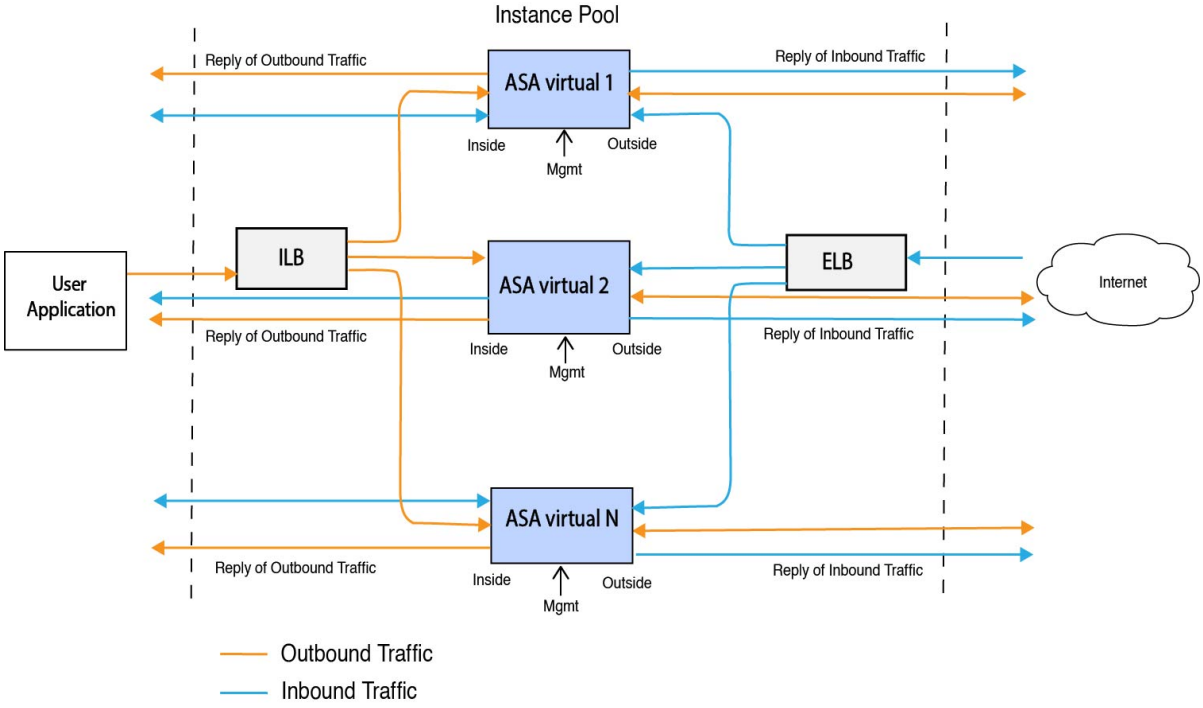
OCI への ASA 仮想 Auto Scale ソリューションの導入

使用例

この ASA 仮想 の導入例：OCI Auto Scale ソリューションは、導入例の図に示されています。インターネット向けのロードバランサには、リスナーとターゲットグループの組み合わせを使用してポートが有効になっているパブリック IP アドレスがあります。

図 *1*: 導入例の図



ポートベースの分岐は、ネットワークトラフィックに実装できます。この分岐は、NAT ルールによって実現できます。分岐の設定例については、以下のセクションで説明します。

前提条件

権限およびポリシー

ソリューションを導入するために必要な OCI の権限とポリシーは次のとおりです。

1. **ユーザーおよびグループ**

（注） ユーザーとグループを作成するには、OCIユーザーまたはテナンシー管理者である必要があります。

Oracle Cloud Infrastructure のユーザーアカウントと、そのユーザーアカウントが属するグループを作成します。ユーザーアカウントを持つ関連グループが存在する場合は、作成する必要はありません。ユーザーとグループの作成手順については、「グループとユーザーの作成」を参照してください。

2. **グループ ポリシー**

ポリシーを作成したら、それをグループにマッピングする必要があります。ポリシーを作成するには、**[OCI]** > **[アイデンティティとセキュリティ（Identity & Security）]** > **[ポリシー（Policies）]** > **[ポリシーの作成（Create Policy）]**に移動します。次のポリシーを作成して、目的のグループに追加します。

- グループ <*Group_Name*> がコンパートメント <*Compartment_Name*> でメトリックを使用することを許可します。
- グループ <*Group_Name*> がコンパートメント <*Compartment_Name*> でアラームを管理することを許可します。
- グループ <*Group_Name*> がコンパートメント <*Compartment_Name*> で ONS トピックを管理することを許可します。
- グループ <*Group_Name*> がコンパートメント <*Compartment_Name*> でメトリックを検査することを許可します。
- グループ <*Group_Name*> がコンパートメント <*Compartment_Name*> でメトリックを読み取ることを許可します。
- グループ <*Group_Name*> がコンパートメント <*Compartment_Name*> でタグの名前空間を使用することを許可します。
- グループ <*Group_Name*> がコンパートメント <*Compartment_Name*>でロググループを読み取ることを許可します。
- グループ <*Group_Name*> がインスタンスプール コンパートメント <*Compartment_Name*> を使用することを許可します。
- グループ <*Group_Name*> がテナントでクラウドシェルを使用することを許可します。
- グループ <*Group_Name*> がテナントのオブジェクトストレージ名前空間を読み取ることを許可します
- グループ <*Group_Name*> がテナント内のリポジトリを管理することを許可します。

（注） テナントレベルでポリシーを作成することもできます。ユーザーの責任と判断のもとで、すべての権限を自由に指定できます。

3. **Oracle 関数の権限**

   Oracle 関数が別の Oracle Cloud Infrastructure リソースにアクセスできるようにするには、関数をダイナミックグループに含めてから、そのリソースへのダイナミック グループ アクセスを許可するポリシーを作成します。

4. **ダイナミックグループの作成**

   ダイナミックグループを作成するには、**[OCI]**>**[アイデンティティとセキュリティ（Identity & Security）]** > **ダイナミックグループ（Dynamic Group）]** > **[ダイナミックグループの作成（Create Dynamic Group）]**に移動します。

ダイナミックグループの作成時に次のルールを指定します。

```
ALL {resource.type = 'fnfunc', resource.compartment.id = '<Your_Compartment_OCID>'}
```

ダイナミックグループの詳細については、次を参照してください。

- https://docs.oracle.com/en-us/iaas/Content/Functions/Tasks/functionsaccessingociresources.htm
- https://docs.oracle.com/en-us/iaas/Content/Identity/Tasks/managingdynamicgroups.htm

