスケーリングポリシーと組み合わせて、特定のレベルの使用率を維持できます。スケジュールされたスケーリングとターゲット追跡スケーリングの組み合わせにより、キャパシティがすぐに必要になったときに、使用率レベルの急激な増加による影響を軽減できます。

スケーリングポリシーによって必要なインスタンス数が変更されても、ストリーミングセッションに接続しているユーザーは、スケールインやスケールアウトの影響を受けません。スケーリングポリシーによって既存のストリーミングセッションが終了することはありません。既存のセッションは、

ユーザーまたはフリートタイムアウトポリシーによってセッションが終了されるまで、中断されずに継続されます。

CloudWatch メトリクスで AppStream 2.0 の使用状況をモニタリングすると、スケーリングポリシーを時間の経過とともに最適化するのに役立ちます。例えば、初期設定時にリソースを過剰にプロビジョニングすることは一般的で、使用率が低い状態が長期間続くことがあります。あるいは、フリートのプロビジョニングが不足していると、キャパシティ使用率が高くなり、「容量の不足」というエラーが表示されることがあります。CloudWatch メトリクスを確認すると、スケーリングポリシーを調整してこれらのエラーを軽減するのに役立ちます。詳細および使用できる AppStream 2.0 スケーリングポリシーの例については、「Amazon AppStream 2.0 フリートをスケーリングする」を参照してください。

スケーリングポリシー設計のベストプラクティス

スケーリングポリシーを組み合わせる

多くのカスタマーは、AppStream 2.0 の自動スケーリングの能力と柔軟性を高めるために、さまざまな種類のスケーリングポリシーを 1 つのフリートにまとめることを選択しています。例えば、スケジュールされたスケーリングポリシーを設定し、ユーザーが仕事を始めるのを見越して午前 6 時にフリートの最小値を増やし、ユーザーが仕事を停止する前の午後 4 時にフリートの最小値を減らすようにできます。このスケジュールされたスケーリングポリシーをターゲット追跡ポリシーまたはステップスケーリングポリシーと組み合わせて特定の使用率を維持し、使用量が急増した場合は日中にスケールインまたはスケールアウトすることができます。スケジュールされたスケーリングとターゲット追跡スケーリングを組み合わせることで、急に容量が必要になったときに、使用率レベルの急激な増加による影響を軽減できます。

スケーリングチャーンを回避する

ユースケースによってフリートのチャーン率が高くなる可能性があるかどうかを検討してください。チャーンは、多数のユーザーが短期間にセッションを開始し、その後終了したときに発生します。これは、多数のユーザーがサインオフする前に、フリート内のアプリケーションにわずか数分間に同時アクセスした場合に発生する可能性があります。

このような状況では、ユーザーがセッションを終了するとインスタンスも終了するため、フリートのサイズが希望するキャパシティをはるかに下回る可能性があります。ステップスケーリングポリシーでは、チャーンを相殺するほど迅速にインスタンスを追加できず、その結果、フリートが一定のサイズにとどまってしまうことがあります。

チャーンは、フリートの CloudWatch メトリクスを調べることで特定できます。フリートの保留中のキャパシティが 0 以外で、希望するキャパシティに変更がない (またはほとんど変更がない) 期間は、高いチャーンが発生している可能性が高いことを示しています。チャーン率が高い状況を考慮に入れるには、ターゲット追跡スケーリングポリシーを使用し、(100 — ターゲット使用率) が 15 分間のチャーン率を上回るようにたターゲット使用率を選択します。例えば、ユーザーの解約によって 15 分以内にフリートの 10% が終了する場合は、高いチャーンを相殺するためにキャパシティのターゲット使用率を 90% 以下に設定します。

最大プロビジョニング率を理解する

多数のユーザーを対象に AppStream 2.0 フリートを管理しているカスタマーは、プロビジョニング率の制限を検討する必要があります。この制限は、インスタンスを 1 つのフリートに、または AWS アカウント 内のフリートすべてに追加できる速度に影響します。

考慮すべき制限は 2 つあります。

- 単一のフリートの場合、AppStream 2.0 は 1 分あたり最大 20 インスタンスのレートでプロビジョニングします。
- 単一の AWS アカウント の場合、AppStream 2.0 は 1 分あたり 60 インスタンス (バーストは 1 分あたり 100 インスタンス) のレートでプロビジョニングします。

3 つ以上のフリートを並行してスケールアップする場合、アカウントのプロビジョニング率の制限はこれらのフリートで共有されます (例えば、6 つのフリートを並行してスケーリングすると、それぞれ 1 分あたり最大 10 インスタンスをプロビジョニングできます)。また、特定のストリーミングインスタンスがスケーリングイベントに応じてプロビジョニングを完了するまでの時間を考慮します。Active Directory ドメインに参加していないフリートの場合、通常 15 分です。Active Directory ドメインに参加しているフリートの場合、これには 25 分ほどかかることがあります。

これらの制限を考慮し、次の例について検討してみましょう。

- 1 つのフリートを 0 インスタンスから 1000 インスタンスにスケーリングする場合、プロビジョニングが完了するまでに 50 分 (1000 インスタンス/1 分あたり 20 インスタンス) かかり、エンドユーザーがすべてのインスタンスを使用できるようになるまでにさらに 15 ~ 25 分、合計 65 ~ 75 分かかります。
- 3 つのフリートを 0 インスタンスから 333 インスタンスに同時にスケーリングする場合、すべてのフリートがプロビジョニングを完了するまでに 17 分 (999 インスタンス/1 分あたり 60 インスタンス) かかり、エンドユーザーがこれらすべてのインスタンスを使用できるようになるまでにさらに 15 分、合計 32 ~ 42 分かかります。

複数のアベイラビリティーゾーンを使用する

フリートをデプロイするリージョン内の AZ を複数選択します。フリートに複数の AZ を選択すると、フリートはスケーリングイベントに対応してインスタンスを追加できる可能性が高くなります。CloudWatch メトリクス PendingCapacity は、大規模なフリートデプロイにおいてフリート AZ の設計がどの程度最適化されているかを評価するため、最初にチェックします。PendingCapacity の

値が持続的に高い場合、水平スケーリングを (AZ 全体に) 拡張する必要性を示している場合があります。詳細については、「Amazon AppStream 2.0 リソースのモニタリング」を参照してください。

例えば、自動スケーリングがインスタンスをプロビジョニングしてフリートのサイズを増やそうとしたときに、選択した AZ のキャパシティが不足している場合、自動スケーリングは代わりにフリートに指定した他の AZ にインスタンスを追加します。アベイラビリティーゾーンと AppStream 2.0 設計の詳細については、本ドキュメントの「アベイラビリティーゾーン」を参照してください。

キャパシティ不足エラーのメトリクスをモニタリングする

「キャパシティ不足エラー」は AppStream 2.0 フリートの CloudWatch メトリクスです。このメトリクスは、キャパシティ不足により拒否されたセッションリクエストの数を指定します。

スケーリングポリシーを変更する場合、キャパシティ不足エラーが発生したときに通知する CloudWatch アラームを作成すると便利です。これにより、スケーリングポリシーをすばやく調整してユーザーの可用性を最適化できます。管理ガイドには、AppStream 2.0 リソースをモニタリングする詳細な手順が記載されています。

接続方法

AppStream 2.0 でセッションをストリーミングする場合、ユーザーは次の 2 つの接続方法を使用できます。

- ウェブブラウザアクセス — あらゆる HTML5 対応ブラウザがサポートされます。プラグインやダウンロードは不要です。
- AppStream 2.0 Windows クライアント

ベストプラクティスとして、ユーザーのユースケースの機能とデバイスの要件を検討し、どのブラウザまたはデバイスがユーザーの要件を最もよくサポートするかについて検討します。

> **Note**
>
> AppStream 2.0 は、画面解像度が 1,024 x 768 ピクセル未満のデバイスではサポートされません。

サマリー機能とデバイスのサポート

表 3 —サマリー機能とデバイスのサポート

	ウェブブラウザアクセス	AppStream 2.0 Windows クライアント
マルチモニター (最大 2K 解像度)	サポート対象	サポート対象
マルチモニター (最大 4K 解像度)	該当なし	サポート対象
ドローイングタブレットのサポート	サポート対象*	サポート対象
タッチスクリーンデバイスのサポート	サポート対象	該当なし

	ウェブブラウザアクセス	AppStream 2.0 Windows クライアント
USB パススルーデバイスのサポート	該当なし	サポート対象
キーボードショートカット	サポート対象	サポート対象
相対マウスオフセット	サポート対象	サポート対象
ファイル転送	サポート対象	サポート対象
ローカルプリンターのリダイレクト	該当なし	サポート対象
ローカルドライブのリダイレクト	該当なし	サポート対象
ウェブカメラのサポート	サポート対象	サポート対象

*Google Chrome と Mozilla Firefox のみ

ウェブブラウザアクセス

AppStream 2.0 のウェブブラウザアクセスでは、専用クライアントをインストールしなくてもアプリケーションにアクセスできます。ユーザーは、サポートされている HTML5 対応ブラウザを使用して接続できます。ブラウザのプラグインや拡張機能は必要ありません。

ウェブブラウザへのアクセスにより、エンドデバイスのオペレーティングシステムとタイプを幅広く選択できるようになります。

AppStream 2.0 の Windows 用クライアント

AppStream 2.0 の Windows 用クライアントは、Windows PC にインストールするアプリケーションです。このアプリケーションには、ウェブブラウザを使用して AppStream 2.0 にアクセスした場合には利用できない追加の機能があります。例えば、AppStream クライアントでは以下を実行できます。

- 2 台以上のモニターまたは 4K 解像度を使用する

- USB デバイスで AppStream 2.0 を介してストリーミングされるアプリケーションを使用する
- ストリーミングセッション中にローカルドライブとフォルダにアクセスする
- プリントジョブをストリーミングアプリケーションから、ローカルコンピュータに接続されているプリンターにリダイレクトする
- ストリーミングセッション内のビデオ会議や音声会議に、ローカルのウェブカメラを使用する
- ストリーミングセッション中にアクセスするアプリケーションでキーボードショートカットを使用する
- ローカルにインストールされたアプリケーションを操作するのとほぼ同じ方法で、リモートストリーミングアプリケーションを操作する

AppStream 2.0 クライアント接続モード

AppStream 2.0 クライアントには、[ネイティブアプリケーションモード] と [クラシックモード] の 2 つの接続モードがあります。選択した接続モードによって、アプリケーションのストリーミング中に使用できるオプション、およびストリーミングアプリケーションの機能と表示方法が決まります。管理者は、ネイティブアプリケーションモードとクラシックモードを切り替えるユーザーの機能を制御できます。

