aws

ユーザーガイド

AWS エンドユーザーメッセージングプッシュ

AWS エンドユーザーメッセージングプッシュ: ユーザーガイド

Table of Contents

AWS エンドユーザーメッセージングプッシュとは

> **Note**
>
> Amazon Pinpoint のプッシュ通知機能は、 AWS エンドユーザーメッセージングと呼ばれるようになりました。

AWS エンドユーザーメッセージングプッシュを使用すると、プッシュ通知チャネルを介してプッシュ通知を送信することで、アプリケーションのユーザーをエンゲージできます。Apple Push Notification Service (APNsFirebase Cloud Messaging (FCM)、Amazon Device Messaging (ADM)、Baidu Push をサポートしています。

トピック

AWS エンドユーザーメッセージングプッシュを初めてお使いになる方向けの情報

AWS エンドユーザーメッセージングプッシュを初めて使用する場合は、まず以下のセクションを読むことをお勧めします。

AWS エンドユーザーメッセージングプッシュの機能

アプリケーションにプッシュ通知を送信するには、以下のプッシュ通知サービスで個別のチャネルを使用します。

- Firebase Cloud Messaging (FCM)
- Apple プッシュ通知サービス (APNs)

> **Note**
>
> APNs を利用して、iPhone や iPad などの iOS デバイスや、Mac のラップトップやデスクトップなどの macOS デバイスの Safari ブラウザにメッセージを送信できます。

- Baidu Cloud Push
- Amazon Device Messaging (ADM)

AWS エンドユーザーメッセージングプッシュへのアクセス

コンソール、CLI、または API を使用して、サービスへのアクセスを取得するさまざまな方法を簡単に説明します。

次のインターフェイスを使用して AWS 、エンドユーザーメッセージングプッシュを管理できます。

AWS エンドユーザーメッセージングプッシュコンソール

AWS エンドユーザーメッセージングプッシュリソースを作成および管理するためのウェブインターフェイス。にサインアップしている場合は AWS アカウント、 から AWS End User Messaging Push コンソールにアクセスできます AWS マネジメントコンソール。

AWS Command Line Interface

コマンドラインシェルのコマンドを使用して AWS サービスとやり取りします。 AWS Command Line Interface は、Windows、macOS、および Linux でサポートされています。の詳細については AWS CLI、AWS Command Line Interface 「 ユーザーガイド」を参照してください。 AWS エンドユーザーメッセージングプッシュコマンドはAWS CLI 、 コマンドリファレンスにあります。

AWS SDK

HTTP または HTTPS 経由でリクエストを送信するのではなく、言語固有の APIs を使用してアプリケーションを構築するソフトウェア開発者の場合、 はライブラリ、サンプルコード、チュートリアル、その他のリソース AWS を提供します。これらのライブラリは、リクエストへの暗号署名、リクエストの再試行、エラーレスポンスの処理などのタスクを自動化する基本的な機能を提供します。これらの関数は、使用開始をより効率的にするのに役立ちます。詳細については、「AWSでの構築ツール」を参照してください。

リージョナルな可用性

AWS エンドユーザーメッセージングプッシュは、北米、欧州、アジア、オセアニア AWS リージョン の複数の で利用できます。各リージョンで、 は複数のアベイラビリティーゾーン AWS を維持します。これらのアベイラビリティーゾーンは物理的に相互に分離されていますが、低レイテンシーで高スループットの冗長性に優れたプライベートネットワーク接続で統合されています。これらのアベイラビリティーゾーンは、レイテンシーを最小限に抑えながら、非常に高いレベルの可用性と冗長性を提供するために使用されます。

詳細については AWS リージョン、「」のAWS リージョン 「アカウントで使用できる を指定する」を参照してくださいAmazon Web Services 全般のリファレンス。 AWS エンドユーザーメッセージングプッシュが現在利用可能なすべてのリージョンと各リージョンのエンドポイントのリストについては、「」の「Amazon Pinpoint API とサービスエンドポイントのエンドポイントとクォータ」を参照してくださいAmazon Web Services 全般のリファレンス。 Amazon Pinpoint AWS 各リージョンで利用できるアベイラビリティーゾーンの数の詳細については、「AWS グローバルインフラストラクチャ」を参照してください。

のセットアップ AWS アカウント

AWS エンドユーザーメッセージングプッシュを使用してアプリにプッシュ通知を送信する前に、まず十分な IAM アクセス許可 AWS アカウント を持つ を取得する必要があります。これは、 AWS エコシステム内の他のサービス AWS アカウント にも使用できます。

トピック

- にサインアップする AWS アカウント
- 管理アクセスを持つユーザーを作成する

にサインアップする AWS アカウント

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、電話またはテキストメッセージを受け取り、電話キーパッドで検証コードを入力します。

   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザー が作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティベストプラクティスとして、ユーザーに管理アクセス権を割り当て、ルートユーザーアクセスが必要なタスクの実行にはルートユーザーのみを使用するようにしてください。

AWS サインアッププロセスが完了すると、 から確認メールが送信されます。https://aws.amazon.com/ の [マイアカウント] をクリックして、いつでもアカウントの現在のアクティビティを表示し、アカウントを管理することができます。

管理アクセスを持つユーザーを作成する

にサインアップしたら AWS アカウント、日常的なタスクにルートユーザーを使用しないように AWS アカウントのルートユーザー、 を保護し AWS IAM アイデンティティセンター、 を有効にして管理ユーザーを作成します。

を保護する AWS アカウントのルートユーザー

1. ルートユーザーを選択し、 AWS アカウント E メールアドレスを入力して、アカウント所有者AWS マネジメントコンソールとして にサインインします。次のページでパスワードを入力します。

   ルートユーザーを使用してサインインする方法については、AWS サインイン ユーザーガイドのルートユーザーとしてサインインするを参照してください。

2. ルートユーザーの多要素認証 (MFA) を有効にします。

   手順については、IAM ユーザーガイドの AWS アカウント 「ルートユーザー (コンソール) の仮想 MFA デバイスを有効にする」を参照してください。

管理アクセスを持つユーザーを作成する

1. IAM アイデンティティセンターを有効にします。

   手順については、「AWS IAM アイデンティティセンター ユーザーガイド」の「AWS IAM アイデンティティセンターの有効化」を参照してください。

2. IAM アイデンティティセンターで、ユーザーに管理アクセスを付与します。

   を ID ソース IAM アイデンティティセンターディレクトリ として使用する方法のチュートリアルについては、「 AWS IAM アイデンティティセンター ユーザーガイド」の「デフォルトを使用してユーザーアクセスを設定する IAM アイデンティティセンターディレクトリ」を参照してください。

管理アクセス権を持つユーザーとしてサインインする

- IAM アイデンティティセンターのユーザーとしてサインインするには、IAM アイデンティティセンターのユーザーの作成時に E メールアドレスに送信されたサインイン URL を使用します。

IAM Identity Center ユーザーを使用してサインインする方法については、AWS サインイン「 ユーザーガイド」の AWS 「 アクセスポータルにサインインする」を参照してください。

追加のユーザーにアクセス権を割り当てる

1. IAM アイデンティティセンターで、最小特権のアクセス許可を適用するというベストプラクティスに従ったアクセス許可セットを作成します。

   手順については、「AWS IAM アイデンティティセンター ユーザーガイド」の「権限設定を作成する」を参照してください。
2. グループにユーザーを割り当て、そのグループにシングルサインオンアクセス権を割り当てます。

   手順については、「AWS IAM アイデンティティセンター ユーザーガイド」の「Add groups」を参照してください。

AWS エンドユーザーメッセージングプッシュの開始方法

AWS エンドユーザーメッセージングプッシュをセットアップしてアプリにプッシュ通知を送信できるようにするには、まず AWS エンドユーザーメッセージングプッシュがアプリにメッセージを送信することを許可する認証情報を指定する必要があります。提供する認証情報は、使用するプッシュ通知システムによって異なります。

- Apple Push Notification Service (APN) の認証情報については、Apple デベロッパードキュメントの「Apple から暗号化キーとキー ID を取得する」および「Apple からプロバイダー証明書を取得する」を参照してください。
- Firebase コンソールから取得できる Firebase Cloud Messaging (FCM) 認証情報については、「Firebase Cloud Messaging」を参照してください。
- Baidu 認証情報については、「Baidu」を参照してください。
- Amazon Device Messaging (ADM) 認証情報については、「認証情報の取得」を参照してください。

アプリケーションの作成とプッシュチャネルの有効化

AWS End User Messaging Push を使用してプッシュ通知を送信する前に、まずアプリケーションを作成し、プッシュ通知チャネルを有効にする必要があります。

コンテキスト

アプリケーション

アプリケーションは、すべての AWS エンドユーザーメッセージングプッシュ設定のストレージコンテナです。また、アプリケーションは Amazon Pinpoint のチャネル、キャンペーン、ジャーニーの設定も保存します。

キー

AWS エンドユーザーメッセージングプッシュが APNs 認証トークンに暗号で署名するために使用するプライベート署名キー。この署名キーは Apple 開発者アカウントから取得できます。

署名キーを指定すると、AWS エンドユーザーメッセージングプッシュはトークンを使用して、送信するプッシュ通知ごとに APNs で認証します。署名キーを使用すると、APNs 本番環境とサンドボックス環境に通知を送信できます。

証明書とは異なり、署名キーが期限切れになることはありません。1 回のみキーを指定すれば、後で更新する必要はありません。複数のアプリに対して同じ署名キーを使用できます。詳細については、『Xcode ヘルプ』の「Communicate with APNs using authentication tokens」を参照してください。

証明書

プッシュ通知を送信するときに AWS 、エンドユーザーメッセージングプッシュが APNs で認証するために使用する TLS 証明書。APNs 証明書は、本番環境とサンドボックス環境の両方をサポートできます。また、サンドボックス環境のみをサポートすることもできます。証明書は Apple 開発者アカウントから取得できます。

証明書は 1 年後に期限切れになります。この場合、新しい証明書を作成し、 AWS エンドユーザーメッセージングプッシュに提供してプッシュ通知配信を更新する必要があります。詳細については、『Xcode ヘルプ』の「TLS 証明書を使用した APNs との通信」を参照してください。

前提条件

プッシュチャネルを使用するには、プッシュサービスに有効な認証情報が必要です。認証情報の取得の詳細については、「」を参照してくださいAWS エンドユーザーメッセージングプッシュの開始方法。