5. **ダイナミックグループのポリシーの作成**

   ポリシーを追加するには、**[OCI]**>**[アイデンティティとセキュリティ（Identity & Security）]**>**[ポリシー（Policies）]**>**[ポリシーの作成（Create Policy）]**に移動します。次のポリシーをグループに追加します。

   ```
   Allow dynamic-group <Dynamic_Group_Name> to manage all-resources in compartment
   <Compartment_OCID>
   ```

GitHub からのファイルのダウンロード

ASA 仮想：OCI Auto Scale ソリューションは、GitHub リポジトリ形式で配布されます。リポジトリからファイルをプルまたはダウンロードできます。

Python3 環境

make.py ファイルは、複製されたリポジトリ内にあります。このプログラムは、Oracle 関数とテンプレートファイルを ZIP ファイルに圧縮します。それらをターゲットフォルダーにコピーします。これらのタスクを実行するには、Python 3 環境が設定されている必要があります。

（注） この Python スクリプトは Linux 環境でのみ使用できます。

インフラストラクチャ設定

次を設定する必要があります。

1. **VCN**

   ASA 仮想 アプリケーションの要件に応じて VCN を作成します。インターネットへのルートが割り当てられたサブネットが 1 つ以上あるインターネットゲートウェイを備えた VPC を作成します。

   VCN の作成については、「https://docs.oracle.com/en-us/iaas/Content/GSG/Tasks/creatingnetwork.htm」を参照してください。

2. **アプリケーションサブネット**

   ASA 仮想 アプリケーションの要件に応じてサブネットを作成します。このユースケースに従ってソリューションを導入するには、ASA 仮想インスタンスの運用に 3 つのサブネットが必要です。

サブネットの作成については、
https://docs.oracle.com/en-us/iaas/Content/Network/Tasks/managingVCNs_topic-Overview_of_VCNs_and_Subnets.htm#
を参照してください。

3. **外部サブネット**

   サブネットには、インターネットゲートウェイへの「0.0.0.0/0」のルートが必要です。このサブネットには、Cisco ASA 仮想 の外部インターフェイスとインターネット向けロードバランサが含まれています。アウトバウンドトラフィック用に NAT ゲートウェイが追加されていることを確認します。

   詳細については、次のマニュアルを参照してください。

   - https://docs.oracle.com/en-us/iaas/Content/Network/Tasks/managingIGs.htm
   - https://docs.oracle.com/en-us/iaas/Content/Network/Tasks/NATgateway.htm#To_create_a_NAT_gateway

4. **内部サブネット**

   これは、NAT/インターネットゲートウェイの有無にかかわらず、アプリケーションサブネットに似ています。

（注） ASA 仮想 正常性プローブの場合、ポート 80 を介してメタデータサーバー（169.254.169.254）に到達できます。

5. **管理サブネット**

   管理サブネットは、ASA 仮想 への SSH 接続をサポートするようにパブリックにする必要があります。

6. **セキュリティ グループ：ASA 仮想 インスタンスのネットワーク セキュリティ グループ**

   次の要件に対応した ASA 仮想 インスタンスのセキュリティグループを設定します。

   - Oracle 関数（同じ VCN 内）は、ASA 仮想 の管理アドレスへの SSH 接続を実行します。
   - 管理ホストでは、 SSH を介した ASA 仮想 インスタンスへのアクセスが必要になる場合があります。
   - ASA 仮想 はライセンスのために CSSM/Satellite サーバーとの通信を開始します。

7. **オブジェクトストレージの名前空間**

   このオブジェクトストレージの名前空間は、configuration.txt ファイルを持つ静的 Web サイトをホストするために使用されます。configuration.txt ファイルの事前認証済みリクエストを作成する必要があります。この事前認証された URL は、テンプレートの展開時に使用されます。

（注） アップロードされた次の設定に、HTTP URL を介して ASA 仮想 インスタンスからアクセスできることを確認します。

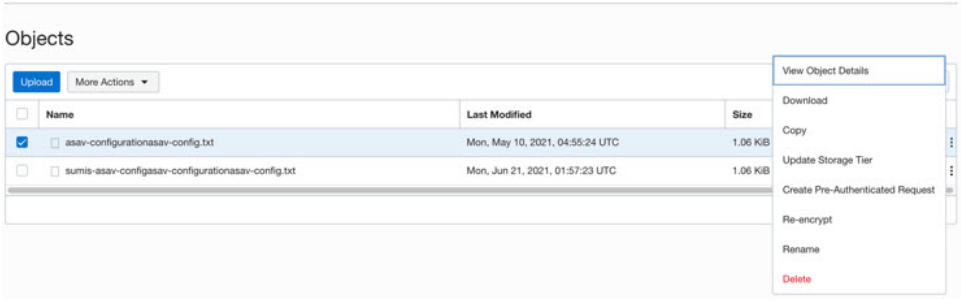
ASA 仮想 を起動すると、`$ copy /noconfirm <configuration.txt file's pre-authenticated request URL > disk0:Connfiguration.txt` コマンドが実行されます。

このコマンドにより、ASA 仮想 の起動を configuration.txt ファイルで設定できるようになります。

8. **configuration.txt ファイルのアップロード**

   ASA 仮想 構成ファイルの事前認証済みリクエスト URL を作成するには、次の手順を実行します。

   1. **[バケット（Buckets）] > [バケットの作成（Create Bucket）]**の順にクリックします。
   2. [アップロード（Upload）] をクリックします。
   3. 構成ファイルがアップロードされたら、下の図に示すように、[事前認証済みリクエストの作成（Create Pre-Authenticated Request）] を選択します。



（注） これで、オラクル関数から構成ファイルにアクセスできるようになります。

ネットワーク構成

1. **インバウンドトラフィック**

   オブジェクト、ライセンス、NAT ルール、およびアクセスポリシーの設定で説明されているように、configuration.txt 内の <*Application VM IP*> アドレスが正しいことを確認します。

2. **アウトバウンドトラフィック**

   - オブジェクト、ライセンス、NAT ルール、およびアクセスポリシーの設定で説明されているように、configuration.txt 内の <*External Server IP*> アドレスが正しいことを確認します。
   - 外部 VCN に 1 つの NAT ゲートウェイがあることを確認します。

- 次の図の例に示すように、 NAT ゲートウェイを経由する外部 VCN のルートテーブル内の同じ <*External Server IP*> アドレスを追加してください。

Destination	Target Type	Target
0.0.0.0/0	Internet Gateway	outside-ig
8.8.8.8/32	NAT Gateway	nat-gw

パスワードの暗号化

（注） この手順の詳細については、「Vault とシークレットの作成」を参照してください。

ASA 仮想 のパスワードは、自動スケーリング中に使用されるすべての ASA 仮想 インスタンスを設定するために使用されます。また、ASA 仮想 インスタンスの CPU 使用率データを取得するために使用されます。