- クラシックモードでは、AppStream 2.0 セッションウィンドウでアプリケーションをストリーミングします。これは、エンドユーザーがウェブブラウザでアプリケーションをストリーミングする方法と似ています。エンドユーザーがローカルファイル接続やプリンターリダイレクトなどの追加機能を利用する一方で、ブラウザと同じ方法でアプリケーションをストリーミングしたい場合は、クラシックモードを使用します。クラシックモードはデフォルトの接続モードとして推奨されます。デスクトップビューでサポートされているモードはクラシックモードだけです。
- ネイティブアプリケーションモードでは、エンドユーザーはローカルにインストールされた他のアプリケーションと同様にリモートストリーミングアプリケーションを操作できます。エンドユーザーがローカルにインストールされたアプリケーションを操作することに慣れている場合は、ネイティブアプリケーションモードを使用するとシームレスな操作が可能になります。リモートストリーミングアプリケーションは、ローカルにインストールされたアプリケーションとほぼ同じように機能します。リモートストリーミングアプリケーションのアイコンはローカルアプリケーションのアイコンと同じように、ローカル PC のタスクバーに表示されます。ローカルアプリケーションのアイコンとは異なり、ネイティブアプリケーションモードのストリーミングアプリケーションのアイコンには AppStream 2.0 ロゴが含まれます。ネイティブアプリケーションモードは、ユーザーがアプリケーションのキーボードショートカットを使用し、キーボードショートカットで個々のローカルアプリケーションと個別のリモートアプリケーションを簡単に切り替えたい場合に推奨される接続モードです。

クライアントのデプロイと管理

ユーザーは AppStream 2.0 クライアントをインストールするか、PowerShell スクリプトをリモートで実行して AppStream 2.0 クライアントをインストールするか、カスタマイズされた設定で AppStream 2.0 クライアントをインストールできます。

ユーザーがストリーミングセッションで使用できるようにする USB デバイスを認定する必要があります。その USB デバイスが認定されていない場合、それは AppStream 2.0 によって検出されず、セッションと共有することはできません。デバイスが認証された後、ユーザーは新しいストリーミングセッションを開始するたびにデバイスを AppStream 2.0 と共有する必要があります。

AppStream 2.0 クライアントを大規模にデプロイする場合は、AWS では、エンタープライズデプロイツールの使用を推奨します。エンタープライズデプロイツールには、AppStream クライアントインストールファイルとグループポリシー管理用テンプレートが含まれています。

カスタムドメイン

AppStream 2.0 をプログラム的にデプロイする場合、ストリーミングセッションの使い慣れたエクスペリエンスをユーザーに提供できるカスタムドメインを作成できます。AppStream 2.0 の SAML 2.0 IdP デプロイでは、ユーザーアクセスは AppStream 2.0 ではなく IdP から始まることを強調することが重要です。AppStream 2.0 の URL は認証後に IdP によって提供されるため、ユーザーには必要ありません。したがって、SAML 2.0 IdP のデプロイにはカスタムドメイン名は必要ありません。

認証

AppStream 2.0 では、認証は Amazon AppStream 2.0 の外部で、または AppStream 2.0 サービスの一部として行うことができます。 AppStream 2.0 デプロイでの認証方法の選択は、設計の基本的な考慮事項です。組織がユースケース別に AppStream 2.0 の複数のデプロイを持つことは珍しくありません。ユースケースごとに認証方法が異なる場合があります。

AppStream 2.0 の認証方法には 3 つのタイプがあります。

- SAML 2.0
- ユーザープール
- プログラミング

最適化された方法の決定

Amazon AppStream 2.0 は、ほとんどの組織設計要件に柔軟に対応できるように設計されています。最適な認証方法を決定する際には、サービスを利用する人々の目標と目的、および組織のポリシーと手順を考慮することがベストプラクティスです。

ユースケースと組織の目標を組み合わせた例をいくつか紹介します。

表 4 — 組織的な目標のあるユースケース

例	説明	Authentication
ドメインに参加したフリートインスタンスが必要	AppStream イメージにインストールされているアプリケーションは、ドメインに参加しているリソースにのみアクセスできます。	SAML 2.0
Microsoft サービスとの緊密な統合	Microsoft のグループポリシーとバックエンドインフラストラクチャの開発に対して組織が依存している	SAML 2.0
既存のエンタープライズシングルサインオン (SSO)	新しいサービスはすべて、複数の報告プロセスとセキュリ	SAML 2.0

例	説明	Authentication
	ティプロセスが確立されたエンタープライズ SSO ソリューションを活用する必要がある。	
スマートカードによるアプリケーションのサポート	スマートカードリーダーを介してストリーミングされるアプリケーションのセッション内認証されるスマートカード (プライベート ID 検証や共通アクセスカードなど)。	SAML 2.0
臨時スタッフによる一時的な人材	派遣社員には、1 年のうち数か月間、業務を遂行するための社内リソースを含まない少数のアプリケーションが割り当てられる。	ユーザープール
サポートの制限	ユーザー数が 50 人未満で、IT スタッフが限られている小規模な組織で、ID プロバイダー (IdP) の維持にかかる諸経費の削減を検討している	ユーザープール
独立系ソフトウェアベンダー (ISV)	ユーザーの使用権限と認証を含む組織によって構築され、ソリューションの一部として AppStream 2.0 を拡張する独自のソリューション。*	プログラミング
テクノロジーの紹介	ユーザー情報を保存する必要がなく、ソリューションのガイド付きツアーの一環として独自のテクノロジーを紹介する、完全に一時的な環境。	プログラミング

例	説明	Authentication
インタラクティブな Web サイトエクスペリエンス	ストリーミング中の Windows アプリケーションで Web サイトをインタラクティブにする。**	プログラミング

*詳細については、「ソフトウェアベンダー: アプリケーションを任意のデバイスで使用する」を参照してください。

**詳細については、「埋め込み AppStream 2.0 ストリーミングセッション」を参照してください。

組織内に、前述の例に記載されていないユースケースまたはポリシーがある場合は、認証ソリューションと競合しないように、 AppStream 2.0 ワークフロー消費の望ましい最終状態を予測することがベストプラクティスです。

ID プロバイダーの設定

SAML 2.0

Security Assertion Markup Language (SAML) 2.0 は、ユーザーがAWS リソースを使用できるようにするための一般的な導入オプションです。さまざまなサードパーティー SAML 2.0 ID プロバイダーが 2.0 をサポートしています AppStream。 AppStream 2.0 リソースがドメインに参加しているかどうかにかかわらず、SAML 2.0 IdP では IAM を使用する必要があります。

ほとんどの は、SAML アプリケーションごとに特定の SAML 属性を持つ一意の metadata.xml IdPs を生成するため、 AppStream 2.0 スタックごとに、SAML IdP と信頼関係を持つロールと、SAML IdP の要件と AppStream 2.0 スタックの ARN に一致する条件で appstream:Stream への単一のアクセス許可を持つポリシーが必要です。

AppStream 2.0 管理ガイドには、単一の AppStream 2.0 スタック設計の設定例が記載されています。マルチスタックデプロイの場合は、SAML 2.0 マルチスタックアプリケーションカタログを使用するためのオプション手順を参照してください。

ユーザープール

AppStream 2.0 のユーザープールタブは、小規模な概念実証のための有効なオプションです。ベストプラクティスとして、 AppStream 2.0 を使用して本番アプリケーションを配信するユースケースや組織では、ユーザープールを避けることをお勧めします。

ユーザープールについて注意すべき重要な点の 1 つは、ユーザーの E メールアドレスでは大文字と小文字が区別されるということです。そのため、ユーザー認証情報の正しい入力方法をユーザーに確実に伝えることがベストプラクティスです。

ストリーミング URL

集中型サービス (通常は ISVs) から AppStream 2.0 リソースを呼び出すデプロイの場合、プログラムによる認証はアプリケーションに依存して をプログラムで呼び出し、情報を動的AWSに渡し、ユーザーのために AppStream 2.0 セッションを作成します。CreateStreamingURL オペレーションを使用してストリーミング URLs「プログラム」と呼ばれます) を使用します。CreateStreamingURL 呼び出しを行うユーザーは、appstream:CreateStreamingURL のアクセス許可を持つ有効なユーザーまたはロールを使用している必要があります。

プログラムによるアクセスのポリシーを作成するときは、デフォルトの「*」の代わりにリソースセクションで正確な AppStream 2.0 スタック ARN を指定してアクセスを保護することがベストプラクティスです。例:

Example

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "appstream:createStreamingURL"
            ],
            "Resource": "arn:aws:appstream:us-east-
1:031421429609:stack/BestPracticesStack"
        }
    ]
}
```

> **Note**
>
> describe stacks API または AWS CLI を使用して、 AppStream 2.0 スタックの ARNs をすばやく取得できます。

AppStream 2.0 インスタンスは汎用インスタンスとして起動する必要があります。アプリケーションから渡された情報を通じて、 AppStream 2.0 インスタンスはセッションコンテキストを使用して環境を確立し、ユーザーにとってモノを動的にします。

ローカル GPOs を使用してユーザーのログオン時に設定を指定できますが、セッションコンテキストは、 を使用しCreateStreamingURL、セッションで使用する AppStream顧客 ID やデータベース接続設定などの主要な属性を渡す場合のベストプラクティスです。

アプリケーションの使用権限

AppStream 2.0 は、ユーザーに表示されるアプリケーションカタログを動的に構築できます。アプリケーションの使用権限は、SAML 2.0 属性、または AppStream 2.0 動的アプリケーションフレームワークに基づいています。

ほとんどのシナリオでは、SAML 2.0 を使用する属性ベースのアプリケーションの使用権限が推奨されます。アプリケーションパッケージの配信を管理するには、動的アプリケーションフレームワークが推奨されます。

Microsoft Active Directory との統合

Amazon AppStream 2.0 Image Builder とフリートは Microsoft Active Directory と統合できます。これにより、ユーザーの認証と許可を一元的に行うことができ、ドメインに参加している AppStream 2.0 インスタンスに Active Directory グループポリシーを適用できます。ドメインに参加した AppStream フリートを使用すると、オンプレミス環境と同じ管理上のメリットが得られます。これには、ネットワークファイル共有、ユーザーアプリの使用権限、ローミングプロファイル、プリンターアクセス、およびその他のポリシーベースの設定の一元管理が含まれます。

AppStream 2.0 環境を Active Directory と統合する場合、AppStream 2.0 スタックへの初期認証は引き続き SAML2.0 IdP によって管理されていることに注意することが重要です。ユーザーが IdP に対して正常に認証されると、ユーザーがセッションを起動するときに、Active Directory ドメインのドメインパスワードまたはスマートカード認証を入力する必要があります。

AppStream 2.0 で使用する Active Directory ドメインサービス (ADDS) 環境を設計する場合、2 つのサービスオプションがあり、利用できるさまざまなデプロイシナリオがあります。また、AppStream 2.0 のネットワークについては、必ず Active Directory サイトトポロジの所有者に確認してください。

サービスオプション

Active Directory は、AWSマネージド Microsoft Active Directory (AD) を使用して導入することもできます。AWSマネージド Microsoft AD は、Microsoft Active Directory を実行できるフルマネージドサービスです。Microsoft Active Directory は EC2 またはオンプレミスで実行されているセルフホスト環境でも使用できます。

デプロイシナリオ

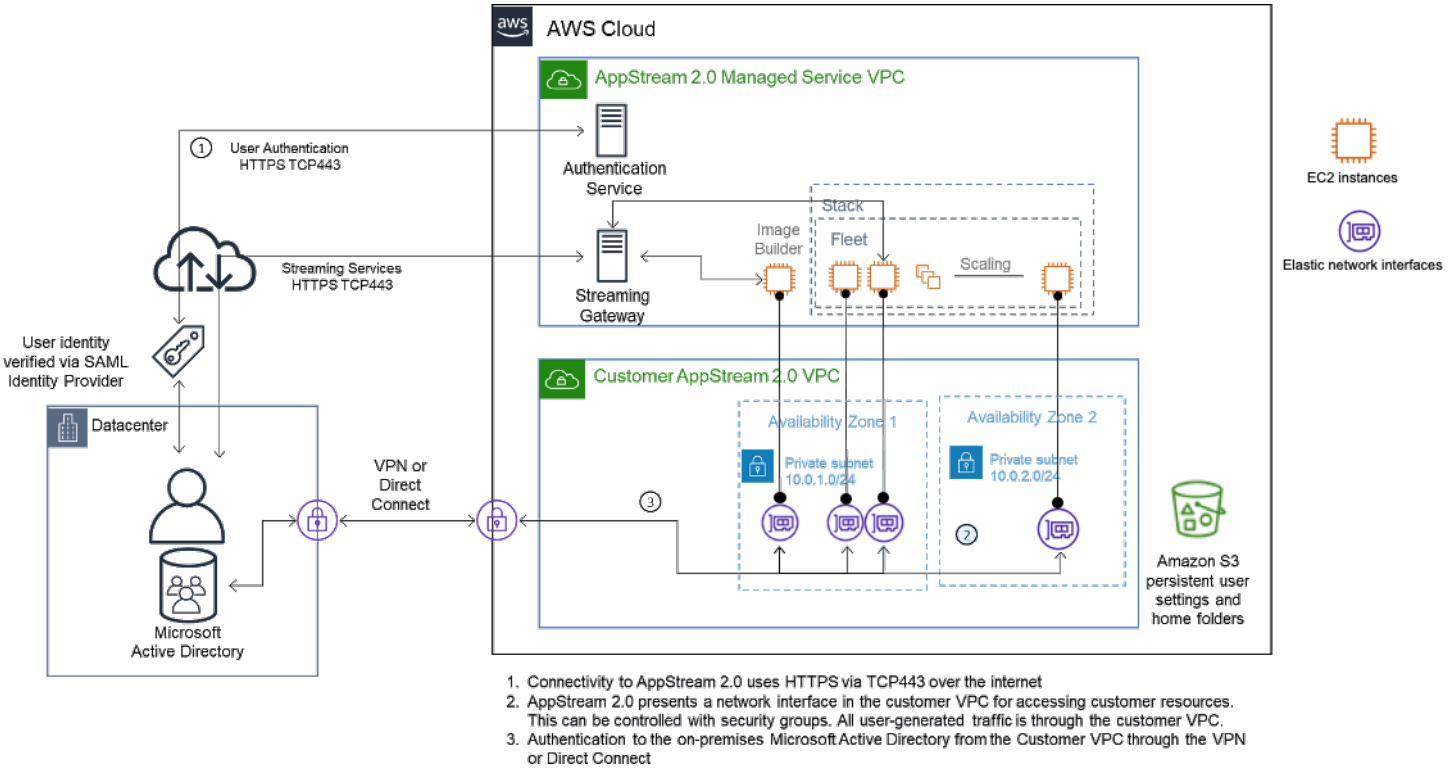
以下のデプロイシナリオは、AppStream 2.0 と Microsoft マネージド AD、またはカスタマーのセルフマネージド Active Directory との統合オプションとして一般的に使用され、推奨されています。以下に示すアーキテクチャ図はすべて、Amazon のコアコンストラクトを使用しています。

- Amazon 仮想プライベートクラウド (VPC) — 4 つの AZ にまたがる少なくとも 4 つのプライベートサブネットを備えた AppStream 2.0 サービス専用の Amazon VPC の作成。プライベートサブネットのうちの 2 つは AppStream フリートとイメージビルダーに使用されます。残りの 2 つのサブネットは、EC2 または Microsoft マネージド (AD) のドメインコントローラーに使用されます。

- Dynamic Host Configuration Protocol (DHCP) オプションセット — VPC にプロビジョニングされる AppStream 2.0 フリートと Image Builder に設定情報を渡すための標準を提供します。DHCP オプションセットは VPC レベルで定義されます。これにより、カスタマーは、プロビジョニング時にインスタンス化される AppStream 2.0 で使用される特定のドメイン名と DNS 設定を定義できます。
- AWS ディレクトリサービス — Amazon Microsoft マネージド AD は、AppStream 2.0 ワークロードと組み合わせて使用される 2 つのプライベートサブネットにデプロイできます。
- AppStream 2.0 フリート — AppStream 2.0 フリートまたは Image Builder は AWS マネージド VPC でホストされます。AppStream 2.0 の各インスタンスには 2 つの Elastic Network Interface (ENI) があります。プライマリインターフェイス (`eth0`) は、管理目的と、ストリーミングゲートウェイ経由のインスタンスへのエンドユーザー接続の仲介に使用されます。セカンダリインターフェイス (`eth1`) はカスタマー VPC に挿入され、カスタム VPC またはオンプレミスの他のリソースにアクセスするために使用できます。

シナリオ 1: オンプレミスにデプロイされた Active Directory ドメインサービス (ADDS)

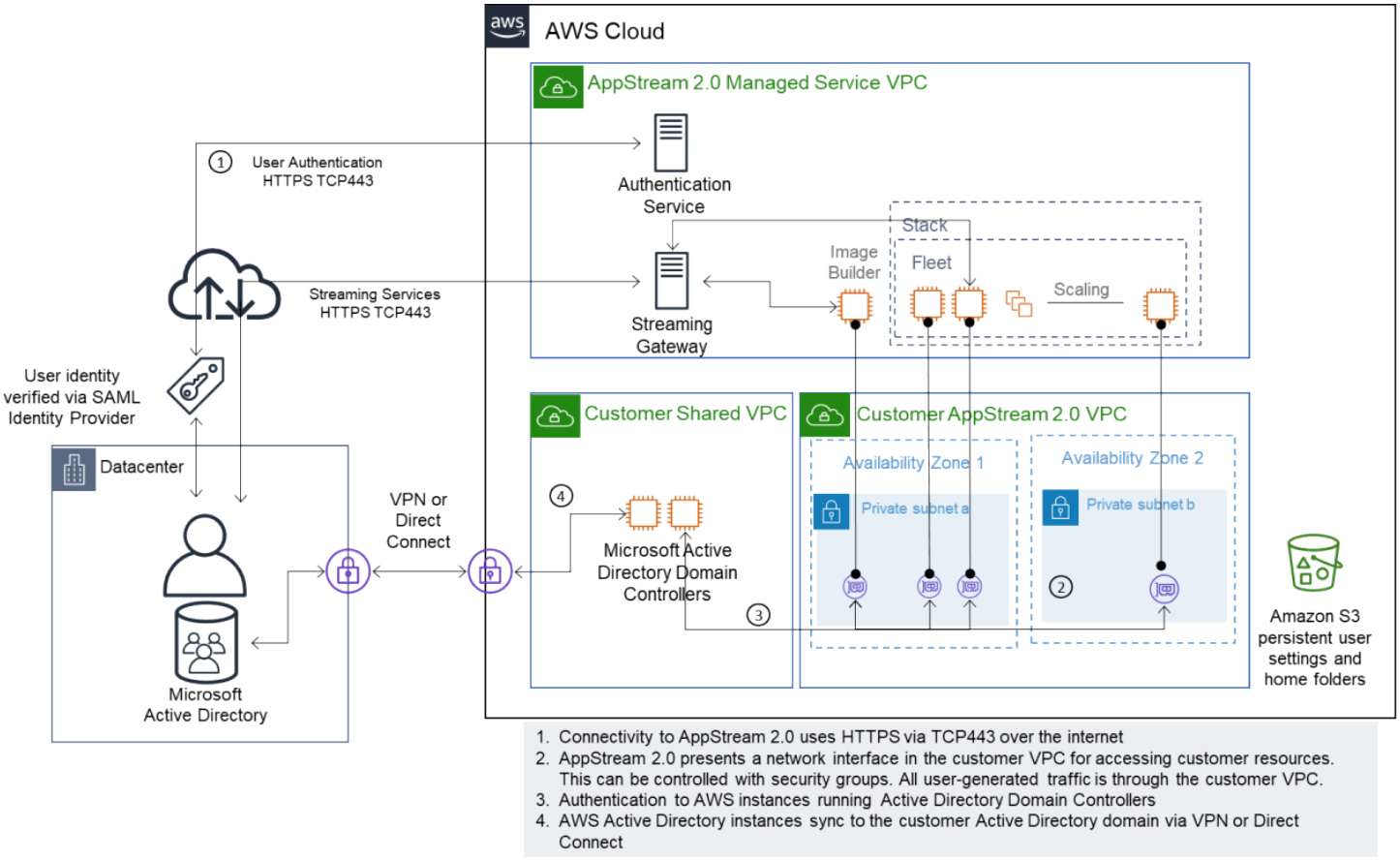
すべての認証トラフィックは、カスタマー VPC からカスタマーゲートウェイまで VPN または Direct Connect 接続を通過します。このシナリオのメリットは、カスタマー VPC に追加のドメインコントローラーをプロビジョニングしなくても、すでにデプロイされている可能性のある AD 環境を使用できることです。一方、デメリットは、AppStream 2.0 フリートのユーザーの認証と許可が VPN または Direct Connect のみに依存していることです。ネットワーク接続に問題があると、AppStream 2.0 フリートまたは Image Builder に直接影響が発生します。デュアル VPN トンネルまたは異なるパスの Direct Connect 接続を提供することで、この潜在的なリスクを軽減できます。