手順

アプリケーションを作成し、プッシュチャネルのいずれかを有効にするには、次の手順に従います。この手順を完了するには、アプリケーション名を入力するだけで済みます。プッシュチャネルは、後で有効または無効にできます。

1. https://console.aws.amazon.com/push-notifications/ で AWS End User Messaging Push コンソールを開きます。
2. [Create application] を選択します。
3. アプリケーション名 にアプリケーションの名前を入力します。
4. （オプション) このオプションのステップに従って、Apple Push Notification Service (APNsを有効にします。

    a. Apple Push Notification Service (APNs) で Enable を選択します。

    b. デフォルトの認証タイプで、次のいずれかを選択します。

        i. キー認証情報を選択した場合は、Apple 開発者アカウントから次の情報を入力します。AWS エンドユーザーメッセージングプッシュでは、認証トークンを構築するためにこの情報が必要です。

        - [Key ID] – 署名キーに割り当てられた ID。
        - [Bundle identifier] – iOS アプリケーションに割り当てられた ID。
        - [Team identifier] – Apple デベロッパーアカウントチームに割り当てられた ID。
        - [Authentication key] – 認証キーを作成するときに Apple デベロッパーアカウントからダウンロードする .p8 ファイル。

        ii. [Certificate credentials] を選択した場合は、次の情報を入力します。

        - [SSL certificate] – TLS 証明書の .p12 ファイル。
        - Certificate password – 証明書にパスワードを指定している場合は、そのパスワードをここに入力します。

- [証明書タイプ] – 使用する証明書の種類を選択します。

5. （オプション) このオプションのステップに従って、Firebase Cloud Messaging (FCM) を有効にします。

   a. Firebase Cloud Messaging (FCM) で Enable を選択します。
   b. デフォルトの認証タイプで、次のいずれかを選択します。

      i. トークン認証情報 (推奨) では、ファイルを選択 を選択し、サービス JSON ファイルを選択します。
      ii. キー認証情報には、API キーにキーを入力します。

6. （オプション) このオプションのステップに従って、Baidu Cloud Push を有効にします。

   a. Baidu Cloud Push で Enable を選択します。
   b. API キーには、API キーを入力します。
   c. シークレットキーには、シークレットキーを入力します。

7. （オプション) このオプションのステップに従って、Amazon Device Messaging を有効にします。

   a. Amazon Device Messaging で、Enable を選択します。
   b. クライアント ID には、クライアント ID を入力します。
   c. クライアントシークレットには、クライアントシークレットを入力します。

8. [Create application] を選択します。

プッシュチャネルの無効化

プッシュチャネルを無効にするには、次の手順に従ってください。

1. https://console.aws.amazon.com/push-notifications/ で AWS End User Messaging Push コンソールを開きます。
2. プッシュ認証情報を含むアプリケーションを選択します。
3. （オプション) Apple Push Notification Service (APNs) の場合は、Enableをクリアします。
4. （オプション) Firebase Cloud Messaging (FCM) の場合は、Enable をクリアします。
5. （オプション) Baidu Cloud Push clear Enable の場合。
6. （オプション) Amazon Device Messaging の場合、有効化をクリアします。
7. [Save changes] (変更の保存) をクリックします。

メッセージを送信する

AWS エンドユーザーメッセージングプッシュ API は、トランザクションプッシュ通知を特定のデバイス識別子に送信できます。このセクションでは、 AWS SDK を使用して AWS エンドユーザーメッセージングプッシュ API を介してプッシュ通知を送信するために使用できる完全なコード例を示します。

これらの例を使用して、 AWS エンドユーザーメッセージングプッシュがサポートするプッシュ通知サービスを介してプッシュ通知を送信できます。現在、 AWS エンドユーザーメッセージングプッシュは、Firebase Cloud Messaging (FCM)、Apple Push Notification Service (APNs)、Baidu Cloud Push、Amazon Device Messaging (ADM) のチャネルをサポートしています。

エンドポイント、セグメント、チャネルのコード例の詳細については、「コード例」を参照してください。

> **Note**
>
> Firebase Cloud Messaging (FCM) サービスを介してプッシュ通知を送信する場合は、 AWS エンドユーザーメッセージングプッシュ API への呼び出しGCMでサービス名を使用します。Google Cloud Messaging (GCM) サービスは、2018 年 4 月 10 日に Google によって停止されました。ただし、 AWS エンドユーザーメッセージングプッシュ API は、GCM GCMサービスの中止前に記述された API コードとの互換性を維持するために、FCM サービスを介して送信するメッセージにサービス名を使用します。

GCM (AWS CLI)

次の例では、send-messages を使用して で GCM プッシュ通知を送信します AWS CLI。token をデバイスの一意のトークンに置き換え、*611e3e3cdd47474c9c1399a50example* をアプリケーション識別子に置き換えます。

```
aws pinpoint send-messages \
--application-id 611e3e3cdd47474c9c1399a50example \
--message-request file://myfile.json \
--region us-west-2

Contents of myfile.json:
{
```

```
  "Addresses": {
    "token": {
      "ChannelType" : 'GCM'
    }
  },
  "MessageConfiguration": {
    "GCMMessage": {
      "Action": "URL",
      "Body": "This is a sample message",
      "Priority": "normal",
      "SilentPush": True,
      "Title": "My sample message",
      "TimeToLive": 30,
      "Url": "https://www.example.com"
      }
  }
}
```

次の例では、send-messages を使用して、 ですべてのレガシーキーを使用して GCM プッシュ通知を送信します AWS CLI。token をデバイスの一意のトークンに置き換え、*611e3e3cdd47474c9c1399a50example* をアプリケーション識別子に置き換えます。

```
aws pinpoint send-messages \
--application-id 611e3e3cdd47474c9c1399a50example \
--message-request
'{
  "MessageConfiguration": {
    "GCMMessage":{
      "RawContent": "{\"notification\": {\n \"title\": \"string\",\n \"body\":
 \"string\",\n \"android_channel_id\": \"string\",\n \"body_loc_args\": [\n \"string
\"\n ],\n \"body_loc_key\": \"string\",\n \"click_action\": \"string\",\n \"color\":
 \"string\",\n \"icon\": \"string\",\n \"sound\": \"string\",\n \"tag\": \"string
\",\n \"title_loc_args\": [\n \"string\"\n ],\n \"title_loc_key\": \"string\"\n },
\"data\":{\"message\":\"hello in data\"} }",
      "TimeToLive" : 309744
     }
   },
  "Addresses": {
    "token": {
      "ChannelType":"GCM"
      }
   }
}'
```

```
\ --region us-east-1
```

次の例では、send-messages を使用して、 を使用して FCMv1 メッセージペイロードで GCM プッシュ通知を送信します AWS CLI。token をデバイスの一意のトークンに置き換え、*611e3e3cdd47474c9c1399a50example* をアプリケーション識別子に置き換えます。

```
aws pinpoint send-messages \
--application-id 6a2dafd84bec449ea75fb773f4c41fa1 \
--message-request
'{
  "MessageConfiguration": {
    "GCMMessage":{
      "RawContent": "{\n \"fcmV1Message\": \n {\n \"message\" :{\n  \"notification
\": {\n \"title\": \"string\",\n \"body\": \"string\"\n },\n \"android\": {\n
 \"priority\": \"high\",\n \"notification\": {\n \"title\": \"string\",\n \"body
\": \"string\",\n \"icon\": \"string\",\n \"color\": \"string\",\n \"sound\":
 \"string\",\n \"tag\": \"string\",\n \"click_action\": \"string\",\n \"body_loc_key
\": \"string\",\n \"body_loc_args\": [\n \"string\"\n ],\n \"title_loc_key
\": \"string\",\n \"title_loc_args\": [\n \"string\"\n ],\n \"channel_id\":
 \"string\",\n \"ticker\": \"string\",\n \"sticky\": true,\n \"event_time\":
 \"2024-02-06T22:11:55Z\",\n \"local_only\": true,\n \"notification_priority\":
 \"PRIORITY_UNSPECIFIED\",\n \"default_sound\": false,\n \"default_vibrate_timings
\": true,\n \"default_light_settings\": false,\n \"vibrate_timings\": [\n \"22s
\"\n ],\n \"visibility\": \"VISIBILITY_UNSPECIFIED\",\n \"notification_count\": 5,
\n \"light_settings\": {\n \"color\": {\n \"red\": 1,\n \"green\": 2,\n \"blue\":
 3,\n \"alpha\": 6\n },\n \"light_on_duration\": \"112s\",\n \"light_off_duration
\": \"1123s\"\n },\n \"image\": \"string\"\n },\n \"data\": {\n \"dataKey1\":
 \"priority message\",\n \"data_key_3\": \"priority message\",\n \"dataKey2\":
 \"priority message\",\n \"data_key_5\": \"priority message\"\n },\n \"ttl\":
 \"10023.32s\"\n },\n \"apns\": {\n \"payload\": {\n \"aps\": {\n \"alert\": {\n
 \"subtitle\": \"string\",\n \"title-loc-args\": [\n \"string\"\n ],\n \"title-loc-
key\": \"string\",\n \"launch-image\": \"string\",\n \"subtitle-loc-key\": \"string
\",\n \"subtitle-loc-args\": [\n \"string\"\n ],\n \"loc-args\": [\n \"string
\"\n ],\n \"loc-key\": \"string\",\n \"title\": \"string\",\n \"body\": \"string
\"\n },\n \"thread-id\": \"string\",\n \"category\": \"string\",\n \"content-
available\": 1,\n \"mutable-content\": 1,\n \"target-content-id\": \"string\",\n
 \"interruption-level\": \"string\",\n \"relevance-score\": 25,\n \"filter-criteria
\": \"string\",\n \"stale-date\": 6483,\n \"content-state\": {},\n \"timestamp\":
 673634,\n \"dismissal-date\": 4,\n \"attributes-type\": \"string\",\n \"attributes
\": {},\n \"sound\": \"string\",\n \"badge\": 5\n }\n }\n },\n \"webpush\": {\n
 \"notification\": {\n \"permission\": \"granted\",\n \"maxActions\": 2,\n \"actions
\": [\n \"title\"\n ],\n \"badge\": \"URL\",\n \"body\": \"Hello\",\n \"data\": {\n
 \"hello\": \"hey\"\n },\n \"dir\": \"auto\",\n \"icon\": \"icon\",\n \"image\":
```

```
 \"image\",\n \"lang\": \"string\",\n \"renotify\": false,\n \"requireInteraction\":
 true,\n \"silent\": false,\n \"tag\": \"tag\",\n \"timestamp\": 1707259524964,\n
 \"title\": \"hello\",\n \"vibrate\": [\n 100,\n 200,\n 300\n ]\n },\n \"data\": {\n
 \"data1\": \"priority message\",\n \"data2\": \"priority message\",\n \"data12\":
 \"priority message\",\n \"data3\": \"priority message\"\n }\n },\n \"data\": {\n
 \"data7\": \"priority message\",\n \"data5\": \"priority message\",\n \"data8\":
 \"priority message\",\n \"data9\": \"priority message\"\n }\n }\n \n}\n}",
      "TimeToLive" : 309744
    }
  },
  "Addresses": {
    token: {
      "ChannelType":"GCM"
    }
   }
}'
\ --region us-east-1
```