したがって、パスワードを時々保存して処理する必要があります。頻繁な変更と脆弱性のため、プレーンテキスト形式での「パスワードの編集や保存はできません。パスワードには、暗号化された形式のみを使用する必要があります。

暗号化された形式のパスワードを取得するには、次の手順を実行します。

手順

ステップ 1 Vault を作成します。

OCI Vault は、マスター暗号化キーを安全に作成および保存するサービスと、それらを使用する際に暗号化および復号化する方法を提供します。したがって、Vault は、自動スケールソリューションの残りの部分と同じコンパートメントに作成する必要があります（まだ作成していない場合）。

[OCI] > [アイデンティティとセキュリティ（Identity & Security）] > [Vault] > [新規Vaultの選択または作成（Choose or Create New Vault）]に移動します。

。

ステップ 2 マスター暗号化キーを作成します。

プレーンテキストのパスワードを暗号化するには、マスター暗号化キーが 1 つ必要です。

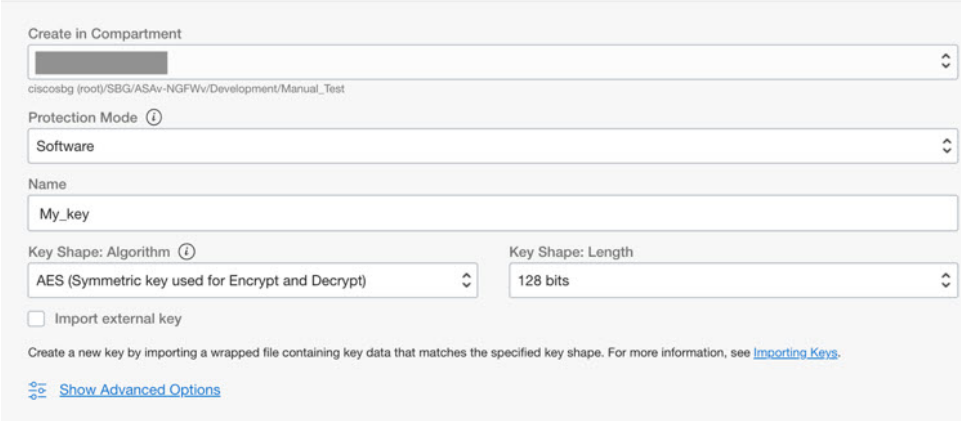
[OCI] > [アイデンティティとセキュリティ（Identity & Security）] > [Vault] > [キーの選択または作成（Choose or Create Key）]に移動します。

任意のビット長で、指定されたアルゴリズムのいずれかから任意のキーを選択します。

1. AES：128、192、256
2. RSA：2048、3072、4096

3. ECDSA：256、384、521

図 *2:* キーの作成



ステップ 3 暗号化されたパスワードを作成します。

1. **[OCI] > [CloudShell（OCI Cloud Terminal）を開く（Open CloudShell (OCI Cloud Terminal)**に移動します。
2. *<Password>* をお使いのパスワードに置き換えて、次のコマンドを実行します。

```
echo -n '<Password>' | base64
```

3. 選択した Vault から、暗号化エンドポイントとマスター暗号化キーの OCID をコピーします。次のように値を置き換えてから、暗号化コマンドを実行します。
   - KEY_OCID：キーの OCID
   - Cryptographic_Endpoint_URL：Vault の暗号化エンドポイント URL
   - Password：パスワード

   暗号化コマンド

```
oci kms crypto encrypt --key-id Key_OCID --endpoint
Cryptographic_Endpoint_URL --plaintext <base64-value-of-password>
```

4. 上記のコマンドの出力から暗号文をコピーし、必要に応じて使用します。

ASA 構成ファイルの準備

アプリケーションが展開されているか、アプリケーションの展開プランが利用可能であるかを確認します。

手順

ステップ 1 展開する前に、次の入力パラメータを収集します。

パラメータ	データタイプ	説明
tenancy_ocid	文字列	アカウントが属するテナントの OCID。テナントの OCID を見つける方法については、こちらを参照してください。 テナントの OCID は `ocid1.tenancy.oc1..<unique_ID>` のようになります。
compartment_id	文字列	リソースを作成するコンパートメントの OCID。 例： `ocid1.compartment.oc1..<unique_ID>`
compartment_name	文字列	コンパートメント名
region	文字列	リソースを作成するリージョンの一意の識別子。 例： `us-phoenix-1, us-ashburn-1`
lb_size	文字列	事前にプロビジョニングする外部および内部ロードバランサの合計帯域幅（入力および出力）を決定するテンプレート。 サポートされる値：100 Mbps、10 Mbps、10 Mbps-Micro、400 Mbps、8000 Mbps 例：100 Mbps
availability_domain	カンマ区切り値	例：Tpeb:PHX-AD-1 （注） クラウドシェルで **oci iam availability-domain list** コマンドを実行して、可用性ドメイン名を取得します。

パラメータ	データタイプ	説明
min_and_max_instance_count	カンマ区切り値	インスタンスプールに保持するインスタンスの最小数と最大数。 例：1,5
autoscale_group_prefix	文字列	テンプレートを使用して作成したリソースの名前に付けるプレフィックス。たとえば、リソースプレフィックスとして「autoscale」を指定すると、すべてのリソースはautoscale_resource1、autoscale_resource2 のように名前が付けられます。
asav_config_file_url	URL	ASA 仮想の構成用にオブジェクトストレージにアップロードする構成ファイルの URL。 （注） 構成ファイルの事前認証済みリクエスト URL を指定する必要があります 例： https://objectstorage.<region-name>.oraclecloud.com/<object-storage-name>/oci-asav-configuration.txt
mgmt_subnet_ocid	文字列	使用する管理サブネットのOCID。
inside_subnet_ocid	文字列	使用する内部サブネットのOCID。
outside_subnet_ocid	文字列	使用する外部サブネットのOCID。
mgmt_nsg_ocid	文字列	使用する管理サブネットのネットワーク セキュリティ グループのOCID。
inside_nsg_ocid	文字列	使用する内部サブネットのネットワーク セキュリティ グループのOCID。
outside_nsg_ocid	文字列	使用する外部サブネットのネットワーク セキュリティ グループのOCID。

