デベロッパーガイド

AWS Blockchain Templates

AWS Blockchain Templates: デベロッパーガイド

Table of Contents

AWS Blockchain Templates は 2019 年 4 月 30 日に廃止されました。このサービスやサポートドキュメントは今後更新されません。で最高の Managed Blockchain エクスペリエンスを得るには AWS、Amazon Managed Blockchain (AMB) を使用することをお勧めします。Amazon Managed Blockchain の開始方法の詳細については、Hyperledger Fabric に関するワークショップ、またはイーサリアムノードのデプロイに関するブログをご覧ください。AMB に関するご質問やサポートが必要な場合は、 または AWS アカウントチームにお問い合わせください サポート。

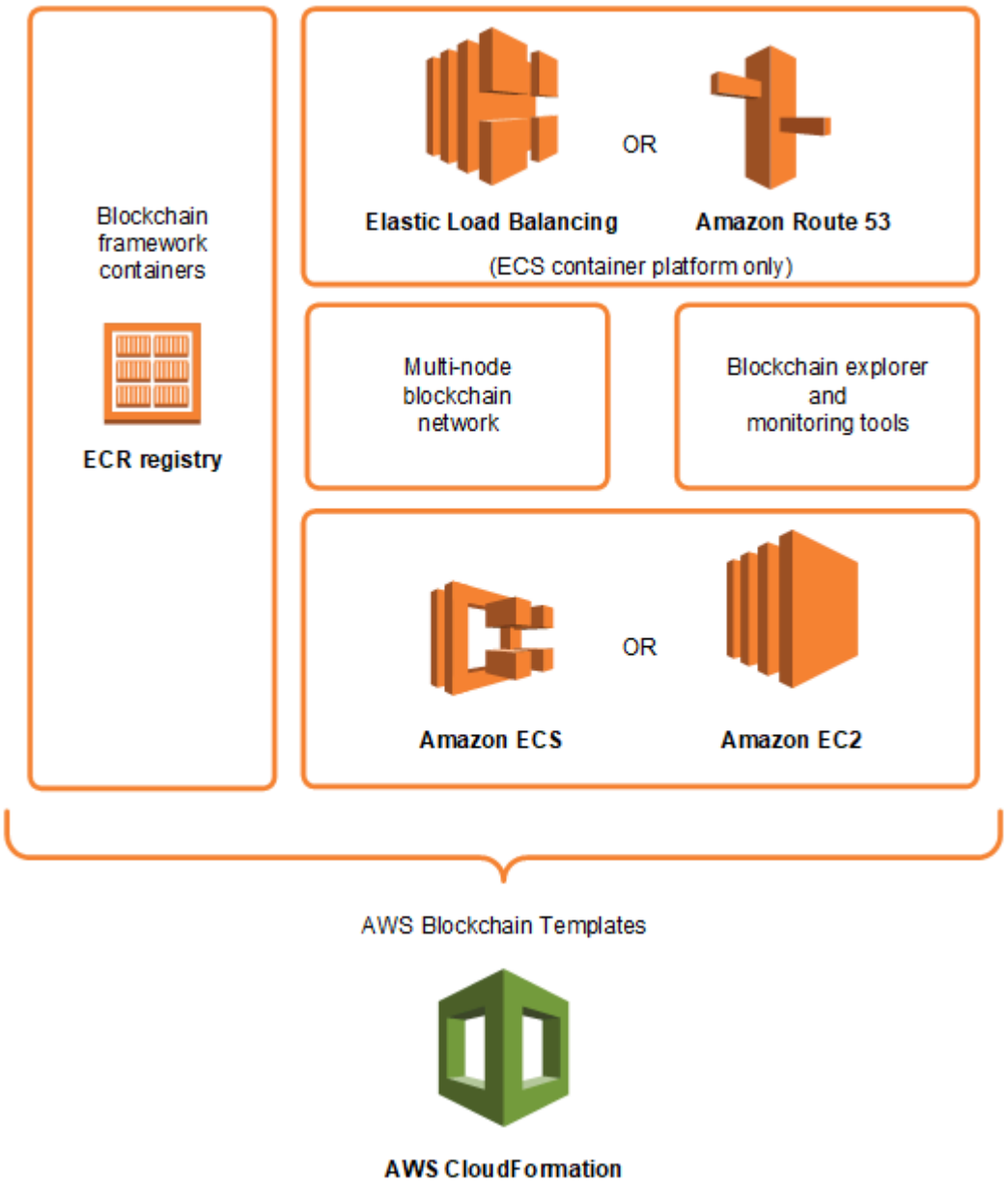
翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。