GCM に ImageUrlフィールドを使用する場合、pinpoint はキーが であるフィールドをデータ通知として送信します。これによりpinpoint.notification.imageUrl、イメージがすぐにレンダリングされない場合があります。RawContent を使用するか、アプリとの統合などのデータキーの処理を追加してください AWS Amplify。

Safari (AWS CLI)

AWS エンドユーザーメッセージングプッシュを使用して、Apple の Safari ウェブブラウザを使用する macOS コンピュータにメッセージを送信できます。Safari ブラウザにメッセージを送信するには、Raw メッセージの内容を指定し、メッセージのペイロードに特定の属性を含める必要があります。これを行うには、Amazon Pinpoint ユーザーガイドの raw メッセージペイロードを使用してプッシュ通知テンプレートを作成するか、キャンペーンメッセージで raw メッセージの内容を直接指定します。

> **Note**
>
> この特別な属性は、Safari ウェブブラウザを使用する macOS ラップトップおよびデスクトップコンピュータに送信するために必要です。iPhone や iPad などの iOS デバイスへの送信には必要ありません。

Safari ウェブブラウザにメッセージを送信するには、Raw メッセージペイロードを指定する必要があります。Raw メッセージのペイロードは、aps オブジェクト内に url-args 配列を含む必

要があります。url-args 配列は、Safariウェブブラウザにプッシュ通知を送信するために必要です。ただし、配列に空の要素が 1 つ含まれていてもかまいません。

次の例では、send-messages を使用して、 を使用して Safari ウェブブラウザに通知を送信します AWS CLI。token をデバイスの一意のトークンに置き換え、*611e3e3cdd47474c9c1399a50example* をアプリケーション識別子に置き換えます。

```
aws pinpoint send-messages \
--application-id 611e3e3cdd47474c9c1399a50example \
--message-request
'{
  "Addresses": {
    "token":
    {
      "ChannelType":"APNS"
    }
  },
  "MessageConfiguration": {
    "APNSMessage": {
        "RawContent":
          "{\"aps\": {\"alert\": { \"title\": \"Title of my message\", \"body\":
 \"This is a push notification for the Safari web browser.\"},\"content-available\":
 1,\"url-args\": [\"\"]}}"
     }
  }
}'
\ --region us-east-1
```

Safari のプッシュ通知について詳しくは、『Apple デベロッパーウェブサイト』の「Configuring Safari Push Notifications」をご覧ください。

APNS (AWS CLI)

次の例では、send-messages を使用して、 で APNS プッシュ通知を送信します AWS CLI。token をデバイスの一意のトークンに、*611e3e3cdd47474c9c1399a50example* をアプリケーション識別子に、*GAME_INVITATION* を一意の識別子に置き換えます。

```
aws pinpoint send-messages \
--application-id 611e3e3cdd47474c9c1399a50example \
--message-request
'{
    "Addresses": {
     "token":
```

```
    {
      "ChannelType":"APNS"
    }
  },
  "MessageConfiguration": {
    "APNSMessage": {
      "RawContent": "{\"aps\" : {\"alert\" : {\"title\" : \"Game Request\",
\"subtitle\" : \"Five Card Draw\",\"body\" : \"Bob wants to play poker\"},\"category
\" : \"GAME_INVITATION\"},\"gameID\" : \"12345678\"}"
      }
    }
}'
\ --region us-east-1
```

JavaScript (Node.js)

この例を使用して、Node.js で AWS SDK for JavaScript を使用してプッシュ通知を送信します。この例は、SDK for JavaScript in Node.js がすでにインストールされ、設定されていることを前提としています。

この例では、共有認証情報ファイルを使用して、既存の ユーザーのアクセスキーとシークレットアクセスキーを指定するものと想定しています。詳細については、『AWS SDK for JavaScript in Node.js デベロッパーガイド』の「認証情報の設定」を参照してください。

```
'use strict';

const AWS = require('aws-sdk');

// The AWS Region that you want to use to send the message. For a list of
// AWS Regions where the API is available
const region = 'us-east-1';

// The title that appears at the top of the push notification.
var title = 'Test message sent from End User Messaging Push.';

// The content of the push notification.
var message = 'This is a sample message sent from End User Messaging Push by using
 the '
            + 'AWS SDK for JavaScript in Node.js';

// The application ID that you want to use when you send this
// message. Make sure that the push channel is enabled for the project that
// you choose.
```

```
var applicationId = 'ce796be37f32f178af652b26eexample';

// An object that contains the unique token of the device that you want to send
// the message to, and the push service that you want to use to send the message.
var recipient = {
  'token': 'a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d8e9f0',
  'service': 'GCM'
  };

// The action that should occur when the recipient taps the message. Possible
// values are OPEN_APP (opens the app or brings it to the foreground),
// DEEP_LINK (opens the app to a specific page or interface), or URL (opens a
// specific URL in the device's web browser.)
var action = 'URL';

// This value is only required if you use the URL action. This variable contains
// the URL that opens in the recipient's web browser.
var url = 'https://www.example.com';

// The priority of the push notification. If the value is 'normal', then the
// delivery of the message is optimized for battery usage on the recipient's
// device, and could be delayed. If the value is 'high', then the notification is
// sent immediately, and might wake a sleeping device.
var priority = 'normal';

// The amount of time, in seconds, that the push notification service provider
// (such as FCM or APNS) should attempt to deliver the message before dropping
// it. Not all providers allow you specify a TTL value.
var ttl = 30;

// Boolean that specifies whether the notification is sent as a silent
// notification (a notification that doesn't display on the recipient's device).
var silent = false;

function CreateMessageRequest() {
  var token = recipient['token'];
  var service = recipient['service'];
  if (service == 'GCM') {
    var messageRequest = {
      'Addresses': {
        [token]: {
          'ChannelType' : 'GCM'
        }
      },
```

```
      'MessageConfiguration': {
        'GCMMessage': {
          'Action': action,
          'Body': message,
          'Priority': priority,
          'SilentPush': silent,
          'Title': title,
          'TimeToLive': ttl,
          'Url': url
        }
      }
    };
  } else if (service == 'APNS') {
    var messageRequest = {
      'Addresses': {
        [token]: {
          'ChannelType' : 'APNS'
        }
      },
      'MessageConfiguration': {
        'APNSMessage': {
          'Action': action,
          'Body': message,
          'Priority': priority,
          'SilentPush': silent,
          'Title': title,
          'TimeToLive': ttl,
          'Url': url
        }
      }
    };
  } else if (service == 'BAIDU') {
    var messageRequest = {
      'Addresses': {
        [token]: {
          'ChannelType' : 'BAIDU'
        }
      },
      'MessageConfiguration': {
        'BaiduMessage': {
          'Action': action,
          'Body': message,
          'SilentPush': silent,
          'Title': title,
```

```
          'TimeToLive': ttl,
          'Url': url
        }
      }
    };
  } else if (service == 'ADM') {
    var messageRequest = {
      'Addresses': {
        [token]: {
          'ChannelType' : 'ADM'
        }
      },
      'MessageConfiguration': {
        'ADMMessage': {
          'Action': action,
          'Body': message,
          'SilentPush': silent,
          'Title': title,
          'Url': url
        }
      }
    };
  }

  return messageRequest
}

function ShowOutput(data){
  if (data["MessageResponse"]["Result"][recipient["token"]]["DeliveryStatus"]
      == "SUCCESSFUL") {
    var status = "Message sent! Response information: ";
  } else {
    var status = "The message wasn't sent. Response information: ";
  }
  console.log(status);
  console.dir(data, { depth: null });
}

function SendMessage() {
  var token = recipient['token'];
  var service = recipient['service'];
  var messageRequest = CreateMessageRequest();

  // Specify that you're using a shared credentials file, and specify the
```

```
  // IAM profile to use.
  var credentials = new AWS.SharedIniFileCredentials({ profile: 'default' });
  AWS.config.credentials = credentials;

  // Specify the AWS Region to use.
  AWS.config.update({ region: region });

  //Create a new Pinpoint object.
  var pinpoint = new AWS.Pinpoint();
  var params = {
    "ApplicationId": applicationId,
    "MessageRequest": messageRequest
  };

  // Try to send the message.
  pinpoint.sendMessages(params, function(err, data) {
    if (err) console.log(err);
    else     ShowOutput(data);
  });
}

SendMessage()
```

Python

AWS SDK for Python (Boto3)を使用してプッシュ通知を送信するには、この例を使用します。この例は、SDK for Python (Boto3) がすでにインストールされ、設定されていることを前提としています。