シナリオ 1 — オンプレミスでデプロイされた Active Directory ドメインサービス (ADDS)

シナリオ 2: Active Directory ドメインサービス (ADDS) AWS カスタマー VPC に拡張する

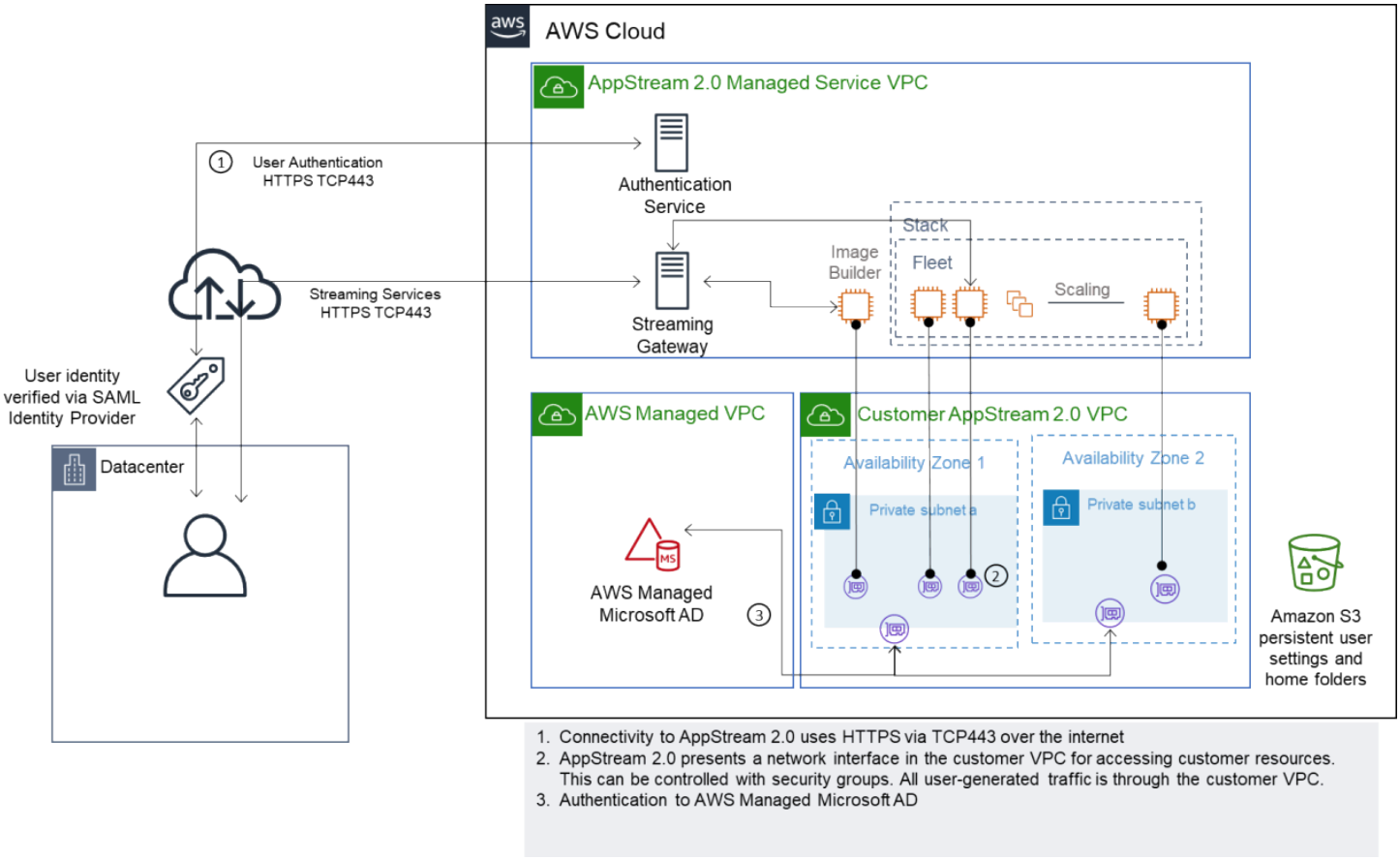
Active Directory はカスタマー VPC まで拡張されます。カスタマー VPC の新しいドメインコントローラーに対して Active Directory サイトを作成する必要があります。認証トラフィックは、VPN または Direct Connect 接続を経由する代わりに、AWS カスタマー VPC のドメインコントローラーにルーティングされます。



シナリオ 2 - Active Domain サービスの AWS カスタマー仮想プライベートクラウドへの拡張

シナリオ 3: AWS マネージド Microsoft Active Directory

AWS マネージド Microsoft AD は AWS クラウド にデプロイされ、AppStream 2.0 フリートと Image Builder のアイデンティティおよびリソースドメインとして使用されます。



シナリオ 3 — AWS マネージド Active Directory

Active Directory サービスサイトトポロジ

Active Directory サービスサイトトポロジは、物理ネットワークを論理的に表現したものです。

サイトトポロジは、クライアントクエリと Active Directory レプリケーショントラフィックを効率的にルーティングするのに役立ちます。サイトトポロジを適切に設計および管理することで、組織は次のメリットが得られます。

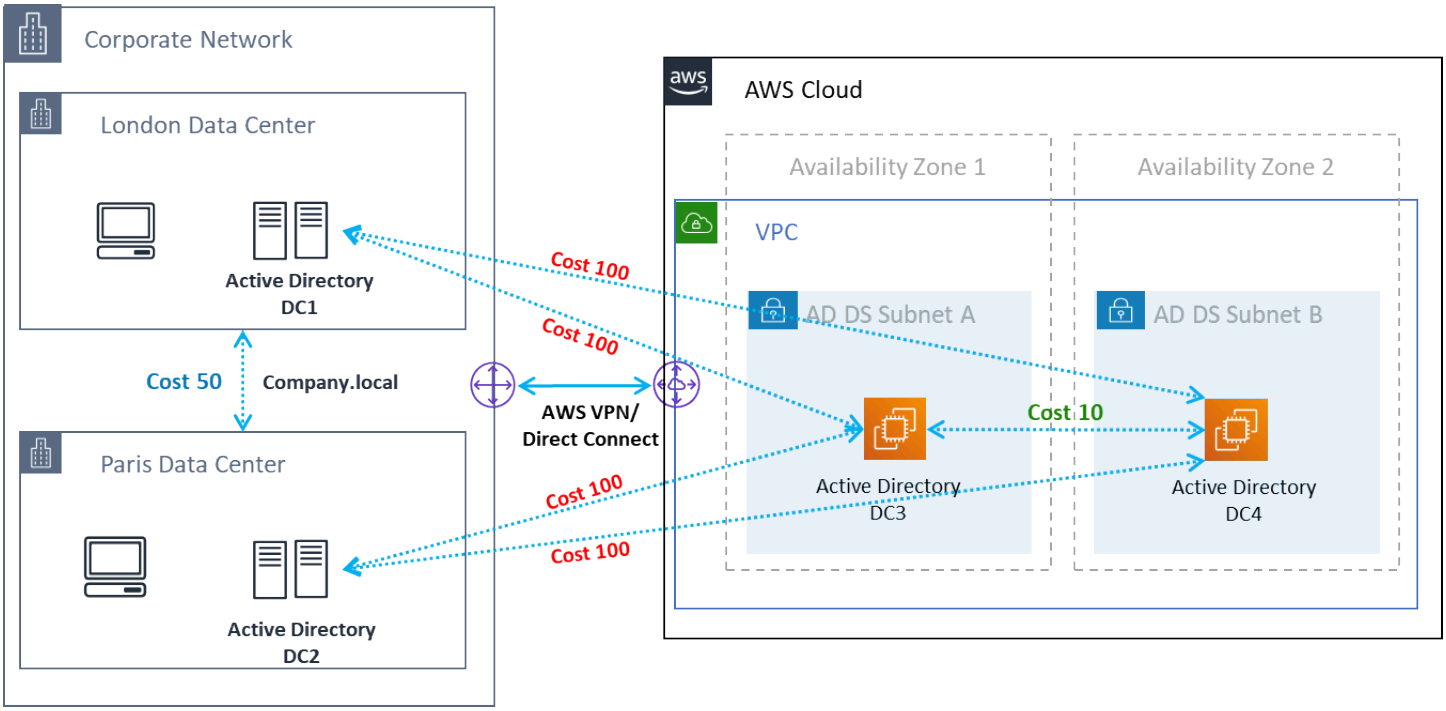
- オンプレミスと AWS クラウド を同期するときに Active Directory データを複製するコストを最小限に抑えます。
- クライアントコンピューターがドメインコントローラーなどの最も近いリソースを検索する機能を最適化します。これにより、低速のワイドエリアネットワーク (WAN) リンクのネットワークトラフィックを減らし、ログオンとログオフのプロセスを改善し、リソースへのアクセスオペレーションをスピードアップできます。

AppStream 2.0 サービスを導入するときは、AppStream 2.0 インスタンスのサブネットに使用されるアドレス範囲が環境に適したサイトに割り当てられていることを確認してください。

シナリオ 1 とシナリオ 2 では、ログイン時間と Active Directory リソースへのアクセス時間において最適なユーザーエクスペリエンスを実現するには、サイトとサービスが重要になります。

サイトトポロジは、同じサイト内のドメインコントローラーや、サイトの境界を超えたドメインコントローラーの間で Active Directory レプリケーションを制御します。

正しいサイトトポロジを定義することで、クライアントアフィニティが保証されます。つまり、クライアント (この場合は AppStream 2.0 ストリーミングインスタンス) は任意のローカルドメインコントローラを使用します。



Active Directory サイトとサービスの — クライアントアフィニティ

> **Tip**
>
> ベストプラクティスとして、オンプレミスの AD DS と AWS クラウド間のサイトリンクには高いコストを設定します。上の図は、クライアントアフィニティがサイトに依存しないようにサイトリンクに割り当てる必要があるコスト (コスト 100) の一例です。

サイトトポロジの詳細については、「サイトトポロジの設計」を参照してください。

Active Directory の組織単位

AWS では、設定した組織単位 (OU) を 1 つの AppStream 2.0 Directory Config オブジェクトに保存することを推奨しています。AppStream 2.0 スタックごとに独自の OU を持つことがベストプラクティスです。これにより、スタックごとに特定の GPO を柔軟に設定できます。AppStream 2.0 固有のポリシーがオンプレミスデスクトップと混同されないように、OU を必ず AppStream 2.0 コンピュータオブジェクト専用にします。AppStream 2.0 をデプロイする AWS リージョン ごとに、サブ OU を使用することを検討してください。

Active Directory コンピュータオブジェクトのクリーンアップ

AppStream 2.0 インスタンスは一時的なものです。フリートは、フリートがスケールアウトしたりスケールインしたりするときに Active Directory コンピュータオブジェクトを作成して再利用します。

AWS では、AppStream フリートが削除された後に存在する可能性のある古い Active Directory コンピュータオブジェクトを削除するため、AD クリーンアッププロセスを作成することを推奨しています。

セキュリティ

クラウドセキュリティは Amazon Web Services (AWS) の最優先事項です。セキュリティとコンプライアンスは AWS と顧客の間で責任を共有します。詳細については、「責任共有モデル」を参照してください。AWS および AppStream 2.0 を利用するカスタマーは、スタック、フリート、イメージ、ネットワークなどのさまざまなレイヤーにセキュリティ対策を実装することが重要です。

AppStream 2.0 は本質的に一時的なため、多くの場合、アプリケーションやデスクトップ配信の安全なソリューションとして選択されます。Windows デプロイで一般的なウイルス対策ソリューションが、ユーザーセッションの終了時に事前定義されて削除される環境のユースケースに適しているかどうかについて検討してください。ウイルス対策は仮想化されたインスタンスにオーバーヘッドを追加するため、不要なアクティビティを軽減することがベストプラクティスとなっています。例えば、起動時に (一時的な) システムボリュームをスキャンしても AppStream 2.0 の全体的なセキュリティは向上しません。

AppStream 2.0 のセキュリティに関する 2 つの重要な質問は、主に次の点に関連しています。

- セッション終了後もユーザーの状態を維持することが必須か？
- ユーザーはセッション内でどのくらいのアクセス権限を持つべきか？

永続データの保護

AppStream 2.0 のデプロイでは、ユーザーの状態を何らかの形で保持しなければならない場合があります。個々のユーザーのデータを永続化する場合や、共有フォルダを使用してコラボレーション用にデータを保持する場合などです。AppStream 2.0 インスタンスストレージは一時的であり、暗号化オプションはありません。

AppStream 2.0 では、Amazon S3 のホームフォルダとアプリケーション設定を通じてユーザーの状態を永続化します。一部のユースケースでは、ユーザーの状態の永続化をより細かく制御する必要があります。このようなユースケースについては、AWS は、サーバーメッセージブロック (SMB) ファイル共有の使用を推奨しています。

ユーザーの状態とデータ