パラメータ	データタイプ	説明
elb_listener_port	カンマ区切り値	外部ロードバランサリスナーの通信ポートのリスト。 例： 80
ilb_listener_port	カンマ区切り値	内部ロードバランサリスナーの通信ポートのリスト。 例： 80
health_check_port	文字列	ヘルスチェックを実行するロードバランサのバックエンドサーバーポート。 例：8080
instance_shape	文字列	作成するインスタンスのシェープ。シェイプにより、インスタンスに割り当てられるCPUの数、メモリの量、およびその他のリソースが決定されます。 サポートされているシェープ：「VM.Standard2.4」および「VM.Standard2.8」
lb_bs_policy	文字列	内部および外部ロードバランサのバックエンドセットに使用するロードバランサポリシー。ロードバランサポリシーの仕組みについて詳しくは、こちらを参照してください。 サポートされている値：「ROUND_ROBIN」、「LEAST_CONNECTIONS」、「IP_HASH」
image_name	文字列	インスタンスの構成に使用するマーケットプレイスのイメージ名。 デフォルト値：「Cisco ASA 仮想ファイアウォール（ASAv）」 （注） カスタムイメージを展開する場合は、custom_image_ocidパラメータを設定する必要があります。

パラメータ	データタイプ	説明
image_version	文字列	使用する OCI Marketplace で利用可能な ASA 仮想イメージのバージョン。現在、9.15.1.15 および 9.16.1 バージョンが利用可能です。 デフォルト値：「Cisco ASA 仮想ファイアウォール（ASAv）」
scaling_thresholds	カンマ区切り値	スケールインとスケールアウトで使用する CPU 使用率のしきい値。スケールインとスケールアウトのしきい値をカンマで区切って入力します。 例：15,50 15 はスケールインのしきい値、50 はスケールアウトのしきい値です。
custom_image_ocid	文字列	マーケットプレイスイメージを使用しない場合に、インスタンス構成に使用するカスタムイメージの OCID。 （注） custom_image_ocid はオプションパラメータです
asav_password	文字列	ASA 仮想 を構成するために SSH 接続する際の、ASA 仮想 の暗号化形式のパスワード。パスワードを暗号化する方法については、コンフィギュレーションガイドを使用するか、こちらを参照してください。
cryptographic_endpoint	文字列	暗号化エンドポイントは、パスワードの復号化に使用される URL です。Vault で検索できます。
master_encryption_key_id	文字列	パスワードの暗号化に使用されたキーの OCID。Vault で検索できます。

パラメータ	データタイプ	説明
プロファイル名（Profile Name）		OCI のユーザーのプロファイル名です。ユーザーのプロファイルセクションの下にあります。 例：oracleidentitycloudservice/<user>@<mail>.com
オブジェクトストレージの名前空間		テナントの作成時に作成される一意の識別子です。この値は**[OCI]**>**[管理（Administration）]**>**[（テナントの詳細Tenancy Details）]**で確認できます。
認証トークン（Authorization Token）		OCI コンテナレジストリに Oracle 関数をプッシュすることを許可する Docker へのログイン時のパスワードとして使用されます。トークンを取得するには、**[OCI]** > **[アイデンティティ（Identity）]** > **[ユーザー（Users）]**>**[ユーザの詳細（User Details）]** > **[認証トークン（Auth Tokens）]** > **[トークンの生成（Generate Token）]**に移動します。

ステップ 2 ロードバランサの正常性プローブとアクセスポリシーのオブジェクト、ライセンス、NAT ルールを設定します。

```
! Default route via outside
route outside 0.0.0.0 0.0.0.0 <Outside Subnet gateway> 2

! Health Check Configuration
object network metadata-server
host 169.254.169.254
object service health-check-port
service tcp destination eq <health-check-port>
object service http-port
service tcp destination eq <traffic port>
route inside 169.254.169.254 255.255.255.255 <Inside Subnet GW> 1

! Health check NAT
nat (outside,inside) source static any interface destination static interface metadata-server service
 health-check-port http-port
nat (inside,outside) source static any interface destination static interface metadata-server service
 health-check-port http-port

! Outbound NAT
object network inside-subnet
subnet <Inside Subnet> <Inside Subnet Gateway>
object network external-server
host <External Server IP>
nat (inside,outside) source static inside-subnet interface destination static interface external-server
```

```
! Inbound NAT
object network outside-subnet
subnet <Outside Subnet> <Outside Subnet GW>
object network http-server-80
host <Application VM IP>
nat (outside,inside) source static outside-subnet interface destination static interface http-server-80


!
dns domain-lookup outside
DNS server-group DefaultDNS

! License Configuration
call-home
profile license
destination transport-method http
destination address http <URL>
debug menu license 25 production
license smart
feature tier standard
throughput level <Entitlement>
licence smart register idtoken <License token> force
!
```

これらの正常性プローブ接続およびデータプレーンがアクセスポリシーで許可されている必要があります。

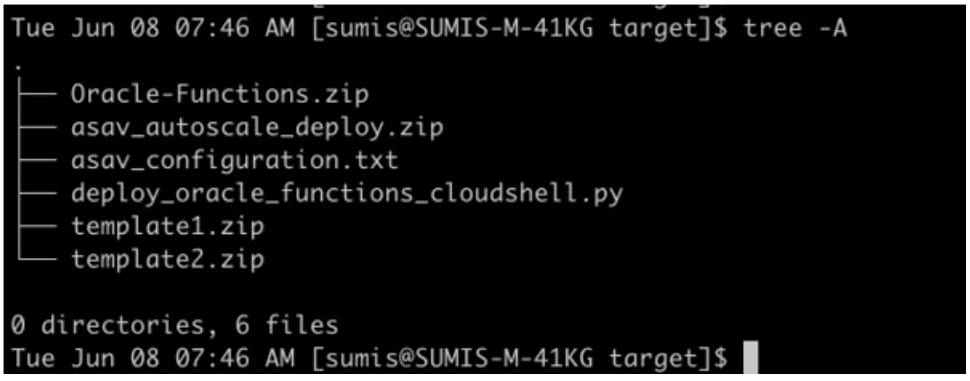
ステップ 3 設定の詳細を使用して *configuration.txt* ファイルを更新します。

ステップ 4 ユーザーが作成したオブジェクトストレージスペースに *configuration.txt* ファイルをアップロードし、アップロードしたファイルの事前認証リクエストを作成します。

（注）
スタックの展開で、configuration.txt の事前認証済みリクエスト URL が使用されていることを確認します。

ステップ 5 ZIP ファイルを作成します。

make.py ファイルは、複製されたリポジトリ内にあります。`python3 make.py build` コマンドを実行して、zip ファイルを作成します。対象フォルダには以下のファイルがあります。



（注）
クラウドシェルを使用して Auto Scale ソリューションを展開する場合は、`python3 make.py build` を実行する前に *easy_deploy/deployment_parameters.json* ファイルを更新します。更新については、「入力パラメータの収集」および「Oracle 関数の展開」を参照してください。

Auto Scale ソリューションの展開

展開の前提条件となる手順を完了したら、OCI スタックの作成を開始します。手動展開を実行するか、（クラウドシェルを使用した展開）を実行できます。該当するバージョン用の展開スクリプトとテンプレートは、GitHub リポジトリから入手できます。