この例では、共有認証情報ファイルを使用して、既存の ユーザーのアクセスキーとシークレットアクセスキーを指定するものと想定しています。詳細については、『AWS SDK for Python (Boto3) API Reference』の「認証情報」を参照してください。

```
import json
import boto3
from botocore.exceptions import ClientError

# The AWS Region that you want to use to send the message. For a list of
# AWS Regions where the API is available
region = "us-east-1"

# The title that appears at the top of the push notification.
title = "Test message sent from End User Messaging Push."
```

```
# The content of the push notification.
message = ("This is a sample message sent from End User Messaging Push by using the
 "
           "AWS SDK for Python (Boto3).")

# The application ID to use when you send this message.
# Make sure that the push channel is enabled for the project or application
# that you choose.
application_id = "ce796be37f32f178af652b26eexample"

# A dictionary that contains the unique token of the device that you want to send
 the
# message to, and the push service that you want to use to send the message.
recipient = {
    "token": "a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d8e9f0",
    "service": "GCM"
    }

# The action that should occur when the recipient taps the message. Possible
# values are OPEN_APP (opens the app or brings it to the foreground),
# DEEP_LINK (opens the app to a specific page or interface), or URL (opens a
# specific URL in the device's web browser.)
action = "URL"

# This value is only required if you use the URL action. This variable contains
# the URL that opens in the recipient's web browser.
url = "https://www.example.com"

# The priority of the push notification. If the value is 'normal', then the
# delivery of the message is optimized for battery usage on the recipient's
# device, and could be delayed. If the value is 'high', then the notification is
# sent immediately, and might wake a sleeping device.
priority = "normal"

# The amount of time, in seconds, that the push notification service provider
# (such as FCM or APNS) should attempt to deliver the message before dropping
# it. Not all providers allow you specify a TTL value.
ttl = 30

# Boolean that specifies whether the notification is sent as a silent
# notification (a notification that doesn't display on the recipient's device).
silent = False
```

```
# Set the MessageType based on the values in the recipient variable.
def create_message_request():

    token = recipient["token"]
    service = recipient["service"]

    if service == "GCM":
        message_request = {
            'Addresses': {
                token: {
                    'ChannelType': 'GCM'
                }
            },
            'MessageConfiguration': {
                'GCMMessage': {
                    'Action': action,
                    'Body': message,
                    'Priority' : priority,
                    'SilentPush': silent,
                    'Title': title,
                    'TimeToLive': ttl,
                    'Url': url
                }
            }
        }
    elif service == "APNS":
        message_request = {
            'Addresses': {
                token: {
                    'ChannelType': 'APNS'
                }
            },
            'MessageConfiguration': {
                'APNSMessage': {
                    'Action': action,
                    'Body': message,
                    'Priority' : priority,
                    'SilentPush': silent,
                    'Title': title,
                    'TimeToLive': ttl,
                    'Url': url
                }
            }
        }
```

```
    elif service == "BAIDU":
        message_request = {
            'Addresses': {
                token: {
                    'ChannelType': 'BAIDU'
                }
            },
            'MessageConfiguration': {
                'BaiduMessage': {
                    'Action': action,
                    'Body': message,
                    'SilentPush': silent,
                    'Title': title,
                    'TimeToLive': ttl,
                'Url': url
                }
            }
        }
    elif service == "ADM":
        message_request = {
            'Addresses': {
                token: {
                    'ChannelType': 'ADM'
                }
            },
            'MessageConfiguration': {
                'ADMMessage': {
                    'Action': action,
                    'Body': message,
                    'SilentPush': silent,
                    'Title': title,
                    'Url': url
                }
            }
        }
    else:
        message_request = None

    return message_request

# Show a success or failure message, and provide the response from the API.
def show_output(response):
    if response['MessageResponse']['Result'][recipient["token"]]['DeliveryStatus']
 == "SUCCESSFUL":
```

```
        status = "Message sent! Response information:\n"
    else:
        status = "The message wasn't sent. Response information:\n"
    print(status, json.dumps(response,indent=4))

# Send the message through the appropriate channel.
def send_message():

    token = recipient["token"]
    service = recipient["service"]
    message_request = create_message_request()

    client = boto3.client('pinpoint',region_name=region)

    try:
        response = client.send_messages(
            ApplicationId=application_id,
            MessageRequest=message_request
        )
    except ClientError as e:
        print(e.response['Error']['Message'])
    else:
        show_output(response)

send_message()
```

その他のリソース

- プッシュチャネルテンプレートの詳細については、Amazon Pinpoint ユーザーガイド」の「プッシュ通知テンプレートの作成」を参照してください。

アプリケーションでのプッシュ通知の受信

以下のトピックでは、Swift、Android、React Native、または Flutter アプリを変更してプッシュ通知を受信する方法について説明します。

トピック

Swift プッシュ通知の設定

iOS アプリのプッシュ通知は Apple Push Notification Service (APNs) を使用して送信されます。iOS デバイスにプッシュ通知を送信するには、Apple 開発者ポータルでアプリ ID を作成する必要があり、必要な証明書を作成する必要があります。これらのステップの完了の詳細については、Amplify ドキュメントの「プッシュ通知サービスの設定 AWS 」を参照してください。

APNs トークンの使用

ベストプラクティスとして、アプリケーションの再インストール時に顧客のデバイストークンが再生成されるようにアプリケーションを開発する必要があります。

受信者がデバイスを新しいメジャーバージョンの iOS (iOS 12 から iOS 13 など) にアップグレードし、後でアプリを再インストールした場合、アプリケーションにより新しいトークンが生成されます。アプリケーションによりトークンが更新されない場合、古いトークンを使用して通知が送信されます。その結果、トークンが無効になったため、Apple Push Notification Service (APNs) は通知を拒否します。通知を送信しようとすると、APNs からメッセージ失敗通知を受け取ります。

Android プッシュ通知のセットアップ

Android アプリケーションのプッシュ通知は、Google Cloud Messaging (GCM) の代わりに Firebase Cloud Messaging (FCM) を使用して送信されます。Android デバイスにプッシュ通知を送信する前

に、FCM 認証情報を取得する必要があります。その後それらの認証情報により、Android プロジェクトを作成し、プッシュ通知を受け取るサンプルアプリを起動することができます。これらのステップの完了の詳細については、Amplify ドキュメントの「プッシュ通知 AWS 」セクションを参照してください。

Flutter プッシュ通知のセットアップ

Flutter アプリケーションのプッシュ通知は、Android の場合は Firebase Cloud Messaging (FCM)、iOS の場合は APN を使用して送信されます。これらのステップを完了する方法の詳細については、AWS Amplify Flutter ドキュメントの「Push notifications」のセクションを参照してください。

React Native プッシュ通知のセットアップ

React Native アプリケーションのプッシュ通知は、Android の場合は Firebase Cloud Messaging (FCM)、iOS の場合は APN を使用して送信されます。これらのステップを完了する方法の詳細については、AWS Amplify JavaScript ドキュメントの「Push notifications」のセクションを参照してください。

AWS エンドユーザーメッセージングプッシュでアプリケーションを作成する

AWS エンドユーザーメッセージングプッシュでプッシュ通知の送信を開始するには、アプリケーションを作成する必要があります。次に、適切な認証情報を入力して、使用するプッシュ通知チャンネルを有効にする必要があります。

AWS エンドユーザーメッセージングプッシュコンソールを使用して、新しいアプリケーションを作成し、プッシュ通知チャネルを設定できます。詳細については、「アプリケーションの作成とプッシュチャネルの有効化」を参照してください。

API、 AWS SDK、または AWS Command Line Interface () を使用してアプリケーションを作成およびセットアップすることもできますAWS CLI。アプリケーションを作成するには、 Appsリソースを使用します。プッシュ通知チャンネルを設定するには、次のリソースを使用してください。

- Apple Push Notification Service を利用して、iOSデバイスのユーザーにメッセージを送信するための APNs チャンネルです。
- Amazon Kindle Fire デバイスのユーザーにメッセージを送信する ADM チャンネル。

- Baidu ユーザーにメッセージを送信する Baidu チャンネル。
- Google Cloud Messaging (GCM) に代わるFirebase Cloud Messaging (FCM) を利用して、Android 端末にメッセージを送信する GCM チャンネルです。

プッシュ通知の処理

プッシュ通知の送信に必要な認証情報を取得したら、プッシュ通知を受信できるようにアプリケーションを更新できます。詳細については、 AWS Amplify ドキュメントの「プッシュ通知 — 開始方法」を参照してください。

アプリケーションの削除

この手順では、アカウントとアプリケーション内のすべてのリソースからアプリケーションを削除します。

コンテキスト

アプリケーション

アプリケーションは、すべての AWS エンドユーザーメッセージングプッシュ設定のストレージコンテナです。また、アプリケーションは Amazon Pinpoint のチャネル、キャンペーン、ジャーニーの設定も保存します。

手順

1. https://console.aws.amazon.com/push-notifications/ で AWS End User Messaging Push コンソールを開きます。
2. アプリケーションを選択し、削除を選択します。
3. アプリケーションの削除ウィンドウで「」と入力し**delete**、「削除」を選択します。

> **Important**
>
> Amazon Pinpoint のチャネル、キャンペーン、ジャーニー、セグメントもすべて削除されます。

ベストプラクティス

お客様の利益を最優先にしておりますが、メッセージの配信性能に影響するような状況が発生する場合があります。次のセクションでは、プッシュメッセージを目的のユーザーに確実に届けるための推奨事項について説明します。

大量のプッシュ通知を送信する

大量のプッシュ通知を送信する前に、スループット要件をサポートするようにアカウントが設定されていることを確認してください。デフォルトでは、すべてのアカウントは 1 秒あたり 25,000 件のメッセージを送信するように設定されています。1 秒間に 25,000 通以上のメッセージを送信できるようにする必要がある場合は、クォータの増加をリクエストすることができます。詳細については、「AWS エンドユーザーメッセージングプッシュのクォータ」を参照してください。

FCM や APNs など、使用する各プッシュ通知プロバイダーの認証情報でアカウントが正しく設定されていることを確認します。

最後に、例外を処理する方法を検討します。プッシュ通知サービスごとに、異なる例外メッセージが用意されています。トランザクション送信の場合、API コールのメインのステータスコード 200 を受け取り、メッセージ送信中に対応するプラットフォームトークン (FCM など) または証明書 (APNs など) が無効と判断されるとエンドポイントごとに永続エラーのステータスコード 400 を受け取ります。

AWS エンドユーザーメッセージングプッシュのセキュリティ

でのクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャからメリットを得られます。

セキュリティは、 AWS とお客様の間の責任共有です。責任共有モデルではこれをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ – AWS は、 で AWS サービスを実行するインフラストラクチャを保護する責任を担います AWS クラウド。 は、お客様が安全に使用できるサービス AWS も提供します。サードパーティーの監査者は、AWS コンプライアンスプログラムコンプライアンスプログラムの一環として、当社のセキュリティの有効性を定期的にテストおよび検証。 AWS エンドユーザーメッセージングプッシュに適用されるコンプライアンスプログラムの詳細については、「コンプライアンスプログラムAWS による対象範囲内のサービスコンプライアンスプログラム」を参照してください。
- クラウド内のセキュリティ – お客様の責任は、使用する AWS サービスによって決まります。また、ユーザーは、データの機密性、会社の要件、適用される法律や規制など、その他の要因についても責任を負います。