AWS Blockchain Templates とは

AWS Blockchain Templates を使用すると、さまざまなブロックチェーンフレームワーク AWS を使用して でブロックチェーンネットワークをすばやく作成してデプロイできます。ブロックチェーンは、絶えず増大するトランザクションやスマートコントラクトのセットを改ざんや変更に対して暗号化で強化する、分散型データベーステクノロジーです。

ブロックチェーンネットワークは、国外への支払い、供給チェーンの管理、土地の登録、クラウドファンディング、ガバナンス、金融取引などのビジネスプロセスの効率とイミュータビリティを高める、ピアツーピアネットワークです。これにより、お互いに知らない人や組織でも信頼し、トランザクションレコードを個別に確認できます。

AWS Blockchain Templates を使用して、ブロックチェーンネットワークを作成するための CloudFormation スタックを設定および起動します。どの AWS リソースとサービスを使用するかは、選択する AWS Blockchain Template と指定するオプションによって異なります。使用可能なテンプレートとその機能の詳細については、「AWS Blockchain Templates と機能」を参照してください。AWS Blockchain Templates を使用して AWS 作成された のブロックチェーンネットワークの基本コンポーネントを次の図に示します。



開始方法

開始する最適な場所は、ブロックチェーンに関する専門知識のレベル、 AWS特に AWS Blockchain Templates に関連するサービスによって異なります。

AWS とブロックチェーンに習熟している

使用するフレームワークについての「AWS Blockchain Templates と機能」のトピックから始めてください。リンクを使用して AWS Blockchain Template を起動し、ブロックチェーンネットワークを構成するか、テンプレートをダウンロードして自分でチェックアウトします。

ブロックチェーンに習熟 AWS していて、ブロックチェーンを初めて使用する

「AWS Blockchain Templates の開始方法」のチュートリアルから始めてください。このチュートリアルでは、デフォルト設定を使用して入門用の Ethereum ブロックチェーンネットワークを作成する方法について説明しています。終了したら、ブロックチェーンフレームワークの概要リンクについて「AWS Blockchain Templates と機能」を参照して、設定の選択と機能の詳細について学んでください。

ブロックチェーンの初心者 AWS で習熟している

「AWS Blockchain Templates のセットアップ」から開始してください。これにより AWS、アカウントやユーザープロファイルなどの の基礎をセットアップできます。次に、「AWS Blockchain Templates の開始方法」のチュートリアルを参照してください。このチュートリアルでは、入門用の Ethereum ブロックチェーンネットワークを作成する方法について説明しています。たとえ最終的に Ethereum を使用しない場合でも、関連するサービスをセットアップする実践的な経験が得られます。この経験はすべてブロックチェーンフレームワークに役立ちます。最後に、フレームワークの「AWS Blockchain Templates と機能」セクションのトピックを参照してください。

AWS とブロックチェーンを初めて使用する

「AWS Blockchain Templates のセットアップ」から開始してください。これにより AWS、アカウントやユーザープロファイルなどの の基礎をセットアップできます。「AWS Blockchain Templates の開始方法」のチュートリアルを参照してください。このチュートリアルでは、入門用の Ethereum ブロックチェーンネットワークを作成する方法について説明しています。サービス AWS および Ethereum の詳細については、リンクを参照してください。