手動展開

エンドツーエンドの Auto Scale ソリューションの展開は、次の 3 つの手順で構成されます。Terraform Template-1 スタックの展開、Oracle 関数の展開、次いで Terraform Template-2 の展開

Terraform Template-1 スタックの展開

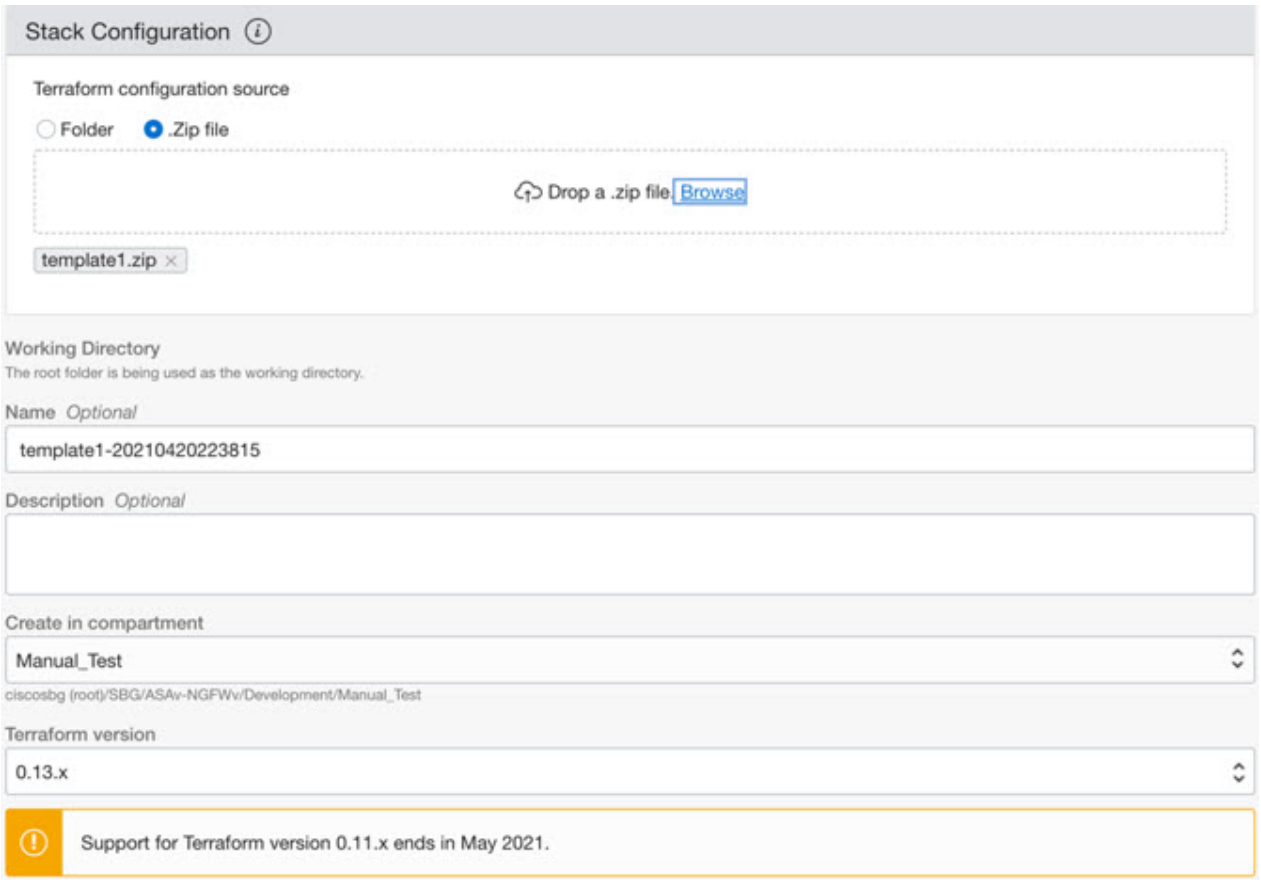
手順

ステップ 1 OCI ポータルにログインします。

地域は、画面の右上隅に表示されます。目的の地域内に存在していることを確認してください。

ステップ 2 **[デベロッパーサービス（Developer Service）] > [リソースマネージャ（Resource Manager）] > [スタック（Stack）] > [スタックの作成（Create Stack）]**の順に選択します。

[マイ設定（My Configuration）] を選択し、次の図に示すように、対象フォルダ内にある *Terraform template1.zip* ファイルを Terraform の設定ソースとして選択します。



ステップ 3 [トランスフォームバージョン（Transform version）] ドロップダウンリストで、0.13.x または 0.14.x を選択します。

ステップ 4 次の手順では、入力パラメータの収集で収集した詳細情報をすべて入力します。

（注）
有効な入力パラメータを入力しnoてください。そうしないと、以降の手順でスタックの展開に失敗する可能性があります。

ステップ 5 次の手順で**[Terraformアクション（Terraform Actions）]** > **[適用（Apply）]**を選択します。

正常に展開されたら、Oracle 関数の展開に進みます。

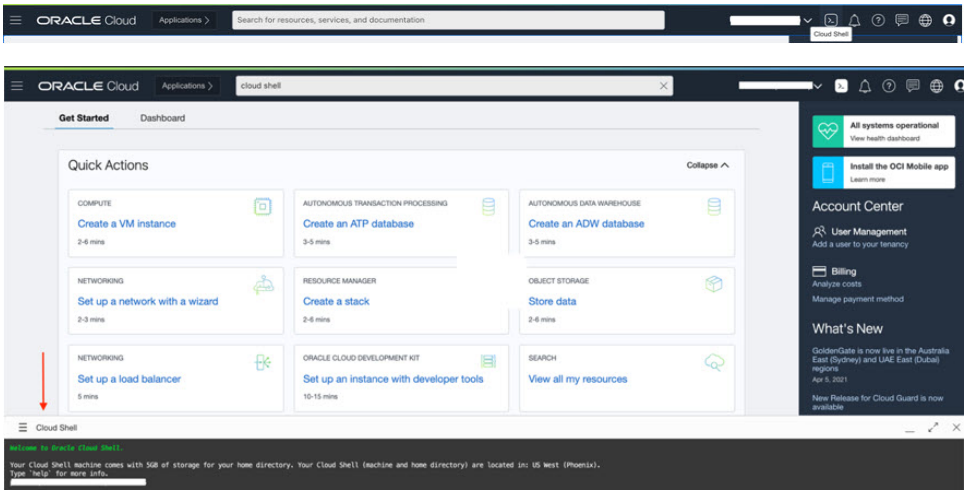
Oracle 関数の展開

（注） この手順は、*Terraform Template-1* の導入が成功した後にのみ実行する必要があります。

OCI では、Oracle 関数は Docker イメージとしてアップロードされ、OCI コンテナレジストリに保存されます。Oracle 関数は、導入時に OCI アプリケーション（Terraform Template-1 で作成）の 1 つにプッシュする必要があります。