このドキュメントは、 AWS エンドユーザーメッセージングプッシュを使用する際に責任共有モデルを適用する方法を理解するのに役立ちます。以下のトピックでは、セキュリティおよびコンプライアンスの目的を達成するように AWS エンドユーザーメッセージングプッシュを設定する方法について説明します。また、 AWS エンドユーザーメッセージングプッシュリソースのモニタリングや保護に役立つ他の AWS サービスの使用方法についても説明します。

トピック

- セキュリティのベストプラクティス

AWS エンドユーザーメッセージングプッシュでのデータ保護

責任 AWS 共有モデル、 AWS エンドユーザーメッセージングプッシュのデータ保護に適用されます。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。ユーザーは、このインフラストラクチャでホストされるコンテンツに対する管理を維持する責任があります。また、使用する「 AWS のサービス 」のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、データプライバシーに関するよくある質問を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログに投稿された「AWS 責任共有モデルおよび GDPR」のブログ記事を参照してください。

データ保護の目的で、認証情報を保護し AWS アカウント 、 AWS IAM アイデンティティセンターまたは AWS Identity and Access Management (IAM) を使用して個々のユーザーを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須ですが、TLS 1.3 を推奨します。
- で API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。CloudTrail 証跡を使用して AWS アクティビティをキャプチャする方法については、「 AWS CloudTrail ユーザーガイド」のCloudTrail 証跡の使用」を参照してください。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度な管理されたセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API AWS を介して にアクセスするときに FIPS 140-3 検証済み暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-3」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報を、タグ、または [名前] フィールドなどの自由形式のテキストフィールドに含めないことを強くお勧めします。これは、コンソール、API、または AWS CLI SDK を使用して AWS エンドユーザーメッセージングプッシュまたは他の AWS のサービス を使用する場合も同様です。 AWS SDKs タグ、または名前に使用される自由記述のテキスト

フィールドに入力したデータは、請求または診断ログに使用される場合があります。外部サーバーに URL を提供する場合、そのサーバーへのリクエストを検証できるように、認証情報を URL に含めないことを強くお勧めします。

データ暗号化

AWS エンドユーザーメッセージングプッシュデータは、転送中および保管中に暗号化されます。AWS エンドユーザーメッセージングプッシュにデータを送信すると、データは受信および保存時に暗号化されます。 AWS エンドユーザーメッセージングプッシュからデータを取得すると、現在のセキュリティプロトコルを使用してデータが送信されます。

保管中の暗号化

AWS エンドユーザーメッセージングプッシュは、保存するすべてのデータを暗号化します。これには、設定データ、ユーザーおよびエンドポイントデータ、分析データ、および AWS エンドユーザーメッセージングプッシュに追加またはインポートするデータが含まれます。データを暗号化するために、 AWS End User Messaging Push は、サービスがユーザーに代わって所有および維持する internal AWS Key Management Service (AWS KMS) キーを使用します。これらのキーは定期的に更新されます。詳細については AWS KMS、「 AWS Key Management Service デベロッパーガイド」を参照してください。

転送中の暗号化

AWS エンドユーザーメッセージングプッシュは、HTTPS および Transport Layer Security (TLS) 1.2 以降を使用してクライアントやアプリケーションと通信します。他の AWS サービスと通信するために、 AWS エンドユーザーメッセージングプッシュは HTTPS と TLS 1.2 を使用します。さらに、コンソール、 AWS SDK、または を使用して AWS エンドユーザーメッセージングプッシュリソースを作成および管理する場合 AWS Command Line Interface、すべての通信は HTTPS および TLS 1.2 を使用して保護されます。

キー管理

AWS エンドユーザーメッセージングプッシュデータを暗号化するために、 AWS エンドユーザーメッセージングプッシュは、サービスがユーザーに代わって所有および維持する内部 AWS KMS キーを使用します。これらのキーは定期的に更新されます。独自のキー AWS KMS や他のキーをプロビジョニングして使用して、 AWS エンドユーザーメッセージングプッシュに保存するデータを暗号化することはできません。

ネットワーク間トラフィックのプライバシー

インターネットワークトラフィックのプライバシーとは、 AWS エンドユーザーメッセージングプッシュとオンプレミスのクライアントとアプリケーション間、および AWS エンドユーザーメッセージングプッシュと同じ AWS リージョン内の他の AWS リソース間の接続とトラフィックを保護することです。以下の機能とプラクティスは、 AWS エンドユーザーメッセージングプッシュのインターネットネットワークトラフィックのプライバシーを確保するのに役立ちます。

AWS エンドユーザーメッセージングプッシュとオンプレミスクライアントおよびアプリケーション間のトラフィック

AWS エンドユーザーメッセージングプッシュとオンプレミスネットワーク上のクライアントおよびアプリケーションとの間にプライベート接続を確立するには、 を使用できます Direct Connect。これにより、標準の光ファイバーイーサネットケーブルを使用して、ネットワークを AWS Direct Connect ロケーションにリンクできます。ケーブルの一端はユーザーのルーターに接続します。もう 1 つのエンドは Direct Connect ルーターに接続されています。詳細については、「 Direct Connect ユーザーガイド」の「What is Direct Connect ?」(とは？) を参照してください。

公開された APIs を介した AWS エンドユーザーメッセージングプッシュへのアクセスを保護するために、API コールの AWS エンドユーザーメッセージングプッシュ要件に準拠することをお勧めします。 AWS エンドユーザーメッセージングプッシュでは、クライアントが Transport Layer Security (TLS) 1.2 以降を使用する必要があります。また、クライアントは、Ephemeral Diffie-Hellman (DHE) や Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) などの Perfect Forward Secrecy (PFS) を使用した暗号スイートもサポートしている必要があります。モードは、Java 7 以降など、最近のほとんどのシステムでサポートされています。

さらに、リクエストは、 AWS アカウントの AWS Identity and Access Management (IAM) プリンシパルに関連付けられているアクセスキー ID とシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

AWS エンドユーザーメッセージングプッシュと他の AWS リソース間のトラフィック

AWS エンドユーザーメッセージングプッシュと同じ AWS リージョン内の他の AWS リソース間の通信を保護するために、 AWS エンドユーザーメッセージングプッシュはデフォルトで HTTPS と TLS 1.2 を使用します。

AWS エンドユーザーメッセージングプッシュの Identity and Access Management

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインイン) し、誰に AWS エンドユーザーメッセージングプッシュリソースの使用を許可する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで使用できる AWS のサービス です。

トピック

オーディエンス

AWS Identity and Access Management (IAM) の使用方法は、ロールによって異なります。

- サービスユーザー - 機能にアクセスできない場合は、管理者にアクセス許可をリクエストします (「AWS エンドユーザーメッセージングプッシュのアイデンティティとアクセスのトラブルシューティング」を参照)。
- サービス管理者 - ユーザーアクセスを決定し、アクセス許可リクエストを送信します (「AWS エンドユーザーメッセージングプッシュと IAM の連携方法」を参照)
- IAM 管理者 - アクセスを管理するためのポリシーを作成します (「AWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例」を参照)

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。、IAM ユーザー AWS アカウントのルートユーザー、または IAM ロールを引き受けることで認証される必要があります。

(AWS IAM アイデンティティセンター IAM Identity Center)、シングルサインオン認証、Google/Facebook 認証情報などの ID ソースからの認証情報を使用して、フェデレーティッド ID としてサインインできます。サインインの詳細については、「AWS サインイン ユーザーガイド」の「AWS アカウントにサインインする方法」を参照してください。

プログラムによるアクセスの場合、 は SDK と CLI AWS を提供してリクエストを暗号化して署名します。詳細については、「IAM ユーザーガイド」の「API リクエストに対するAWS 署名バージョン 4」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、まず、すべての AWS のサービス および リソースへの完全なアクセス権を持つ AWS アカウント root ユーザーと呼ばれる 1 つのサインインアイデンティティから始めます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザー認証情報を必要とするタスクについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

フェデレーテッドアイデンティティ

ベストプラクティスとして、人間のユーザーが一時的な認証情報 AWS のサービス を使用して にアクセスするには、ID プロバイダーとのフェデレーションを使用する必要があります。

フェデレーティッド ID は、エンタープライズディレクトリ、ウェブ ID プロバイダー、または ID Directory Service ソースからの認証情報 AWS のサービス を使用して にアクセスするユーザーです。フェデレーテッドアイデンティティは、一時的な認証情報を提供するロールを引き受けます。

アクセスを一元管理する場合は、 AWS IAM アイデンティティセンターをお勧めします。詳細については、「AWS IAM アイデンティティセンター ユーザーガイド」の「IAM アイデンティティセンターとは」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、特定の個人やアプリケーションに対する特定のアクセス許可を持つアイデンティティです。長期認証情報を持つ IAM ユーザーの代わりに一時的な認証情報を使用することをお勧めします。詳細については、IAM ユーザーガイドの「ID プロバイダーとのフェデレーションを使用して にアクセスする必要がある AWS」を参照してください。

IAM グループは、IAM ユーザーの集合を指定し、大量のユーザーに対するアクセス許可の管理を容易にします。詳細については、「IAM ユーザーガイド」の「IAM ユーザーに関するユースケース」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可を持つアイデンティであり、一時的な認証情報を提供します。ユーザーから IAM ロール (コンソール) に切り替えるか、 または API オペレーションを呼び出すことで、ロールを引き受けることができます。 AWS CLI AWS 詳細については、「IAM ユーザーガイド」の「ロールを引き受けるための各種方法」を参照してください。

IAM ロールは、フェデレーションユーザーアクセス、一時的な IAM ユーザーのアクセス許可、クロスアカウントアクセス、クロスサービスアクセス、および Amazon EC2 で実行するアプリケーションに役立ちます。詳細については、IAM ユーザーガイド の IAM でのクロスアカウントリソースアクセス を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、ID AWS またはリソースにアタッチします。ポリシーは、アイデンティティまたはリソースに関連付けられたときにアクセス許可を定義します。は、プリンシパルがリクエストを行うときにこれらのポリシー AWS を評価します。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は、ポリシーを使用して、どのプリンシパルがどのリソースに対して、どのような条件でアクションを実行できるかを定義することで、誰が何にアクセスできるかを指定します。

デフォルトでは、ユーザーやロールにアクセス許可はありません。IAM 管理者は IAM ポリシーを作成してロールに追加し、このロールをユーザーが引き受けられるようにします。IAM ポリシーは、オペレーションの実行方法を問わず、アクセス許可を定義します。