ほとんどの Windows アプリケーションは、ユーザーが作成したアプリケーションデータと同じ場所に配置すると最適かつ最も安全に動作するため、このデータを AppStream 2.0 フリートと同じ AWS リージョン に保持することがベストプラクティスです。このデータを暗号化することがベストプラクティスです。ユーザーのホームフォルダはデフォルトで、AWS キー管理サービス (AWS KMS) の

Amazon S3 マネージド暗号化キーを使用して保存中のファイルとフォルダを暗号化します。AWS コンソールまたは Amazon S3 バケットにアクセスできる AWS 管理ユーザーは、それらのファイルに直接アクセスできることに注意してください。

ユーザーファイルやフォルダを保存するために Windows ファイル共有のサーバーメッセージブロック (SMB) ターゲットを必要とする設計では、この処理は自動で行われるか、または設定が必要です。

表 5 — ユーザーデータを保護するためのオプション

SMB ターゲット	保管時の暗号化	転送時の暗号化	ウイルス対策 (AV)
FSx for Windows File Server	AWS KMS によって自動	SMB 暗号化によって自動	リモートインスタンスにインストールされた AV は、マップされたドライブでスキャンを実行します。
ファイルゲートウェイ、AWS Storage Gateway	デフォルトでは、S3 の AWS Storage Gateway に保存されたすべてのデータは、Amazon S3 マネージド暗号化キー (SSE-S3) によるサーバー側の暗号化を使用して暗号化されます。オプションで、保存されたデータを AWS Key Management Service (KMS) で暗号化するさまざまなゲートウェイタイプを設定できます。	あらゆるタイプのゲートウェイアプライアンスと AWS ストレージ間で転送されるデータはすべて SSL を使用して暗号化されます。	リモートインスタンスにインストールされた AV は、マップされたドライブでスキャンを実行します。
EC2 ベースの Windows File Server	EBS 暗号化を有効にする	PowerShell、`Set-SmbServer`	サーバーにインストールされた AV は、

SMB ターゲット	保管時の暗号化	転送時の暗号化	ウイルス対策 (AV)
		`Configuration - EncryptData $True`	ローカルドライブでスキャンを実行します。

エンドポイントセキュリティとウイルス対策

Amazon AppStream 2.0 インスタンスは本質的に一時的であり、データに永続性がないため、永続デスクトップで必要となるアクティビティによってユーザーのエクスペリエンスとパフォーマンスが損なわれないようにするには、別のアプローチが必要です。Endpoint Security エージェントは、組織のポリシーがある場合、または E メール、ファイル入力、外部ウェブブラウジングなどの外部データ入力で使用される場合、AppStream 2.0イメージにインストールされます。

一意識別子の削除

Endpoint Security エージェントにはグローバル一意識別子 (GUID) がある場合があり、フリートインスタンスの作成プロセス中にリセットする必要があります。ベンダーは、イメージから生成されたインスタンスごとに新しい GUID が確実に生成されるように、製品をイメージにインストールする手順を定めています。

GUID が生成されないようにするには、AppStream 2.0 Assistant を実行してイメージを生成する前の最後のアクションとして Endpoint Security エージェントをインストールします。

パフォーマンスの最適化

Endpoint Security ベンダーは、AppStream 2.0 のパフォーマンスを最適化するスイッチと設定を提供しています。設定はベンダーによって異なり、ベンダーのドキュメント (通常は VDI に関するセクション) に記載されています。一般的な設定には以下が含まれますが、これらに限定されません。

- 起動時のスキャンをオフにして、インスタンスの作成、起動、ログインにかかる時間を最小限に抑える
- 不要なスキャンを防ぐため、定期スキャンをオフにする
- ファイルが列挙されないように署名キャッシュをオフにする
- VDI に最適化された IO 設定を有効にする
- パフォーマンスを確保するためにアプリケーションが必要とする除外

Endpoint Security ベンダーは、パフォーマンスを最適化する仮想デスクトップ環境での使用方法を提供しています。

- Trend Micro Office Scan 仮想デスクトップインフラストラクチャのサポート - Apex One/OfficeScan (trendmicro.com)
- CrowdStrike と CrowdStrike Falcon をデータセンターに設置する方法
- Sophos と Sophos Central エンドポイント: ID の重複を避けるためにゴールドイメージにインストールする方法、および Sophos Central: 仮想デスクトップ環境に Windows エンドポイントをインストールする際のベストプラクティス
- McAfee と McAfee Agent の仮想デスクトップインフラストラクチャシステムへのプロビジョニングとデプロイ
- Microsoft Endpoint Security と非永続 VDI マシン用の Microsoft Defender Antivirus の設定 - Microsoft Tech Community

スキャンの除外

セキュリティソフトウェアが AppStream 2.0 インスタンスにインストールされている場合、セキュリティソフトウェアがプロセスに干渉しないようにする必要があります。

表 6 — AppStream 2.0 プロセスセキュリティソフトウェアが干渉してはいけないプロセス。

サービス	プロセス
AmazonCloudWatchAgent	"C:\Program Files\Amazon\AmazonCloudWatchAgent\start-amazon- cloudwatch-agent.exe"
AmazonSSMAgent	"C:\Program Files\Amazon\SSM\amazon-ssm-agent.exe"
NICE DCV	"C:\Program Files\NICE\DCV\Server\bin\dcvserver.exe" "C:\Program Files\NICE\DCV\Server\bin\dcvagent.exe"
AppStream 2.0	"C:\ProgramFiles\Amazon\AppStream2\StorageConnector\StorageConnector.exe" フォルダ "C:\Program Files\Amazon\Photon\"

サービス	プロセス
	".\Agent\PhotonAgent.exe" ".\Agent\s5cmd.exe" ".\WebServer\PhotonAgentWebServer.exe" ".\CustomShell\PhotonWindowsAppSwitcher.exe" ".\CustomShell\PhotonWindowsCustomShell.exe" ".\CustomShell\PhotonWindowsCustomShellBackground.exe"

フォルダ

セキュリティソフトウェアが AppStream 2.0 インスタンスにインストールされている場合、そのソフトウェアは次のフォルダに干渉しないようにします。

Example