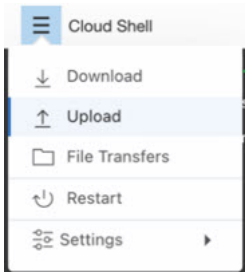
手順

ステップ 1 OCI のクラウドシェルを開きます。



ステップ 2 *deploy_oracle_functions_cloudshell.py* と *Oracle-Functions.zip* をアップロードします。

クラウドシェルのハンバーガーメニューから [アップロード（Upload）] を選択します。



ステップ 3 **ls** コマンドを使用してファイルを確認します。

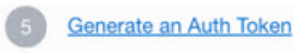
ステップ 4 `python3 Deploy_Oracle_Functions.py -h` を実行します。以下の図に示すように、`deploy_oracle_functions_cloudshell.py` スクリプトには、いくつかの入力パラメータが必要です。詳細は help 引数を使用して確認できます。

```
                                $ python3 Deploy_Oracle_Functions.py -h
usage: Deploy_Oracle_Functions.py [-h] -a  -r  -p  -c  -o  -t

*** Script to deploy Oracle Function for OCI ASAv Autoscale Solution ***

Instruction to find values of required arguments:
Application Name: Name of Application created by first Terraform Template
Region Identifier: OCI -> Administration -> Region Management
Profile Name: OCI -> Profile
Compartment OCID: OCI -> Identity -> Compartment -> Compartment Details
Object Storage Namespace: OCI -> Administration -> Tenancy Details
Authorization Token: OCI -> Identity -> Users -> User Details -> Auth Tokens -> Generate Token

optional arguments:
  -h, --help  show this help message and exit
  -a          Name of Application in OCI to which functions will be deployed
  -r          Region Identifier
  -p          Profile Name of User
  -c          Compartment OCID
  -o          Object Storage Namespace
  -t          Authorization Token for Docker Login (*Please Put in Quotes)
```

スクリプトを実行するには、次の引数を渡します。

表 *1*: 引数と詳細

引数	特記事項
アプリケーション名（**Application Name**）	Terraform Template-1 の導入で作成した OCI アプリケーションの名前です。この値は、Template-1 で付与された「**autoscale_group_prefix**」とサフィックス「**_application**」を組み合わせたものです。
リージョン識別子（**Region Identifier**）	リージョン識別子は、さまざまな地域の OCI で固定された地域コードワードです。 例：フェニックスの場合は「us-phoenix-1」、メルボルンの場合は「ap-melbourne-1」。 すべてのリージョンとそのリージョン識別子のリストを取得するには、**[OCI]** > **[管理（Administration）]** > **[リージョン管理（Region Management）]**に移動します。
プロファイル名（**Profile Name**）	OCI のシンプルなユーザープロファイル名です。 例：*oracleidentitycloudservice/<user> @<mail> .com* 名前は、ユーザーのプロファイルセクションの下にあります。
コンパートメント **OCID**（**Compartment OCID**）	これは、コンパートメントの OCID（Oracle Cloud 識別子）です。ユーザーが OCI アプリケーションを格納しているコンパートメントの OCID。 **[OCI]** > **[アイデンティティ（Identity）]** > **[コンパートメント（Compartment）]** > **[コンパートメントの詳細（Compartment Details）]**に移動します。

引数	特記事項
オブジェクトストレージの名前空間（Object Storage Namespace）	テナントの作成時に作成される一意の識別子です。 **[OCI]** > **[管理（Administration）]** > **[テナントの詳細（Tenancy Details）]**に移動します。
認証トークン（Authorization Token）	これは、OCI コンテナレジストリに Oracle 関数をプッシュすることを許可する Docker ログイン用のパスワードとして使用されます。導入スクリプトでトークンを引用符で囲んで指定します。 **[OCI]** > **[アイデンティティ（Identity）]** > **[ユーザー（Users）]** > **[ユーザの詳細（User Details）]** > **[認証トークン（Auth Tokens）]** > **[トークンの生成（Generate Token）]**に移動します。 何らかの理由でユーザーの詳細が表示されない場合は、**[開発者サービス（Developer services）]** > **[機能（Functions）]**をクリックします。Terraform Template-1 で作成したアプリケーションに移動します。[利用を開始する（Getting Started）] をクリックし、[クラウドシェルの設定（Cloud Shell Setup）] を選択すると、手順を進めていく中で、以下に示すように認証トークンを生成するためのリンクが表示されます。 

ステップ 5 有効な入力引数を渡して、`python3 Deploy_Oracle_Functions.py` コマンドを実行します。すべての機能を展開するには時間がかかります。その後、ファイルを削除してクラウドシェルを閉じることができます。

Terraform Template-2 の展開

Template-2 は、アラーム、関数を呼び出すための ONS トピックなど、アラーム作成に関連するリソースを展開します。Template-2 の展開は、Terraform Template-1 の展開に似ています。

手順

ステップ 1 OCI ポータルにログインします。

地域は、画面の右上隅に表示されます。目的の地域内に存在していることを確認してください。

ステップ 2 **[デベロッパーサービス（Developer Service）]** > **[リソースマネージャ（Resource Manager）]** > **[スタック（Stack）]** > **[スタックの作成（Create Stack）]**の順に選択します。