アイデンティティベースのポリシー

アイデンティティベースのポリシーは、アイデンティティ (ユーザー、グループ、またはロール) にアタッチできる JSON アクセス許可ポリシードキュメントです。これらのポリシーは、アイデンティティがどのリソースに対してどのような条件下でどのようなアクションを実行できるかを制御します。アイデンティティベースポリシーの作成方法については、IAM ユーザーガイド の カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する を参照してください。

アイデンティティベースのポリシーは、インラインポリシー (単一の ID に直接埋め込む) または管理ポリシー (複数の ID にアタッチされたスタンドアロンポリシー) にすることができます。管理ポリシーとインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の「管理ポリシーとインラインポリシーのいずれかを選択する」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。例としては、IAM ロール信頼ポリシーや Amazon S3 バケットポリシーなどがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは、IAM の AWS マネージドポリシーを使用できません。

その他のポリシータイプ

AWS は、より一般的なポリシータイプによって付与されるアクセス許可の上限を設定できる追加のポリシータイプをサポートしています。

- アクセス許可の境界 – アイデンティティベースのポリシーで IAM エンティティに付与することのできるアクセス許可の数の上限を設定します。詳細については、「IAM ユーザーガイド」の「IAM エンティティのアクセス許可境界」を参照してください。
- サービスコントロールポリシー (SCP) - AWS Organizations内の組織または組織単位の最大のアクセス許可を指定します。詳細については、「AWS Organizations ユーザーガイド」の「サービスコントロールポリシー」を参照してください。
- リソースコントロールポリシー (RCP) – は、アカウント内のリソースで利用できる最大数のアクセス許可を定義します。詳細については、「AWS Organizations ユーザーガイド」の「リソースコントロールポリシー (RCP)」を参照してください。
- セッションポリシー – ロールまたはフェデレーションユーザーの一時セッションを作成する際にパラメータとして渡される高度なポリシーです。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成されるアクセス許可を理解するのがさらに難しくなります。が複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、「IAM ユーザーガイド」の「ポリシー評価ロジック」を参照してください。

AWS エンドユーザーメッセージングプッシュと IAM の連携方法

IAM を使用して AWS エンドユーザーメッセージングプッシュへのアクセスを管理する前に、 AWS エンドユーザーメッセージングプッシュで使用できる IAM 機能を確認してください。

AWS エンドユーザーメッセージングプッシュで使用できる IAM 機能

IAM 機能	AWS エンドユーザーメッセージングプッシュのサポート
アイデンティティベースのポリシー	あり
リソースベースのポリシー	はい
ポリシーアクション	あり
ポリシーリソース	あり
ポリシー条件キー	あり
ACL	なし
ABAC (ポリシー内のタグ)	部分的
一時認証情報	あり
プリンシパルアクセス権限	あり
サービスロール	あり
サービスリンクロール	いいえ

AWS エンドユーザーメッセージングプッシュおよびその他の AWS のサービスがほとんどの IAM 機能と連携する方法の概要については、IAM ユーザーガイドのAWS 「IAM と連携する のサービス」を参照してください。

AWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシー

アイデンティティベースのポリシーのサポート: あり

アイデンティティベースポリシーは、IAM ユーザー、ユーザーグループ、ロールなど、アイデンティティにアタッチできる JSON 許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースポリシーの作成方法については、「IAM ユーザーガイド」の「カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する」を参照してください。

IAM アイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、およびアクションを許可または拒否する条件を指定できます。JSON ポリシーで使用できるすべての要素について学ぶには、「IAM ユーザーガイド」の「IAM JSON ポリシーの要素のリファレンス」を参照してください。

AWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例

AWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例。

AWS エンドユーザーメッセージングプッシュ内のリソースベースのポリシー

リソースベースのポリシーのサポート: あり

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーで、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

クロスアカウントアクセスを有効にするには、全体のアカウント、または別のアカウントの IAM エンティティを、リソースベースのポリシーのプリンシパルとして指定します。詳細については、IAM ユーザーガイドのIAM でのクロスアカウントリソースアクセスを参照してください。

AWS エンドユーザーメッセージングプッシュのポリシーアクション

ポリシーアクションのサポート: あり

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

JSON ポリシーの Action 要素にはポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。このアクションは関連付けられたオペレーションを実行するためのアクセス許可を付与するポリシーで使用されます。

AWS エンドユーザーメッセージングプッシュアクションのリストを確認するには、「サービス認可リファレンス」のAWS 「エンドユーザーメッセージングプッシュで定義されるアクション」を参照してください。

AWS エンドユーザーメッセージングプッシュのポリシーアクションは、アクションの前に次のプレフィックスを使用します。

```
mobiletargeting
```

単一のステートメントで複数のアクションを指定するには、アクションをカンマで区切ります。

```
"Action": [
      "mobiletargeting:action1",
      "mobiletargeting:action2"
         ]
```

AWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例。

AWS エンドユーザーメッセージングプッシュのポリシーリソース

ポリシーリソースのサポート: あり

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

Resource JSON ポリシー要素はアクションが適用されるオブジェクトを指定します。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。リソースレベ

ルのアクセス許可をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

AWS エンドユーザーメッセージングプッシュリソースタイプとその ARNsAWS「エンドユーザーメッセージングプッシュで定義されるリソース」を参照してください。 各リソースの ARN を指定できるアクションについては、AWS「エンドユーザーメッセージングプッシュ で定義されるアクション」を参照してください。

AWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例。

AWS エンドユーザーメッセージングプッシュのポリシー条件キー

サービス固有のポリシー条件キーのサポート: あり

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

Condition 要素は、定義された基準に基づいてステートメントが実行される時期を指定します。イコールや未満などの条件演算子を使用して条件式を作成して、ポリシーの条件とリクエスト内の値を一致させることができます。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」のAWS「グローバル条件コンテキストキー」を参照してください。

AWS エンドユーザーメッセージングプッシュ条件キーのリストを確認するには、「サービス認可リファレンス」のAWS「エンドユーザーメッセージングプッシュの条件キー」を参照してください。条件キーを使用できるアクションとリソースについては、AWS「エンドユーザーメッセージングプッシュ で定義されるアクション」を参照してください。

AWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例。

AWS エンドユーザーメッセージングプッシュACLs

ACL のサポート: なし

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするためのアクセス許可を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

AWS エンドユーザーメッセージングプッシュによる ABAC

ABAC (ポリシー内のタグ) のサポート: 一部

属性ベースのアクセスコントロール (ABAC) は、タグと呼ばれる属性に基づいてアクセス許可を定義する認可戦略です。IAM エンティティと AWS リソースにタグをアタッチし、プリンシパルのタグがリソースのタグと一致するときにオペレーションを許可する ABAC ポリシーを設計できます。

タグに基づいてアクセスを管理するには、`aws:ResourceTag/key-name`、`aws:RequestTag/key-name`、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。

サービスがすべてのリソースタイプに対して 3 つの条件キーすべてをサポートする場合、そのサービスの値はありです。サービスが一部のリソースタイプに対してのみ 3 つの条件キーのすべてをサポートする場合、値は「部分的」になります。

ABAC の詳細については、「IAM ユーザーガイド」の「ABAC 認可でアクセス許可を定義する」を参照してください。ABAC をセットアップする手順を説明するチュートリアルについては、「IAM ユーザーガイド」の「属性ベースのアクセスコントロール (ABAC) を使用する」を参照してください。

AWS エンドユーザーメッセージングプッシュでの一時的な認証情報の使用

一時的な認証情報のサポート: あり

一時的な認証情報は、 AWS リソースへの短期的なアクセスを提供し、フェデレーションまたはスイッチロールの使用時に自動的に作成されます。長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成 AWS することをお勧めします。詳細については、「IAM ユーザーガイド」の「IAM の一時的な認証情報」および「AWS のサービス と IAM との連携」を参照してください。

AWS エンドユーザーメッセージングプッシュのクロスサービスプリンシパルアクセス許可

転送アクセスセッション (FAS) のサポート: あり

転送アクセスセッション (FAS) は、 を呼び出すプリンシパルのアクセス許可と AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストをリクエストする を使用します。FAS

リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

AWS エンドユーザーメッセージングプッシュのサービスロール

サービスロールのサポート: あり

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、IAM ユーザーガイドの AWS のサービスに許可を委任するロールを作成するを参照してください。

> **Warning**
>
> サービスロールのアクセス許可を変更すると、 AWS エンドユーザーメッセージングプッシュ機能が破損する可能性があります。 AWS エンドユーザーメッセージングプッシュが指示する場合にのみ、サービスロールを編集します。

AWS エンドユーザーメッセージングプッシュのサービスにリンクされたロール

サービスにリンクされたロールのサポート: なし

サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールのアクセス許可を表示できますが、編集することはできません。

サービスにリンクされたロールの作成または管理の詳細については、「IAM と提携するAWS のサービス」を参照してください。表の「サービスリンクロール」列に Yes と記載されたサービスを見つけます。サービスにリンクされたロールに関するドキュメントをサービスで表示するには、[はい] リンクを選択します。

AWS エンドユーザーメッセージングプッシュのアイデンティティベースのポリシーの例

デフォルトでは、ユーザーとロールには、 AWS エンドユーザーメッセージングプッシュリソースを作成または変更するアクセス許可はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。

これらのサンプルの JSON ポリシードキュメントを使用して IAM アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーを作成する (コンソール)」を参照してください。

各リソースタイプの ARN の形式など、 AWS エンドユーザーメッセージングプッシュで定義されるアクションとリソースタイプの詳細については、「サービス認可リファレンス」のAWS 「エンドユーザーメッセージングプッシュのアクション、リソース、および条件キー」を参照してください。 ARNs

トピック

- ポリシーに関するベストプラクティス
- AWS エンドユーザーメッセージングプッシュコンソールの使用
- 自分の権限の表示をユーザーに許可する

ポリシーに関するベストプラクティス

ID ベースのポリシーは、アカウント内の AWS エンドユーザーメッセージングプッシュリソースを作成、アクセス、または削除できるかどうかを決定します。これらのアクションでは、 AWS アカウントに費用が発生する場合があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行 – ユーザーとワークロードにアクセス許可の付与を開始するには、多くの一般的なユースケースにアクセス許可を付与するAWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義することで、アクセス許可をさらに減らすことをお勧めします。詳細については、IAM ユーザーガイド の AWS マネージドポリシー または ジョブ機能のAWS マネージドポリシー を参照してください。
- 最小特権を適用する – IAM ポリシーでアクセス許可を設定する場合は、タスクの実行に必要な許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用して許可を適用する方法の詳細については、IAM ユーザーガイド の IAM でのポリシーとアクセス許可 を参照してください。
- IAM ポリシーで条件を使用してアクセスをさらに制限する - ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。たとえば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、サービスアクションが などの特定の を通じて使用されている場合に AWS のサービス、サービスアクションへのアクセス