```
C:\Program Files\Amazon\*
C:\ProgramData\Amazon\*
C:\Program Files (x86)\AWS Tools\*
C:\Program Files (x86)\AWS SDK for .NET\*
C:\Program Files\NICE\*
C:\ProgramData\NICE\*
C:\AppStream\*
C:\Program Files\Internet Explorer\*
C:\Program Files\nodejs\
```

Endpoint Security コンソールの健全性

Amazon AppStream 2.0 は、ユーザーがアイドルタイムアウトと切断タイムアウトを超えて接続するたびに、新しい一意のインスタンスを作成します。インスタンスには一意の名前が付けられ、エンドポイントセキュリティ管理コンソールに蓄積されます。4 日以上経過した (または AppStream 2.0 のセッションタイムアウトによってはそれ以下) 未使用の古くなったマシンを削除するように設定すると、コンソールに表示される期限切れのインスタンス数を最小限に抑えることができます。

ネットワーク除外

AppStream 2.0 管理ネットワーク範囲 (198.19.0.0/16) とそれに続くポートとアドレスは、AppStream 2.0 インスタンス内のセキュリティ、ファイアウォール、ウイルス対策ソリューションでブロックしないでください。

表 7 — AppStream 2.0 ストリーミングインスタンスのセキュリティソフトウェアが干渉してはならないポート

[ポート]	使用方法
8300、3128	これはストリーミング接続の確立に使用されます。
8000	これは、AppStream 2.0 によるストリーミングインスタンスの管理に使用されます。
8443	これは、AppStream 2.0 によるストリーミングインスタンスの管理に使用されます。
53	DNS

表 8 — AppStream 2.0 のマネージドサービスアドレスセキュリティソフトウェアがが干渉してはならないアドレス

[ポート]	使用方法
169.254.169.123	NTP
169.254.169.249	NVIDIA GRID ライセンスサービス

[ポート]	使用方法
169.254.169.250	KMS
169.254.169.251	KMS
169.254.169.253	DNS
169.254.169.254	メタデータ

AppStream セッションのセキュリティ保護

アプリケーションとオペレーティングシステムの制御の制限

AppStream 2.0 では、管理者はアプリケーションストリーミングモードでウェブページから起動できるアプリケーションを正確に指定できます。ただし、これによって、指定されたアプリケーションのみを実行できることが保証されるわけではありません。

Windows のユーティリティとアプリケーションは、別の方法でもオペレーティングシステムから起動できます。AWS では、Microsoft AppLocker を使用して、組織が必要とするアプリケーションのみを実行できるようにすることをお勧めします。デフォルトのルールでは、重要なシステムディレクトリへのパスアクセスがすべてのユーザーに許可されるため、変更する必要があります。

> **Note**
>
> Windows Server 2016 と 2019 では、AppLocker ルールを適用するために Windows Application Identity サービスを実行する必要があります。Microsoft AppLocker を使用する AppStream 2.0 からのアプリケーションアクセスについては、AppStream 管理者ガイドに詳しく記載されています。

Active Directory ドメインに参加しているフリートインスタンスの場合は、グループポリシーオブジェクト (GPO) を使用してユーザーとシステム設定を配信し、ユーザーのアプリケーションとリソースへのアクセスを保護します。

ファイアウォールとルーティング

AppStream 2.0 フリートを作成するときは、サブネットとセキュリティグループを割り当てる必要があります。サブネットには、ネットワークアクセスコントロールリスト (NACL) とルートテーブルが既に割り当てられています。新しい Image Builder の起動中、または新しいフリートの作成中に、最大 5 つのセキュリティグループを関連付けることができます。セキュリティグループには、既存のセキュリティグループから最大 5 つの割り当てを関連付けることができます。セキュリティグループごとに、インスタンスとの間で送受信されるネットワークトラフィックのアウトバウンドとインバウンドを制御するルールを追加します。

NACL は1 つまたは複数の サブネットのインバウンドトラフィックとアウトバウンドトラフィックを制御するためのファイアウォールとして機能する任意指定の VPC セキュリティレイヤーです。セキュリティの追加レイヤーを VPC に追加するには、セキュリティグループと同様のルールを指定したネットワーク ACL をセットアップできます。セキュリティグループとネットワーク ACL の違いの詳細については、「セキュリティグループとネットワーク ACL を比較する」のページを参照してください。

セキュリティグループと NACL のルールを設計して適用するときは、権限を最小限に抑えるための AWS Well-Architected のベストプラクティスを検討してください。最小特権とは、タスクを完了するために必要なアクセス許可のみを付与する原則です。

オンプレミス環境を (AWS Direct Connect 経由で) AWS に接続する高速プライベートネットワークを保有するカスタマーは、AppStream 用の VPC エンドポイントの使用を検討してください。この場合、ストリーミングトラフィックは、パブリックインターネットを経由するのではなく、プライベートネットワーク接続を介してルーティングされます。このトピックの詳細については、このドキュメントの「AppStream 2.0 ストリーミングインターフェイス VPC エンドポイント」のセクションを参照してください。

データ損失防止

ここでは、2 種類のデータ損失防止について確認しますす。

クライアントから AppStream 2.0 インスタンスへのデータ転送の制御

表 9 — データの入出力の制御に関するガイダンス

設定	オプション	ガイダンス
クリップボード	• リモートセッションにのみコピーと貼り付け • ローカルデバイスにのみコピー • 無効	この設定を無効にしても、セッション内のコピーと貼り付けは無効になりません。セッションにデータをコピーする必要がある場合は、データ漏洩の可能性を最小限に抑えるため、[リモートセッションにのみ貼り付ける] を選択してください。
ファイル転送	• アップロードとダウンロード • アップロードのみ • ダウンロードのみ • 無効	データ漏えいを防ぐため、この設定は有効にしないでください。
ローカルデバイスへの印刷	• 有効 • 無効	印刷が必要な場合は、組織によって制御およびモニタリングされているネットワークマッピングされたプリンタを使用してください。

既存の組織データ転送ソリューションがスタック設定よりも優れている点について検討してください。これらの設定は、包括的なセキュアデータ転送ソリューションに代わるものではありません。

AppStream 2.0 インスタンスからの出力トラフィックの制御

データ損失が懸念される場合は、ユーザーが AppStream 2.0 インスタンス内に入ったときにアクセスできるものを非表示にすることが重要です。ネットワークの出口 (または出力) パスはどのようになっていますか？ エンドユーザーが AppStream 2.0 インスタンス内でパブリックインターネットにアクセスできるようにすることは一般的な要件であるため、ネットワークパスに WebProxy またはコンテンツフィルタリングソリューションを配置することを検討する必要があります。その他の考慮事項には、ローカルのウイルス対策アプリケーションと AppStream インスタンス内のその他のエン

ドポイントセキュリティ対策が含まれます (詳細については、「エンドポイントセキュリティとウイルス対策」のセクションを参照してください)。

AWS サービスの使用

AWS Identity and Access Management

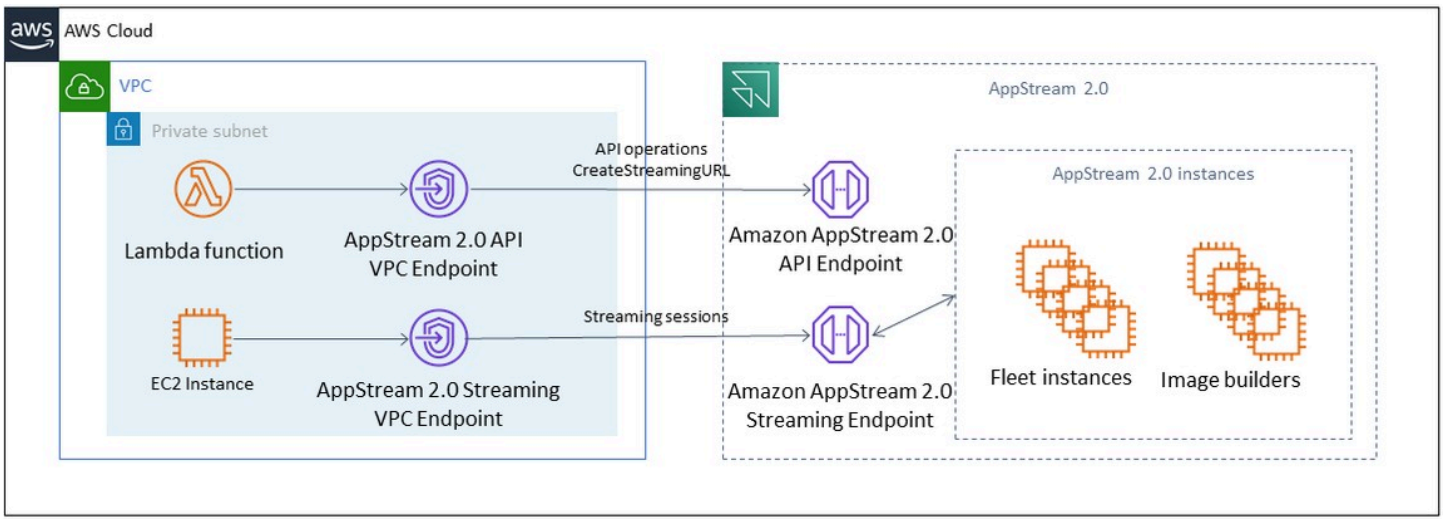
追加の認証情報を管理せずに AppStream 2.0 セッションのユーザーのみにアクセスできるようにするベストプラクティスは、IAM ロールを使用して AWS サービスにアクセスし、そのロールにアタッチされている IAM ポリシーを具体的に指定することです。AppStream 2.0 で IAM ロールを使用する際のベストプラクティスに従ってください。

ユーザーデータをホームフォルダとアプリケーション設定の永続化の両方に保持するために作成された Amazon S3 バケットを保護するための IAM ポリシーを作成します。これにより、AppStream 2.0 管理者以外はアクセスできなくなります。

VPC エンドポイント

VPC エンドポイントは、VPC およびサポートされている AWS のサービス、AWS PrivateLink による VPC エンドポイントサービスとの間のプライベート接続を可能にします。AWS PrivateLink は、プライベート IP アドレスを経由してサービスにプライベートにアクセスできるテクノロジーです。VPC と他のサービス間のトラフィックは、Amazon ネットワークを離れません。パブリックインターネットアクセスが AWS サービスにのみ必要な場合、VPC エンドポイントは NAT ゲートウェイとインターネットゲートウェイの要件を完全に削除します。

自動化ルーチンや開発者が AppStream 2.0 の API 呼び出しを行う必要がある環境では、AppStream 2.0 API オペレーション用のインターフェイス VPC エンドポイントを作成します。例えば、パブリックインターネットへのアクセスがないプライベートサブネットに EC2 インスタンスがある場合、AppStream 2.0 API の VPC エンドポイントを使用して CreateStreamingURL などの AppStream 2.0 API オペレーションを呼び出すことができます。次の図は、AppStream 2.0 API とストリーミング VPC エンドポイントが Lambda 関数と EC2 インスタンスによって使用される設定例を示しています。



VPC エンドポイント

ストリーミング VPC エンドポイントでは、VPC エンドポイントを介してセッションをストリーミングできます。ストリーミングインターフェイスエンドポイントは、VPC 内のストリーミングトラフィックを維持します。ストリーミングトラフィックには、ピクセル、USB、ユーザー入力、オーディオ、クリップボード、ファイルのアップロードとダウンロード、プリンターのトラフィックが含まれます。VPC エンドポイントを使用するには、AppStream 2.0 スタックで VPC エンドポイント設定が有効になっている必要があります。これは、インターネットアクセスが制限されていて、Direct Connect インスタンス経由でアクセスしたほうが有利な場所から、パブリックインターネット経由でユーザーセッションをストリーミングする代替手段となります。VPC エンドポイントを介してユーザーセッションをストリーミングするには、以下が必要です。

- インターフェイスエンドポイントに関連付けられているセキュリティグループは、ユーザーが接続する IP アドレス範囲からポート 443 (TCP) とポート 1400–1499 (TCP) へのインバウンドアクセスを許可する必要があります。
- サブネットのネットワークアクセスコントロールリストでは、一時ネットワークポート 1024-65535 (TCP) から、ユーザーが接続する IP アドレス範囲へのアウトバウンドトラフィックを許可する必要があります。
- ユーザーを認証し、AppStream 2.0 が機能するために必要なウェブアセットを配信するためには、インターネットに接続できることが必須です。

AppStream 2.0 による AWS サービスへのトラフィックの制限について詳しくは、VPC エンドポイントからの作成とストリーミングに関する管理ガイドをご覧ください。

パブリックインターネットへの完全なアクセスが必要な場合は、Image Builder で Internet Explorer のセキュリティ強化構成 (ESC) を無効にするのがベストプラクティスです。詳細については、

「AppStream 2.0 管理ガイド」の「Internet Explorer セキュリティ強化構成を無効にする」を参照してください。

ディザスタリカバリ

Amazon AppStream 2.0 には、最大 3 つのアベイラビリティーゾーンにわたる冗長性が組み込まれています。つまり、あるアベイラビリティーゾーンにアクティブなセッションがあり、そのセッションが機能しなくなった場合、そのユーザーは接続を切断して再接続するだけで、正常なアベイラビリティーゾーンにセッションを予約できます (キャパシティに余裕があること)。これにより、リージョン内の可用性は高くなりますが、リージョンレベルでサービスに問題が発生した場合は、ディザスタリカバリのソリューションにはなりません。

AppStream 2.0 ユーザーにディザスタリカバリプランを提供するには、まずセカンダリリージョンに AppStream 2.0 環境を構築する必要があります。設計の観点から、該当する場合、この環境にはオンプレミス環境への冗長接続が必要であり、プライマリリージョンに依存しないようにする必要があります。例えば、AppStream 2.0 フリートがドメインに参加している場合、サイトとサービスを設定したセカンダリリージョンに追加のドメインコントローラが必要です。AppStream 2.0 の観点から、この環境はプライマリリージョンと同じフリートとスタック設定で構成されている必要があります。フリート自体は同じベースイメージを実行する必要があり、コンソールまたはプログラムでセカンダリリージョンにコピーできます。AppStream 2.0 セッション内で実行されるアプリケーションのバックエンド依存関係がプライマリリージョンに関連付けられている場合は、プライマリリージョンがダウンしてもユーザーがアプリケーションのバックエンドにアクセスできるように、そのアプリケーションにもリージョンの冗長性が必要です。送信先リージョンのサービスレベル制限は、プライマリリージョンと一致している必要があります。

ID ルーティング

DR シナリオでアプリケーションへのアクセスを提供するには、2 つの異なる方法があります。概して、この 2 つの方法は、ユーザーをフェイルオーバーリージョンに誘導する方法が異なります。1 つ目の方法は IdP 内の 1 つの AppStream 2.0 アプリケーション設定を使用して実行され、2 つ目の方法には 2 つの個別のアプリケーション設定があります。

方法 1: アプリケーションのリレーステートを変更する

ユーザーが ID プロバイダー (IdP) から AppStream 2.0 にログインすると、認証後、アクセスする予定のリージョンとスタックに合わせた特定の URL にリレーされます。リレーステート URL の詳細については、「Amazon AppStream 2.0 管理ガイド」を参照してください。管理者は、ユーザーがフェールオーバーするプライマリリージョンと同じ AppStream 2.0 イメージ上に構築されたクロスリージョンスタックを設定できます。管理者は、リレーステート URL をフェイルオーバースタックを指すように更新するだけで、このフェイルオーバーを制御できます。この方法が正しく動作するた

めには、関連する IAM ポリシーにプライマリスタックとフェイルオーバースタックの両方へのアクセスが反映されている必要があります。これらの IAM ポリシーの設定方法の詳細については、以下のポリシーの例を参照してください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "appstream:Stream",
            "Resource": [
            "arn:aws:appstream:PrimaryRegion:190836837966:stack/StackName",
            "arn:aws:appstream:FailoverRegion:190836837966:stack/StackName"
            ],
            "Condition": {
                "StringEquals": {
                    "appstream:userId": "${saml:sub}"
                }
            }
        }
    ]
}
```