Terraform 設定のソースとして、ターゲットフォルダにある *Terraform template template2.zip* を選択します。

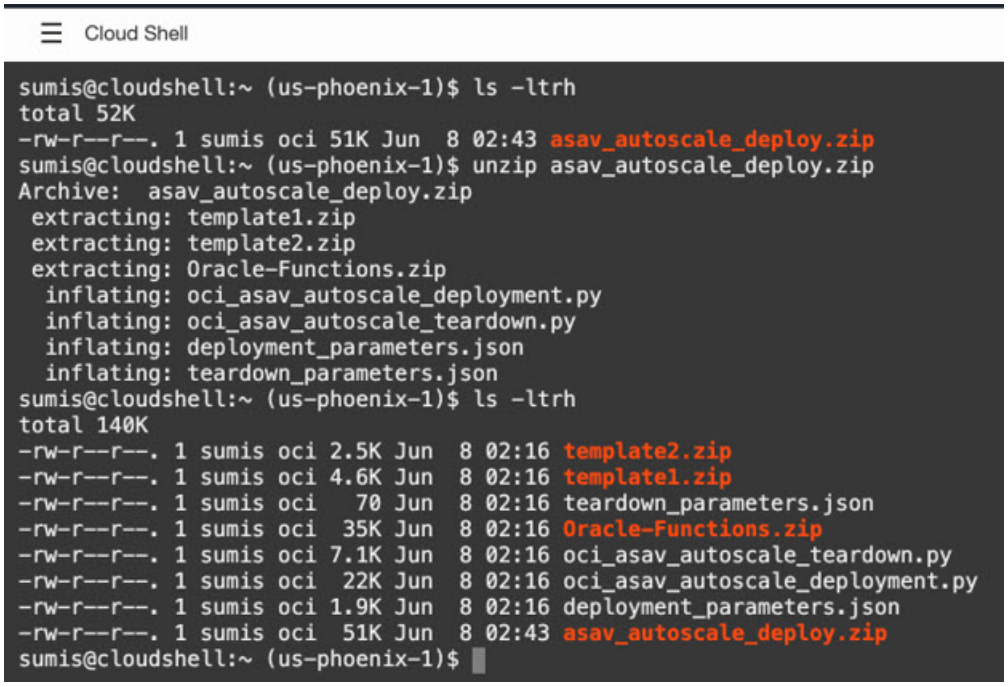
ステップ 3 次のステップで、**Terraformアクション（Terraform Actions）] > [適用（Apply）]**をクリックします。

クラウドシェルを使用した Auto Scale の導入

展開のオーバーヘッドを回避するために、簡単なエンドツーエンドの展開スクリプトを呼び出して、自動スケールソリューション（terraform template1、template2、および Oracle 関数）を展開できます。

手順

ステップ 1 対象フォルダ内にある *asav_autoscale_deploy.zip* ファイルをクラウドシェルにアップロードして、ファイルを抽出します。



ステップ 2 `python3 make.py build` コマンドを実行する前に、*deployment_parameters.json* の入力パラメータが更新されていることを確認してください。

ステップ 3 Auto Scale ソリューションの導入を開始するには、クラウドシェルで `python3 oci_asav_autoscale_deployment.py` コマンドを実行します。

ソリューションの展開が完了するまでに約 10 ～ 15 分かかります。

ソリューションの展開中にエラーが発生した場合、エラーログが保存されます。

展開の検証

すべてのリソースが展開され、Oracle関数がアラームとイベントに接続されているかどうかを検証します。デフォルトでは、インスタンスプールのインスタンスの最小数と最大数はゼロです。OCI UI でインスタンスプールを編集して、必要な最小数と最大数に設定できます。これにより、新しい ASA 仮想 インスタンスがトリガーされます。

1 つのインスタンスのみを起動してワークフローを確認し、そのインスタンスが期待どおりに動作しているかどうかを検証することを推奨します。この検証をポストすると、ASA 仮想 の実際の要件を展開できます。

（注） OCI スケーリングポリシーによる削除を回避するために、最小数の ASA 仮想 インスタンスを**スケールイン保護**として指定します。

アップグレード

Auto Scale スタックのアップグレード

このリリースではアップグレードはサポートされていません。スタックを再導入する必要があります。

ASA 仮想 VM のアップグレード

このリリースでは、ASA 仮想 VM のアップグレードはサポートされていません。必要な ASA 仮想 イメージを使用してスタックを再導入する必要があります。

インスタンスプール

1. インスタンスプール内のインスタンスの最小数と最大数を変更するには、次の手順を実行します。

   [デベロッパーサービス（Developer Services）] > [機能（Function）] > [アプリケーション名（Terraform template-1で作成済み）（Application Name(created by Terraform Template 1)）] > [設定（Configuration）] をクリックします。

   min_instance_count と max_instance_count をそれぞれ変更します。

2. インスタンスの削除/終了は、スケールインと同等ではありません。インスタンスプール内のいずれかのインスタンスがスケールインアクションではなく外部アクションのために削除/終了された場合、インスタンスプールは自動的に新しいインスタンスを開始して回復します。

3. Max_instance _count では、スケールアウトアクションのしきい値制限を定義しますが、UI を介してインスタンスプールのインスタンス数を変更することでしきい値を上回ることができます。UIのインスタンス数が、OCIアプリケーションで設定されたmax_instance_count未満であることを確認します。それ以外の場合は、適切なしきい値に増やします。

4. アプリケーションから直接インスタンスプール内のインスタンスの数を減らしても、プログラムで設定されたクリーンアップアクションは実行されません。両方のロードバランサからバックエンドがドレインおよび削除されないため、ASA 仮想 に供与されているライセンスは失われます。
5. 何らかの理由で、ASA 仮想 インスタンスに異常があり応答せず、一定期間 SSH 経由で到達できない場合、インスタンスがインスタンスプールから強制的に削除され、ライセンスが失われる可能性があります。

Oracle 関数

- Oracle 関数は、実際には Docker イメージです。Docker イメージは、OCI コンテナレジストリのルートディレクトリに保存されます。Docker イメージは削除しないでください。Auto Scale ソリューションで使用される関数も削除されます。
- Terraform Template-1 によって作成された OCI アプリケーションには、Oracle 関数が正しく動作するために必要な重要な環境変数が含まれています。必須でない限り、これらの環境変数の値もフォーマットも変更しないでください。加えられた変更は、新しいインスタンスにのみ反映されます。

ロードバランサのバックエンドセット

OCI でインスタンスプールにロードバランサを関連付ける場合、ASA 仮想 で管理インターフェースとして設定されたプライマリインターフェースを使用した方法のみサポートされています。したがって、内部インターフェイスは内部ロードバランサのバックエンドセットに紐づけられます。外部インターフェイスは、外部ロードバランサのバックエンドセットに紐づけられます。これらの IP はバックエンドセットに自動的に追加されたり、削除されたりしません。Auto Scale ソリューションでは、これら両方のタスクをプログラムで処理します。ただし、外部アクション、メンテナンス、トラブルシューティングの場合は、手動で実行する必要性が生じることがあります。

要件に応じて、リスナーとバックエンドセットを使用して、ロードバランサーで追加のポートを開くことができます。今後のインスタンス IP はバックエンドセットに自動的に追加されますが、既存のインスタンス IP は手動で追加する必要があります。

ロードバランサでのリスナーの追加

ロードバランサでポートをリスナーとして追加するには、**[OCI]** > **[ネットワーキング（Networking）]** > **[ロードバランサ（Load Balancer）]** > **[リスナー（Listener）]** > **[リスナーの作成（Create Listener）]**に移動します。

バックエンドをバックエンドセットに登録

ASA 仮想 インスタンスをロードバランサに登録するには、ASA 仮想 インスタンスの外部インターフェイス IP を外部ロードバランサのバックエンドセットでバックエンドとして設定する必要があります。内部インターフェイス IP は、内部ロードバランサーのバックエンドセットでバックエンドとして設定する必要があります。使用しているポートがリスナーに追加されていることを確認してください。