を許可することもできます CloudFormation。詳細については、IAM ユーザーガイド の IAM JSON ポリシー要素:条件 を参照してください。

- IAM アクセスアナライザー を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM アクセスアナライザー は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、IAM ユーザーガイド の IAM Access Analyzer でポリシーを検証する を参照してください。
- 多要素認証 (MFA) を要求する – で IAM ユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化するために MFA を有効にします。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、IAM ユーザーガイド の MFA を使用した安全な API アクセス を参照してください。

IAM でのベストプラクティスの詳細については、IAM ユーザーガイド の IAM でのセキュリティのベストプラクティス を参照してください。

AWS エンドユーザーメッセージングプッシュコンソールの使用

AWS エンドユーザーメッセージングプッシュコンソールにアクセスするには、最小限のアクセス許可のセットが必要です。これらのアクセス許可により、 の AWS エンドユーザーメッセージングプッシュリソースの詳細を一覧表示および表示できます AWS アカウント。最小限必要な許可よりも制限が厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ (ユーザーまたはロール) に対してコンソールが意図したとおりに機能しません。

AWS CLI または AWS API のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません。代わりに、実行しようとしている API オペレーションに一致するアクションのみへのアクセスが許可されます。

ユーザーとロールが引き続き AWS エンドユーザーメッセージングプッシュコンソールを使用できるようにするには、エンティティに AWSEndUserMessaging AWS 管理ポリシーもアタッチします。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
```

```
  {
   "Sid": "AWSEndUserMessaging",
   "Effect": "Allow",
   "Action": [
    "mobiletargeting:CreateApp",
                "mobiletargeting:GetApp",
                "mobiletargeting:GetApps",
                            "mobiletargeting:DeleteApp",
                            "mobiletargeting:GetChannels",
                            "mobiletargeting:GetApnsChannel",
                            "mobiletargeting:GetApnsVoipChannel",
                            "mobiletargeting:GetApnsVoipSandboxChannel",
                            "mobiletargeting:GetApnsSandboxChannel",
                "mobiletargeting:GetAdmChannel",
                "mobiletargeting:GetBaiduChannel",
                "mobiletargeting:GetGcmChannel",
                "mobiletargeting:UpdateApnsChannel",
                "mobiletargeting:UpdateApnsVoipChannel",
                "mobiletargeting:UpdateApnsVoipSandboxChannel",
                "mobiletargeting:UpdateBaiduChannel",
                "mobiletargeting:UpdateGcmChannel",
                "mobiletargeting:UpdateAdmChannel"
            ],
   "Resource": [
    "*"
   ]
  }
 ]
}
```

自分の権限の表示をユーザーに許可する

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS エンドユーザーメッセージングプッシュのアイデンティティとアクセスのトラブルシューティング

以下の情報は、 AWS エンドユーザーメッセージングプッシュと IAM の使用時に発生する可能性がある一般的な問題の診断と修正に役立ちます。

トピック

- AWS エンドユーザーメッセージングプッシュでアクションを実行する権限がありません
- iam:PassRole を実行する権限がありません
- 自分の 以外のユーザーに AWS エンドユーザーメッセージングプッシュリソース AWS アカウントへのアクセスを許可したい

AWS エンドユーザーメッセージングプッシュでアクションを実行する権限がありません

アクションを実行する権限がないというエラーが表示された場合は、そのアクションを実行できるようにポリシーを更新する必要があります。

次のエラー例は、mateojackson IAM ユーザーがコンソールを使用して、ある *my-example-widget* リソースに関する詳細情報を表示しようとしたことを想定して、その際に必要な mobiletargeting:*GetWidget* アクセス許可を持っていない場合に発生するものです。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 mobiletargeting:GetWidget on resource: my-example-widget
```

この場合、mobiletargeting:*GetWidget* アクションを使用して *my-example-widget* リソースへのアクセスを許可するように、mateojackson ユーザーのポリシーを更新する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

iam:PassRole を実行する権限がありません

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、 AWS エンドユーザーメッセージングプッシュにロールを渡すことができるようにポリシーを更新する必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、既存のロールをそのサービスに渡すことができます。そのためには、サービスにロールを渡すアクセス許可が必要です。

次の例のエラーは、 という IAM ユーザーがコンソールを使用して AWS エンドユーザーメッセージングプッシュでアクションを実行marymajorしようとすると発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与されたアクセス許可が必要です。Mary には、ロールをサービスに渡すアクセス許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

自分の 以外のユーザーに AWS エンドユーザーメッセージングプッシュリソース AWS アカウント へのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- AWS エンドユーザーメッセージングプッシュがこれらの機能をサポートしているかどうかを確認するには、「」を参照してくださいAWS エンドユーザーメッセージングプッシュと IAM の連携方法。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、IAM ユーザーガイドの「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティーに提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、IAM ユーザーガイド の 外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可 を参照してください。
- クロスアカウントアクセスにおけるロールとリソースベースのポリシーの使用方法の違いについては、IAM ユーザーガイド の IAM でのクロスアカウントのリソースへのアクセス を参照してください。

AWS エンドユーザーメッセージングプッシュのコンプライアンス検証

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、AWS のサービス「コンプライアンスプログラムによる対象範囲内のコンプライアンス」を参照し、関心のあるコンプライアンスプログラムを選択します。一般的な情報については、AWS「コンプライアンスプログラム」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「Downloading Reports in AWS Artifact」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS のサービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。を使用する際のコンプライアンス責任の詳細については AWS のサービス、AWS 「 セキュリティドキュメント」を参照してください。

AWS エンドユーザーメッセージングプッシュの耐障害性

AWS グローバルインフラストラクチャは、 AWS リージョン およびアベイラビリティーゾーンを中心に構築されています。 は、低レイテンシー、高スループット、高度に冗長なネットワークで接続された、物理的に分離および分離された複数のアベイラビリティーゾーン AWS リージョン を提供します。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェールオーバーするアプリケーションとデータベースを設計および運用することができます。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、フォールトトレランス、および拡張性が優れています。

AWS リージョン およびアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

AWS グローバルインフラストラクチャに加えて、 AWS End User Messaging Push には、データの耐障害性とバックアップのニーズをサポートするのに役立つ機能がいくつか用意されています。

AWS エンドユーザーメッセージングプッシュのインフラストラクチャセキュリティ

マネージドサービスである AWS エンドユーザーメッセージングプッシュは、ホワイトペーパー「Amazon Web Services: セキュリティプロセスの概要」に記載されている AWS グローバルネットワークセキュリティ手順で保護されています。

AWS 公開された API コールを使用して、ネットワーク経由で AWS エンドユーザーメッセージングプッシュにアクセスします。クライアントは、Transport Layer Security (TLS) 1.2 以降をサポートする必要があります。また、DHE (Ephemeral Diffie-Hellman) や ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) などの Perfect Forward Secrecy (PFS) を使用した暗号スイートもクライアントでサポートされている必要があります。これらのモードは Java 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストにはアクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

設定と脆弱性の分析

マネージドサービスである AWS エンドユーザーメッセージングプッシュは、ホワイトペーパー「Amazon Web Services: セキュリティプロセスの概要」に記載されている AWS グローバルネットワークセキュリティ手順で保護されています。つまり、 は基本的なセキュリティタスクと手順を AWS 管理して実行し、アカウントとリソースの基盤となるインフラストラクチャを強化、パッチ適用、更新、その他の方法で維持します。これらの手順は適切なサードパーティーによって確認され、認証されています。

セキュリティのベストプラクティス

AWS Identity and Access Management (IAM) アカウントを使用して、API オペレーション、特にリソースを作成、変更、削除するオペレーションへのアクセスを制御します。 API には、プロジェクト、キャンペーン、ジャーニーなどのリソースが含まれます。

- リソースを管理するユーザー (本人を含む) ごとに個別のユーザーを作成します。 AWS ルート認証情報を使用してリソースを管理しないでください。
- それぞれの職務の実行に最低限必要になる一連のアクセス許可を各ユーザーに付与します。
- IAM グループを使用して、複数のユーザーのアクセス許可を効果的に管理します。
- IAM 認証情報のローテーションを定期的に行います。

セキュリティの詳細については、「AWS エンドユーザーメッセージングプッシュのセキュリティ」を参照してください。IAM の詳細については、「AWS Identity and Access Management」を参照してください。IAM のベストプラクティスについては、「IAM のベストプラクティス」を参照してください。

AWS エンドユーザーメッセージングプッシュのモニタリング

モニタリングは、 AWS エンドユーザーメッセージングプッシュやその他の AWS ソリューションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。AWS には、 AWS エンドユーザーメッセージングプッシュを監視し、問題が発生したときに報告し、必要に応じて自動アクションを実行するための以下のモニタリングツールが用意されています。

- Amazon CloudWatch は、 AWS リソースと、 で実行しているアプリケーションを AWS リアルタイムでモニタリングします。メトリクスの収集と追跡、カスタマイズしたダッシュボードの作成、および指定したメトリクスが指定したしきい値に達したときに通知またはアクションを実行するアラームの設定を行うことができます。例えば、CloudWatch で Amazon EC2 インスタンスの CPU 使用率などのメトリクスを追跡し、必要に応じて新しいインスタンスを自動的に起動できます。詳細については、「Amazon CloudWatch ユーザーガイド」を参照してください。
- Amazon CloudWatch Logs では、Amazon EC2 インスタンス、CloudTrail、その他ソースから得たログファイルのモニタリング、保存、およびアクセスが可能です。CloudWatch Logs は、ログファイル内の情報をモニタリングし、特定のしきい値が満たされたときに通知します。高い耐久性を備えたストレージにログデータをアーカイブすることも可能です。詳細については、「Amazon CloudWatch Logs ユーザーガイド」を参照してください。
- Amazon EventBridge を使用すると、 AWS サービスを自動化し、アプリケーションの可用性の問題やリソースの変更などのシステムイベントに自動的に対応できます。 AWS サービスからのイベントは、ほぼリアルタイムで EventBridge に配信されます。簡単なルールを記述して、注目するイベントと、イベントがルールに一致した場合に自動的に実行するアクションを指定できます。詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。
- AWS CloudTrail は、 AWS アカウントによって、またはアカウントに代わって行われた API コールおよび関連イベントをキャプチャし、指定した Amazon S3 バケットにログファイルを配信します。が呼び出したユーザーとアカウント AWS、呼び出し元のソース IP アドレス、および呼び出しの発生日時を特定できます。詳細については、AWS CloudTrail ユーザーガイドをご参照ください。