関連 サービス

選択したオプションに応じて、AWS Blockchain Templates は次の AWS サービスを使用してブロックチェーンをデプロイできます。

- Amazon EC2 — ブロックチェーンネットワークのコンピューティング性能を提供します。詳細については、「Amazon EC2 ユーザーガイド」を参照してください。
- Amazon ECS — ブロックチェーンネットワークを使用するように選択した場合は、クラスター内の EC2 インスタンス間でコンテナのデプロイを調整します。詳細については、Amazon Elastic Container Service デベロッパーガイドを参照してください。

- Amazon VPC — 作成した Ethereum リソースへのネットワークアクセスを提供します。アクセシビリティとセキュリティの設定をカスタマイズできます。詳細については、Amazon VPC デベロッパーガイドを参照してください。
- Application Load Balancing — Amazon ECS をコンテナプラットフォームとして使用する場合、使用可能なユーザーインターフェイスおよび内部サービス検出への単一のアクセスポイントとして機能します。詳細については、Application Load Balancers ユーザーガイドの Application Load Balancer とはを参照してください。

AWS Blockchain Templates のセットアップ

AWS Blockchain Templates の使用を開始する前に、次のタスクを完了します。

- キーペアの作成

これらは、すべてのブロックチェーン構成の基本的な前提条件です。さらに、選択したブロックチェーンネットワークには、必要な環境と構成の選択に応じて異なる前提条件が存在する場合があります。詳細については、「AWS Blockchain Templates と機能」のブロックチェーンテンプレートの関連セクションを参照してください。

Amazon ECS クラスターを使用してプライベート Ethereum ネットワークの前提条件を設定するためのステップバイステップの手順については、AWS Blockchain Templates の開始方法を参照してください。

にサインアップする AWS アカウント

の使用を開始するには AWS、 が必要です AWS アカウント。の作成の詳細については AWS アカウント、「 AWS アカウント管理 リファレンスガイド」の「 の開始方法 AWS アカウント」を参照してください。

キーペアの作成

AWS はパブリックキー暗号化を使用して、ブロックチェーンネットワーク内のインスタンスのログイン情報を保護します。各 AWS Blockchain Template を起動するときに、キーペアの名前を指定します。次に、SSH を使用してログインするなど、キーペアを使用してインスタンスに直接アクセスできます。

適切なリージョンで既存のキーペアがある場合は、このステップを省略できます。キーペアをまだ作成していない場合は、Amazon EC2 コンソールを使用して作成できます。Ethereum ネットワークを起動するために使用するのと同じリージョンで、キーペアを作成します。詳細については、Amazon EC2 ユーザーガイドの「リージョンとアベイラビリティゾーン」を参照してください。

キーペアを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。

2. ナビゲーションバーで、キーペアを生成するリージョンを選択します。使用可能なリージョンは場所に関係なく選択できますが、キーペアはリージョンに固有のものです。例えば、米国東部 (オハイオ) リージョンでインスタンスを起動する予定の場合は、その同じリージョン内のインスタンス用にキーペアを作成する必要があります。
3. ナビゲーションペインで キーペア] を選択し、キーペアの作成] を選択します。
4. [キーペア名] に、新しいキーペアの名前を入力します。覚えやすい名前 (IAM ユーザー名など) を選択し、その後に `-key-pair` を続け、さらにリージョン名を続けます。たとえば、me-key-pair-useast2 とします。[作成] を選択します。
5. ブラウザによって秘密キーファイルが自動的にダウンロードされます。ベースファイル名はキーペア名として指定した名前であり、ファイル名の拡張子は `.pem` です。ダウンロードしたプライベートキーのファイルを安全な場所に保存します。

> **Important**
>
> プライベートキーのファイルを保存できるのはこのタイミングだけです。Ethereum ネットワークの起動時に、キーペアの名前を指定します。

詳細については「Amazon EC2 ユーザーガイド」の「Amazon EC2 キーペア」を参照してください。キーペアを使用して EC2 インスタンスに接続する方法の詳細については、Amazon EC2 ユーザーガイド」の「Linux インスタンスに接続する」を参照してください。