OCI のAuto Scale 設定の削除

Terraform を使用して導入されたスタックは、OCI の Resource Manager を使用して、同じ方法で削除できます。スタックを削除すると、そのスタックによって作成されたすべてのリソースが削除され、これらのリソースに関連付けられているすべての情報が完全に削除されます。

（注） スタックを削除する場合は、インスタンスプールのインスタンスの最小数を0にして、インスタンスが終了するまで待つことを推奨します。そうすることで、すべてのインスタンスの削除が容易になり、インスタンスが残りません。

手動で削除するか、クラウドシェルを使用できます。

手動による削除

エンドツーエンドの Auto Scale ソリューションの削除は、次の 3 つの手順で構成されます。Terraform Template-2 スタックの削除、Oracle 関数の削除、次いで Terraform Template-1 スタックの削除

Terraform Template-2 スタックの削除

自動スケール設定を削除するには、最初に Terraform Template-2 スタックを削除する必要があります。

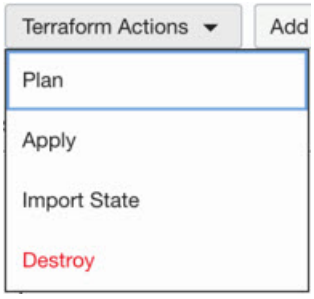
手順

ステップ 1 OCI ポータルにログインします。

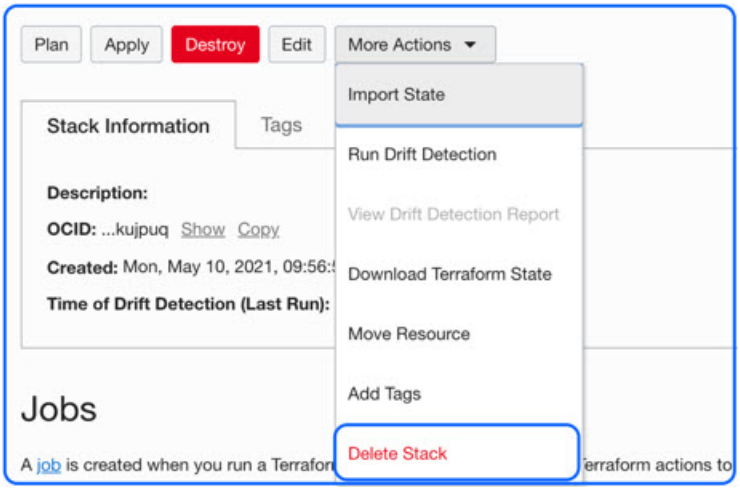
地域は、画面の右上隅に表示されます。目的の地域内に存在していることを確認してください。

ステップ 2 **[デベロッパーサービス（Developer Service）] > [リソースマネージャ（Resource Manager）] > [スタック（Stack）]**の順に選択します。

ステップ 3 Terraform Template-2 によって作成されたスタックを選択し、次の図に示すように [Terraformアクション（Terraform Actions）] ドロップダウンメニューで [破棄（Destroy）] を選択します。



破棄ジョブが作成されます。リソースが順次削除されるまで時間がかかります。破棄ジョブが完了したら、下の図に示すようにスタックを削除できます。



ステップ 4 Oracle 関数の削除に進みます。

Oracle 関数の削除

Oracle 関数の展開は Terraform Template スタック展開の一部としてではなく、クラウドシェルを使用して個別にアップロードします。したがって、削除も Terraform スタックの削除ではサポートされていません。Terraform Template-1 によって作成された OCI アプリケーション内のすべての Oracle 関数を削除する必要があります。

手順

ステップ 1 OCI ポータルにログインします。

地域は、画面の右上隅に表示されます。目的の地域内に存在していることを確認してください。

ステップ 2 **[開発者サービス（Developer Services）]** > **[機能（Functions）]**の順に選択します。Template-1 スタックで作成されたアプリケーション名を選択します。

ステップ 3 このアプリケーション内で各機能にアクセスして削除します。

Terraform Template-1 スタックの削除

（注） Template-1 スタックの削除は、すべての Oracle 関数を削除した後にのみ成功します。

Terraform Template-2 の削除と同じです。

手順

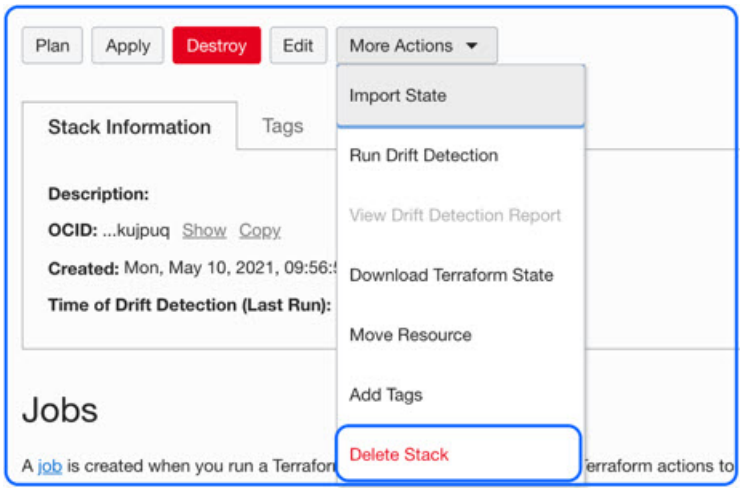
ステップ 1 OCI ポータルにログインします。

地域は、画面の右上隅に表示されます。目的の地域内に存在していることを確認してください。

ステップ 2 **[デベロッパーサービス（Developer Service）] > [リソースマネージャ（Resource Manager）] > [スタック（Stack）]**の順に選択します。

ステップ 3 Terraform Template-2 によって作成されたスタックを選択し、[Terraformアクション（Terraform Actions）] ドロップダウンメニューで [破棄（Destroy）] を選択します。破棄ジョブが作成されます。リソースが順次削除されるまで時間がかかります。

ステップ 4 破棄ジョブが完了したら、下の図に示すように、[その他の操作（More Actions）] ドロップダウンメニューからスタックを削除できます。



Terraform Template-1 スタックの削除が成功したら、すべてのリソースが削除され、残存しているリソースがないことを確認する必要があります。

クラウドシェルを使用した Auto Scale の削除

スクリプトを使用してスタックやオラクル関数を削除するには、 コマンドシェルで `python3 oci_asav_autoscale_teardown.py` コマンドを実行します。スタックが手動で展開されている場合は、stack1 と stack2 のスタック ID を更新し、*teardown_parameters.json* ファイルのアプリケーション ID を更新します。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。