Amazon CloudWatch による AWS エンドユーザーメッセージングプッシュのモニタリング

CloudWatch を使用して AWS CloudWatch は raw データを収集し、読み取り可能なほぼリアルタイムのメトリクスに加工します。これらの統計は 15 か月間保持されるため、履歴情報にアクセスし、ウェブアプリケーションまたはサービスの動作をより的確に把握できます。また、特定のしきい値を監視するアラームを設定し、これらのしきい値に達したときに通知を送信したりアクションを実行したりできます。詳細については、「Amazon CloudWatch ユーザーガイド」を参照してください。

メトリクスとディメンションのリストについては、Amazon Pinpoint ユーザーガイド」の CloudWatch による Amazon Pinpoint のモニタリング」を参照してください。 Amazon Pinpoint

を使用した AWS エンドユーザーメッセージングプッシュ API コールのログ記録 AWS CloudTrail

AWS エンドユーザーメッセージングプッシュは AWS CloudTrail、 AWS エンドユーザーメッセージングプッシュのユーザー、ロール、または サービスによって実行されたアクションを記録する AWS サービスである と統合されています。CloudTrail は AWS 、エンドユーザーメッセージングプッシュのすべての API コールをイベントとしてキャプチャします。キャプチャされた呼び出しには、 AWS エンドユーザーメッセージングプッシュコンソールからの呼び出しと AWS 、エンドユーザーメッセージングプッシュ API オペレーションへのコード呼び出しが含まれます。証跡を作成する場合は、 AWS エンドユーザーメッセージングプッシュのイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。CloudTrail によって収集された情報を使用して、 AWS エンドユーザーメッセージングプッシュに対して行われたリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

CloudTrail の詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

AWS CloudTrail でのエンドユーザーメッセージングプッシュ情報

アカウントを作成する AWS アカウント と、 で CloudTrail が有効になります。 AWS エンドユーザーメッセージングプッシュでアクティビティが発生すると、そのアクティビティはイベント履歴の他の AWS サービスイベントとともに CloudTrail イベントに記録されます。で最近のイベントを表示、検索、ダウンロードできます AWS アカウント。詳細については、 CloudTrail イベント履歴でのイベントの表示を参照してください。

AWS エンドユーザーメッセージングプッシュのイベントなど AWS アカウント、 のイベントの継続的な記録については、証跡を作成します。追跡により、CloudTrail はログファイルを Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成するときに、証跡がすべての AWS リージョンに適用されます。証跡は、 AWS パーティション内のすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、CloudTrail ログで収集されたイベントデータをより詳細に分析し、それに基づく対応を行うように他の AWS サービスを設定できます。詳細については、次を参照してください:

- 追跡を作成するための概要
- 「CloudTrail がサポートされているサービスと統合」
- 「CloudTrail の Amazon SNS 通知の設定」
- 「複数のリージョンから CloudTrail ログファイルを受け取る」および「複数のアカウントから CloudTrail ログファイルを受け取る」

すべての AWS エンドユーザーメッセージングプッシュアクションは CloudTrail によってログに記録され、AWS 「エンドユーザーメッセージングプッシュ API リファレンス」に記載されています。例えば、`GetAdmChannel`、`UpdateApnsChannel`、`GetApnsVoipChannel` の各アクションを呼び出すと、CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます。

- リクエストがルートまたは AWS Identity and Access Management (IAM) ユーザー認証情報を使用して行われたかどうか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の AWS サービスによって行われたかどうか。

詳細については、「CloudTrail userIdentity エレメント」を参照してください。

AWS エンドユーザーメッセージングプッシュログファイルエントリについて

「トレイル」は、指定した Amazon S3 バケットにイベントをログファイルとして配信するように設定できます。CloudTrail のログファイルは、単一か複数のログエントリを含みます。イベントは任意ソースからの単一リクエストを表し、リクエストされたアクション、アクションの日時、リクエスト

パラメータなどの情報を含みます。CloudTrail ログファイルは、パブリック API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

インターフェイスエンドポイント (AWS PrivateLink) を使用して AWS エンドユーザーメッセージングプッシュにアクセスする

を使用して AWS PrivateLink 、VPC と AWS エンドユーザーメッセージングプッシュの間にプライベート接続を作成できます。インターネットゲートウェイ、NAT デバイス、VPN 接続、または Direct Connect 接続を使用せずに、VPC 内にあるかのように AWS エンドユーザーメッセージングプッシュにアクセスできます。VPC のインスタンスは AWS 、エンドユーザーメッセージングプッシュにアクセスするためにパブリック IP アドレスを必要としません。

このプライベート接続を確立するには、 AWS PrivateLinkを利用したインターフェイスエンドポイントを作成します。インターフェイスエンドポイントに対して有効にする各サブネットにエンドポイントネットワークインターフェイスを作成します。これらは、 AWS エンドユーザーメッセージングプッシュ宛てのトラフィックのエントリポイントとして機能するリクエスタマネージドネットワークインターフェイスです。

詳細については、「 AWS PrivateLink ガイド」の「 AWS のサービス からアクセス AWS PrivateLinkする」を参照してください。

AWS エンドユーザーメッセージングプッシュに関する考慮事項

AWS エンドユーザーメッセージングプッシュのインターフェイスエンドポイントを設定する前に、「 AWS PrivateLink ガイド」の「考慮事項」を確認してください。

AWS エンドユーザーメッセージングプッシュは、インターフェイスエンドポイントを介したすべての API アクションの呼び出しをサポートしています。

VPC エンドポイントポリシーは AWS 、エンドユーザーメッセージングプッシュではサポートされていません。デフォルトでは、インターフェイスエンドポイントを介した AWS エンドユーザーメッセージングプッシュへのフルアクセスが許可されます。または、セキュリティグループをエンドポイントネットワークインターフェイスに関連付けて、インターフェイスエンドポイントを介して AWS End User Messaging Push へのトラフィックを制御することもできます。

AWS エンドユーザーメッセージングプッシュ用のインターフェイスエンドポイントを作成する

Amazon VPC コンソールまたは AWS Command Line Interface () を使用して、 AWS エンドユーザーメッセージングプッシュのインターフェイスエンドポイントを作成できますAWS CLI。詳細については、「AWS PrivateLink ガイド」の「インターフェイスエンドポイントを作成」を参照してください。

次のサービス名を使用して AWS 、エンドユーザーメッセージングプッシュのインターフェイスエンドポイントを作成します。

```
com.amazonaws.region.pinpoint
```

インターフェイスエンドポイントのプライベート DNS を有効にすると、デフォルトのリージョン DNS 名を使用して、 AWS エンドユーザーメッセージングプッシュに API リクエストを行うことができます。例えば、`com.amazonaws.us-east-1.pinpoint` と指定します。

インターフェイスエンドポイントのエンドポイントポリシーを作成する

エンドポイントポリシーは、インターフェイスエンドポイントにアタッチできる IAM リソースです。デフォルトのエンドポイントポリシーでは、インターフェイスエンドポイントを介した AWS エンドユーザーメッセージングプッシュへのフルアクセスが許可されます。VPC から AWS End User Messaging Push に許可されるアクセスを制御するには、インターフェイスエンドポイントにカスタムエンドポイントポリシーをアタッチします。

エンドポイントポリシーは以下の情報を指定します。

- アクションを実行できるプリンシパル (AWS アカウント、IAM ユーザー、IAM ロール)。
- 実行可能なアクション。
- このアクションを実行できるリソース。

詳細については、AWS PrivateLink ガイドのControl access to services using endpoint policies (エンドポイントポリシーを使用してサービスへのアクセスをコントロールする)を参照してください。

例: AWS エンドユーザーメッセージングプッシュアクションの VPC エンドポイントポリシー

以下は、カスタムエンドポイントポリシーの例です。このポリシーをインターフェイスエンドポイントにアタッチすると、すべてのリソースのすべてのプリンシパルに対して、リストされている AWS End User Messaging Push アクションへのアクセスが許可されます。

```
{
   "Statement": [
      {
         "Principal": "*",
         "Effect": "Allow",
         "Action": [
             "mobiletargeting:CreateApp",
             "mobiletargeting:DeleteApp"
         ],
         "Resource":"*"
      }
   ]
}
```

AWS エンドユーザーメッセージングプッシュのクォータ

AWS アカウント には、サービスごとに、以前は制限と呼ばれていたデフォルトのクォータがあります AWS 。特に明記されていない限り、クォータは地域固有です。一部のクォータについては引き上げをリクエストできますが、その他のクォータについては引き上げることはできません。

AWS エンドユーザーメッセージングプッシュのクォータを表示するには、Service Quotas コンソールを開きます。ナビゲーションペインで、AWS サービスを選択し、Amazon Pinpoint を選択します。

AWS アカウントには、 AWS エンドユーザーメッセージングプッシュに関連する次のクォータがあります。

リソース	デフォルトのクォータ	引き上げの対象かどうかの確認
キャンペーンで 1 秒あたりに送信できるプッシュ通知の最大数	25000 通知 / 秒	はい、Service Quotas コンソールを使用します
Amazon Device Messaging (ADM) のメッセージペイロードサイズ	メッセージごとに 6 KB	いいえ
Apple Push Notification サービス (APN) メッセージペイロードサイズ	メッセージごとに 4 KB	いいえ
APNS サンドボックスメッセージのペイロードサイズ	メッセージごとに 4 KB	いいえ
Baidu Cloud Push メッセージペイロードサイズ	メッセージごとに 4 KB	いいえ
Firebase Cloud Messaging (FCM) メッセージペイロードサイズ	メッセージごとに 4 KB	いいえ

AWS 「 エンドユーザーメッセージングプッシュユーザーガイド」のドキュメント履歴

次の表に、 AWS エンドユーザーメッセージングプッシュのドキュメントリリースを示します。

変更	説明	日付
初回リリース	AWS エンドユーザーメッセージングプッシュユーザーガイドの初回リリース	2024 年 7 月 24 日

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。