方法 2: IdP 内で 2 つの AppStream 2.0 アプリケーションを設定する

この方法では、管理者が IdP 内で AppStream 2.0 用の 2 つのアプリケーションを個別に構築する必要があります。その後、両方のアプリケーションを表示し、ユーザーにどこに移動するかを選択させるか、またはユーザーはフェールオーバーするまでアプリケーションをロックまたは非表示にできます。この方法は、頻繁に移動するグローバルユーザーがいるユースケースに適しています。これらのユーザーは、最も近いエンドポイントからストリーミングする必要があるため、両方のアプリケーションを割り当てておくと、最も近いリージョンに設定されているアプリケーションを選択できます。これを自動化することもできます。詳細については、このブログ記事を参照してください。

ストレージの永続化

アプリケーション永続化やホームフォルダ同期など、AppStream 2.0 に含まれるデータ永続機能を利用する場合、そのデータをフェールオーバーリージョンに複製する必要があります。これらの機能は、特定の AppStream 2.0 リージョンの Amazon S3 バケットに永続データを保存します。データをリージョン間で保持するには、ソースバケットのすべての変更をフェイルオーバーリージョ

ン AppStream 2.0 バケットに複製する必要があります。これは、Amazon S3 のクロスリージョンレプリケーションなどの Amazon S3 のネイティブ機能を使用して実行できます。各ユーザーの永続データは、ハッシュされたユーザー名のフォルダに保存されます。ユーザー名は同じクロスリージョンでハッシュされるため、データを複製するだけでセカンダリージョンのデータを保持できます。AppStream 2.0 Amazon S3 バケットの詳細については、このガイドを参照してください。

モニタリング

ダッシュボードの使用

フリート使用率のモニタリングは、CloudWatch メトリクスとダッシュボードの作成により実行できる定期的なアクティビティです。または、AppStream 2.0 コンソールから [フリートの使用状況] タブを使用します。ユーザーの行動は常に予測できるわけではなく、需要が最初に評価した事前計画をも上回ることもあるため、フリートの使用状況を定期的にモニタリングします。CloudWatch の AppStream 2.0 メトリクスとディメンションの全リストは、AppStream 2.0 管理ガイドの「モニタリングリソース」に記載されています。

成長の予測

`PendingCapacity` で大規模なジャンプが発生するたびに、自動スケーリングイベントが発生します。ここで重要なのは、新しい AppStream 2.0 フリートインスタンスがユーザーセッションをホストできるようになるまでは、`AvailableCapacity` および `PendingCapacity` に逆相関があることを確認しておくことです。各 AppStream 2.0 フリートの `InsufficientCapacityError` に対して CloudWatch アラームを作成して、自動スケーリングが需要に遅れないように管理者に通知します。

需要がキャパシティを超え、`InsufficientCapacityError` メトリクス値が一般的である場合は、勤務日の開始時のスケジュールされたスケーリングポリシーにより、最小キャパシティの値を上げることを検討します。さらに、需要が満たされた後に最小キャパシティを引き下げる 2 つ目のスケジュールスケーリングポリシーも設定します。最小キャパシティの値を下げても、既存のセッションには影響しません。勤務日の終了前に最小キャパシティを下げると、`ActualCapacity` の値が下がり、スケールが意図したとおり効果的に機能するようになります。これによりコストが最適化されます。

需要が一貫して予測できない場合は、ターゲット追跡スケーリングポリシーを使用して、使用パターンを判断しながら、AppStream 2.0 フリートに需要を満たす十分な `AvailableCapacity` があることを確認します。ターゲット追跡はフリートの消費量の一定の割合を使用するため、引き続きモニタリングします。フリートインスタンスの総数が増えるにつれて、未使用のフリートインスタンスの総数も増えます。最大キャパシティを控えめな値に設定しない限り、これは無駄になる可能性があります。信頼性とコスト最適化のバランスをとるために、複数の種類のスケーリングポリシー (スケジュール管理やターゲット追跡など) を使用します。

ユーザー使用状況のモニタリング

ユーザー料金という形でコストが発生するため、固有のユーザーをモニタリングします。このユーザー料金のコストは、Image Assistant (RDS) のサブスクライバーアクセスライセンス (SAL) によるものです。固有のユーザーの評価は、認証が実行される IdP からのレポートまたは使用状況レポートを通じて実行できます。

使用状況レポートは S3 バケットに個別の `.csv` ファイルとして保存され、ダウンロードしてサードパーティのビジネスインテリジェンス (BI) ツールを使用して分析できます。レポートをダウンロードせずに AWS の使用状況データを分析したり、複数の `.csv` ファイルを連結せずにカスタムの日付範囲のレポートを作成することができます。例えば、Amazon Athena と Amazon QuickSight を使用して、AppStream 2.0 の使用状況データのカスタムレポートおよび可視化を作成できます。

アプリケーションイベントログと Windows イベントログの保存

AppStream 2.0 インスタンスセッションが完了すると、インスタンスは終了します。つまり、セッションで使用されていたアプリケーションと Windows のイベントログはすべて失われます。これらのアプリケーションおよび Windows イベントログを保持する必要がある場合、1 つの方法は、Amazon Data Firehose を使用してイベントログを S3 にリアルタイムに配信し、Amazon OpenSearch Service (OpenSearch Service) で検索することです。クエリが頻繁に発生しないことが予想される場合は、コストを最適化するために、Amazon OpenSearch Service を実行するのではなく、Amazon Athena を使用して検索します。

ネットワークと管理アクティビティの監査

まだセットアップしていない場合は、Amazon AppStream 2.0 を使用して AWS アカウント の AWS CloudTrail を設定するのがベストプラクティスです。特に AppStream 2.0 API 呼び出しを監査するには、`appstream.amazonaws.com` の値でのフィルタイベントソースを使用します。

VPC フローログを有効にして、カスタマーマネージドリソースへのアクセスを監査します。VPC フローログを CloudWatch Logs にパブリッシュすることで、監査が必要な場合にクエリを実行できます。

AppStream 2.0 のフリートが増えるにつれて、サブネット IP の割り当てのモニタリングが重要になります。describe-subnets CLI を実行して、フリートに割り当てられた各サブネットで使用可能な IP アドレスを報告することで、IP の割り当てについてレポートします。最大キャパシティで実行しているすべてのフリートの需要を満たすのに十分な IP アドレスのキャパシティが組織にあることを確認します。

コスト最適化

コスト最適化は、不必要なコストを回避することに焦点を当てています。主なトピックには、お金が使われている場所を理解し、管理すること、そして最も適切かつ正しい数のリソースタイプを選択することが含まれます。経時的な支出とビジネスニーズに合わせたスケーリングを分析します。以下の AppStream 2.0 リソースでは従量制料金が発生します。

- 常時オンのフリートインスタンス
- オンデマンドのフリートインスタンス
- オンデマンドの停止インスタンスの料金
- Image Builder インスタンス
- ユーザー料金

最新の料金情報については、Amazon AppStream 2.0 料金表の AWS ウェブサイトを参照してください。

コスト効率に優れた AppStream 2.0 デプロイの設計

AppStream 2.0 デプロイの計画と設計における最初のステップは、シンプルな価格設定ツールを使用して、使用量に関連する AWS 料金のベースラインを見積もることです。ユーザーの総数、1 時間あたりの実際の同時使用量、インスタンスタイプ、フリート使用率を入力すると、価格設定ツールがユーザー 1 人あたりの価格を見積もります。また、常時オンフリートの代わりにオンデマンドフリートを使用した場合の削減見込み額も表示されます。

カスタマーは、ユーザーのストリーミングニーズを満たすためにプロビジョニングしたインスタンスに対してのみ支払いを行う AppStream 2.0 の価格モデルを好みます。このモデルは、既存のアプリケーションストリーミング環境とは異なります。これらは通常、負荷の低い夜間、週末、休日であっても、ピーク時のキャパシティをプロビジョニングすることが基本となっています。Amazon AppStream 2.0 価格設定ツールは、AppStream 2.0 の使用に関連する AWS 料金の見込み額を提供するだけで、適用される可能性のある税金は含まれていません。実際の料金は、AWS サービスの実際の使用状況など、さまざまな要因によって異なります。

AppStream 2.0 価格設定ツールは、Microsoft Excel または OpenOffice Calc のスプレッドシートとして提供され、フリートに関する基本情報を入力できるほか、使用パターンに基づいてオンデマンドフリートと常時オンフリートの AppStream 2.0 環境のコストの見込み額を提供します。過去または予想される使用傾向に基づいてコストをシミュレートできます。Elastic フリートにはこれらの機能が

組み込まれているため、管理者は使用量の予測、スケーリングポリシーやイメージの作成、管理を行う必要がありません。Elastic フリートと Amazon Linux 2 で実行されているインスタンス (すべてのフリートタイプ) には、ストリーミングセッションの時間 (秒単位) に対して課金されます (最低 15 分)。

インスタンスタイプの選択によるコストの最適化

フリートインスタンスと Image Builder インスタンスでは、アプリケーションに合わせてさまざまなインスタンスファミリーとタイプを選択できます。

エンドユーザーテスト — 次のステップでは、AppStream 2.0 フリートをパイロットユーザーのグループにロールアウトして、選択したインスタンスタイプを検証するためのテストを行います。パイロットユーザーに、メモリ、CPU、グラフィックに関するメトリクスをキャプチャして、通常のワークフローと負荷の高いワークフローをすべてテストするように依頼し、ベースラインのパフォーマンスメトリクスをキャプチャできるようにすることが重要です。パイロットグループには、複数のユーザーエクスペリエンスからアプリケーションをテストしていることを確認するために、アプリケーションを使用するさまざまなユーザーロールを含める必要があります。ユーザー受け入れテストを実施することで、ストリーミングセッションのエクスペリエンスに関するフィードバックを収集できます。スタックの作成または更新時には、カスタムフィードバック URL を使用するオプションがあります。ユーザーは、アプリケーションストリーミングのエクスペリエンスについて [フィードバックを送信] リンクをクリックすると、この URL にリダイレクトされます。パフォーマンスのボトルネックがある場合は、Windows のパフォーマンス指標を使用してリソース制限について分析してください。例えば、現在のフリートインスタンスタイプ stream.standard.medium がリソース制約を示している場合、そのインスタンスタイプを stream.standard.large にアップグレードします。逆に、パフォーマンスメトリクスにより、リソースの使用率が低いことが多いと分かった場合は、インスタンスタイプをダウングレードすることを検討します。

フリートタイプの選択によるコストの最適化

新しい AppStream 2.0 フリートを作成する場合、開発者はフリートタイプを常時オンまたはオンデマンドのどちらかを選択する必要があります。価格の観点からインスタンスタイプを選択する際には、AppStream 2.0 がフリートインスタンスをどのように管理するかを理解することが重要です。常時オンフリートでは、フリートインスタンスは実行状態のままです。そのため、ユーザーがセッションをストリーミングしようとしても、フリートインスタンスはいつでもストリーミングセッションを開始できる状態になっています。

オンデマンドフリートの場合、フリートインスタンスは起動後も停止状態のままになります。停止したインスタンス料金は、実行中のインスタンス料金よりも低くなるため、コスト削減に役立ちます。

オンデマンドフリートインスタンスは停止状態から起動する必要があります。ユーザーはストリーミングセッションが使用可能になるまで約 2 分待つ必要があります。

自己完結型で Amazon Simple Storage Service (Amazon S3) バケットに保存された仮想ハードドライブにインストールできるアプリケーションには、Elastic フリートが適しています。Elastic フリートでは、1 秒あたりの請求がストリーミング中のみのため、ユースケースによってはコストをさらに削減できる可能性があります。料金は、フリートを作成するときに選択したインスタンスタイプ、サイズ、オペレーティングシステムによって異なります。

エンドユーザーが営業時間中にフリートインスタンスを必要とする場合は、同じストリーミングセッションを維持する方が適しています。これは、フリートインスタンスは 1 時間ごとに課金され、新しいストリーミングセッションが開始されるたびに、別のフリートインスタンス料金が発生するためです。

表 10 — AppStream 2.0 フリートタイプの比較

フリートタイプ	利点	考慮事項
常時オン	ストリーミングセッションの待ち時間を短縮できる	ユーザーは、インスタンスを停止状態に保つオプションがないため、時間単位のインスタンス料金を支払います。
オンデマンド	インスタンスが停止状態のままになるためコストを節約できる	ストリーミングセッションの待ち時間が長期化する
Elastic	秒単位の課金は、仮想ハードディスクにインストールできるアプリケーションの使用パターンが散発的なユースケースに役立つ場合があります。	アプリケーションの仮想ハードディスクのサイズが大きくなると、ストリーミングインスタンスへのマウントにかかる時間が長くなる可能性があります。

AppStream 2.0 はフリートの使用状況をモニタリングし、ユーザーの需要を満たすようにフリートのキャパシティを自動的に調整して、可能な限り低いコストで実現します。キャパシティの調整は、現在の使用率またはスケジュールに基づき、定義したスケーリングポリシーを基に行われます。フリー

ト使用状況のメトリクスを定期的に見直して、フリートスケーリングポリシーに高レベルの予備キャパシティがないことを確認します。

スケーリングポリシー

フリート自動スケーリングでは、ユーザーのログインを待つリソースを過剰にコミットする必要がないため、フリートリソースを最適化できます。管理者は、ユーザーの需要に合わせて、さまざまな使用状況に基づいてフリートのサイズを調整できます。CloudWatch AppStream 2.0 フリートメトリクスまたはサードパーティのモニタリングツールを使用して、ユーザーのアクティビティを把握し、予想される使用状況に基づいて AppStream 2.0 フリートを拡張または縮小するスケーリングポリシーを設定します。ユーザーログは、実際の使用状況を把握するために不可欠なメカニズムです。この分析情報を使用して、自動スケーリングに基づいてフリートのサイズを動的に変更することができます。

多くの場合、AppStream 2.0 フリートは最大ユーザー数に基づいて作成されており、夜間や週末など、1 日の異なる時間帯に合わせて調整されることはありません。多くの場合、ストリーミングアプリケーションの同時ユーザー数は、特にユーザーがリモートで柔軟に作業できる場合、ユーザーの総数よりも少なくなります。使用パターンを予測する際には、これらの要素を考慮に入れることが重要です。過大に評価すると AppStream 2.0 インスタンスをプロビジョニングし過ぎることになり、コストが増加します。最適な構成を実現するには、必要に応じて、1 つ以上のスケジュール済みのスケーリングポリシーをスケールアウトポリシーと組み合わせます。

スケーリングポリシーの実装について詳しくは、「Amazon AppStream 2.0 フリートのスケーリング」を参照してください。

ユーザー料金

ユーザー料金は、ユーザーが AppStream 2.0 フリートインスタンスからアプリケーションをストリーミングするたびに、各 AWS リージョン のユーザー 1 人あたり、1 か月ごとに請求されます。AppStream 2.0 ユーザーには、異なるユーザー ID を生成するのではなく、一貫したユーザー ID を用意します。Image Builder に接続するとき、ユーザー料金は請求されません。

学校、大学、および特定の公共機関によっては、Microsoft RDS SAL ユーザー料金を 1 ユーザーにつき 1 か月あたり 0.44 USD の割引を受けることができます。資格要件については、「Microsoft ライセンス条項およびドキュメント」を参照してください。

Microsoft ライセンスモビリティをお持ちの場合は、独自の Microsoft RDS クライアントアクセスライセンス (CAL) を Amazon AppStream 2.0 で使用できる場合があります。お持ちのライセンス

の対象であれば、毎月のユーザー料金は発生しません。既存の Microsoft RDS CAL ライセンスを Amazon AppStream 2.0 で使用できるかどうかの詳細については、「AWS ライセンスモビリティガイダンス」を参照するか、Microsoft のライセンス担当者に相談してください。

Image Builder の使用

AppStream 2.0 Image Builder インスタンスは、時間単位で課金されます。Image Builder インスタンスの料金には、コンピューティング、ストレージ、およびストリーミングプロトコルで使用されるすべてのネットワークトラフィックが含まれます。実行中のすべての Image Builder インスタンスには、該当する実行インスタンス料金が課金されます。この料金は、管理者が接続していない場合でも、インスタンスタイプとサイズに基づきます。

コストを最適化するためのベストプラクティスとして、使用していないときは Image Builder インスタンスをシャットダウンします。CloudWatch イベントルールを使用して、Lambda 関数を呼び出して Image Builder のインスタンスを停止するなど、毎日のジョブをスケジュールできます。

AppStream 2.0 イメージは、AppStream 2.0 マネージドイメージ更新を使用して最新の状態に保つことができます。この更新機能では、最新の Windows オペレーティングシステムの更新とドライバーの更新、および最新の AppStream 2.0 エージェントソフトウェアが提供されます。この方法を使用してイメージを更新すると、マネージドサービスプロセスの一環として Image Builder が自動的に起動および停止されます。

結論

AppStream 2.0 では、既存のデスクトップアプリケーションを AWS に簡単に追加して、ユーザーがそれらを即座にストリーミングできるようにすることが可能です。Windows ユーザーは AppStream 2.0 クライアントまたは HTML5 対応ウェブブラウザをアプリケーションのストリーミングに使用できます。アプリケーションごとに 1 つのバージョンを維持すればよいため、簡単にアプリケーションを管理できます。ユーザーはいつでも最新バージョンのアプリケーションにアクセスできます。アプリケーションは AWS のコンピューティングリソースで実行され、データがユーザーのデバイスに保存されることはないので、ユーザーは常に高性能でセキュアなエクスペリエンスを得ることができます。

デスクトップアプリケーションストリーミングの従来のオンプレミスソリューションとは違い、AppStream は従量制料金です。そのため、先行投資やインフラストラクチャの維持が不要です。すぐに、グローバルにスケーリングできるため、いつでも際立った体験をユーザーに提供できます。

Amazon AppStream 2.0 は既存の IT システムやプロセスに統合できるように設計されており、このホワイトペーパーではそのためのベストプラクティスについて説明しています。このホワイトペーパーのガイドラインに従うと、AWS グローバルインフラストラクチャ上でビジネスに合わせて安全にスケールできる、費用対効果の高いクラウドデスクトップデプロイを実現できます。

寄稿者

このドキュメントの寄稿者は次のとおりです。

- Andrew Wood、Amazon Web Services、シニアソリューションアーキテクト
- Andrew Morgan、Amazon Web Services、EUC スペシャリスト SA
- Arun PC、Amazon Web Services、シニア EUC スペシャリスト SA
- Asriel Agronin、Amazon Web Services、シニアソリューションアーキテクト
- Dustin Shelton、Amazon Web Services、シニア EUC スペシャリスト SA
- Jeremy Schiefer、Amazon Web Services、シニアソリューションアーキテクト
- Navi Magee、Amazon Web Services、プリンシパルソリューションアーキテクト
- Pete Fergus、Amazon Web Services、シニアクラウドサポートエンジニア
- Phil Persson、Amazon Web Services、プリンシパル EUC スペシャリスト SA
- Richard Spaven、Amazon Web Services、シニア EUC スペシャリスト SA
- Spencer DeBrosse、Amazon Web Services、シニアソリューションアーキテクト
- Stephen Stetler、Amazon Web Services、シニアソリューションアーキテクト
- Taka Matsumoto、Amazon Web Services、シニアクラウドサポートエンジニア
- Vasant Sirsat、Amazon Web Services、シニア EUC スペシャリスト SA

詳細情報

詳細については、次を参照してください。

- Amazon AppStream 2.0 管理ガイド
- Amazon AppStream API リファレンス
- Amazon FSx for Windows File Server と FSLogix を使用して Amazon AppStream 2.0 のアプリケーション設定の永続化を最適化する
- Amazon ElasticSearch と Amazon Firehose による Amazon AppStream 2.0 のモニタリング
- Amazon Athena と Amazon QuickSight を使用して Amazon AppStream 2.0 の使用状況レポートを分析する
- Amazon AppStream 2.0 フリートをスケーリングする
- Microsoft AppLocker による Amazon AppStream 2.0 のアプリケーションエクスペリエンスの管理
- Amazon AppStream 2.0 でのカスタムドメインの使用
- 独自の Microsoft RDS CAL を AppStream 2.0 で使用する方法を教えてください。
- Amazon AppStream 2.0 価格設定ツール
- AppStream 2.0 でオンラインソフトウェアトライアルを作成する
- Amazon AppStream 2.0 で SaaS ポータルを作成する

ドキュメントの改訂

このホワイトペーパーの更新に関する通知を受け取るには、RSS フィードにサブスクライブしてください。

変更	説明	日付
ドキュメントの更新	Elastic フリート、属性ベースのアプリケーションの使用権限、マルチスタックアプリケーションカタログ、Linux ベースのフリート、データ入出力、ディザスタリカバリを含む更新、およびその他の更新。	2022 年 6 月 14 日
ドキュメントの更新	HTML バージョンの公開。	2022 年 1 月 19 日
初版発行	ホワイトペーパーの発行。	2021 年 6 月 8 日

注意

お客様は、本書に記載されている情報を独自に評価する責任を負うものとします。本書は、(a) 情報提供のみを目的とし、(b) AWS の現行製品と慣行について説明しており、これらは予告なしに変更されることがあり、(c) AWS およびその関連会社、サプライヤー、またはライセンサーからの契約上の義務や保証をもたらすものではありません。AWS の製品やサービスは、明示または黙示を問わず、一切の保証、表明、条件なしに「現状のまま」提供されます。お客様に対する AWS の責任は AWS 契約によって規定されます。本書は、AWS とお客様との間で締結されるいかなる契約の一部でもなく、その内容を修正するものでもありません。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。