AWS Blockchain Templates の開始方法

このチュートリアルでは、AWS Blockchain Template for Ethereum を使用して、 AWS を介してプライベートブロックチェーンネットワークを作成する方法を示します CloudFormation。作成するネットワークでは、2 つの Ethereum クライアントと 1 つのマイナーが Amazon ECS クラスターの Amazon EC2 インスタンス上で実行されています。Amazon ECS は Amazon ECR から引き出された Docker コンテナでこれらのサービスを実行します。このチュートリアルを開始する前に、ブロックチェーンネットワークと関連する AWS サービスについて知っておくと便利ですが、必須ではありません。

このチュートリアルでは、「AWS Blockchain Templates のセットアップ」の一般的な前提条件を設定していることを前提としています。さらに、テンプレートを使用する前に、Amazon VPC ネットワークや IAM ロールの特定のアクセス許可などの一部の AWS リソースを設定する必要があります。

このチュートリアルで、これらの前提条件を設定する方法を示します。設定オプションは選択済みですが、これらに限定されるものではありません。前提条件を満たす限り、アプリケーションや環境のニーズに応じて他の設定を選択できます。各テンプレートの機能と一般的な前提条件、テンプレートをダウンロードする方法やテンプレートを CloudFormationで直接起動する方法については、「AWS Blockchain Templates と機能」を参照してください。

このチュートリアルの例では米国西部 (オレゴン) リージョン (us-west-2) を使用していますが、AWS Blockchain Templates をサポートする任意のリージョンを使用できます。

- 米国西部 (オレゴン) リージョン (us-west-2)
- 米国東部 (バージニア北部) リージョン (us-east-1)
- 米国東部 (オハイオ) リージョン (us-east-2)

> **Note**
>
> 上記に記載されていないリージョンでテンプレートを実行すると、米国東部 (バージニア北部) リージョン (us-east-1) でリソースが起動します。

このチュートリアルで設定する Ethereum 用の AWS Blockchain Template では、以下のリソースを作成します。

- 指定するタイプと数のオンデマンド EC2 インスタンス。このチュートリアルでは、デフォルトの t2.medium インスタンスタイプを使用します。
- 内部 Application Load Balancer。

チュートリアルの後で、作成したリソースをクリーンアップするための手順が用意されています。

トピック

前提条件の設定

このチュートリアルで指定する Ethereum 用の AWS Blockchain Template 設定では、次のことを行う必要があります。

- VPC とサブネットを作成する
- セキュリティグループを作成する
- Amazon ECS および EC2 インスタンスプロファイルの IAM ロールを作成する
- 要塞ホストの作成

VPC とサブネットを作成する

Ethereum 用の AWS Blockchain Template は、Amazon Virtual Private Cloud (Amazon VPC) を使用して、定義した仮想ネットワーク内でリソースを起動します。このチュートリアルで指定する設定では、2 つのパブリックサブネットを必要とする Application Load Balancer を作成します。各サブネットは別個のアベイラビリティーゾーンに配置する必要があります。さらに、コンテナインスタンス用にプライベートサブネットも必要です。このサブネットは Application Load Balancer と同じアベイラビリティーゾーンに配置する必要があります。最初に VPC ウィザードを使用して、1 つのパブリックサブネットとプライベートサブネットを同じアベイラビリティーゾーンに作成します。次に、この VPC の 2 つ目のパブリックサブネットを別のアベイラビリティーゾーンに作成します。

詳細については、「Amazon VPC ユーザーガイド」の「Amazon VPC とは」を参照してください。

以下で説明するように、Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を使用して、Elastic IP アドレス、VPC、およびサブネットを作成します。

Elastic IP アドレスを作成するには

1. Amazon VPC コンソールの https://console.aws.amazon.com/vpc/ を開いてください。
2. [Elastic IP]、[新しいアドレスの割り当て]、[割り当て] の順に選択します。
3. 作成した Elastic IP アドレスを書き留め、[閉じる] を選択します。
4. Elastic IP アドレスのリストで、先ほど作成した Elastic IP アドレスの [割り当て ID] を見つけます。VPC を作成するときに、これを使用します。

VPC を作成するには

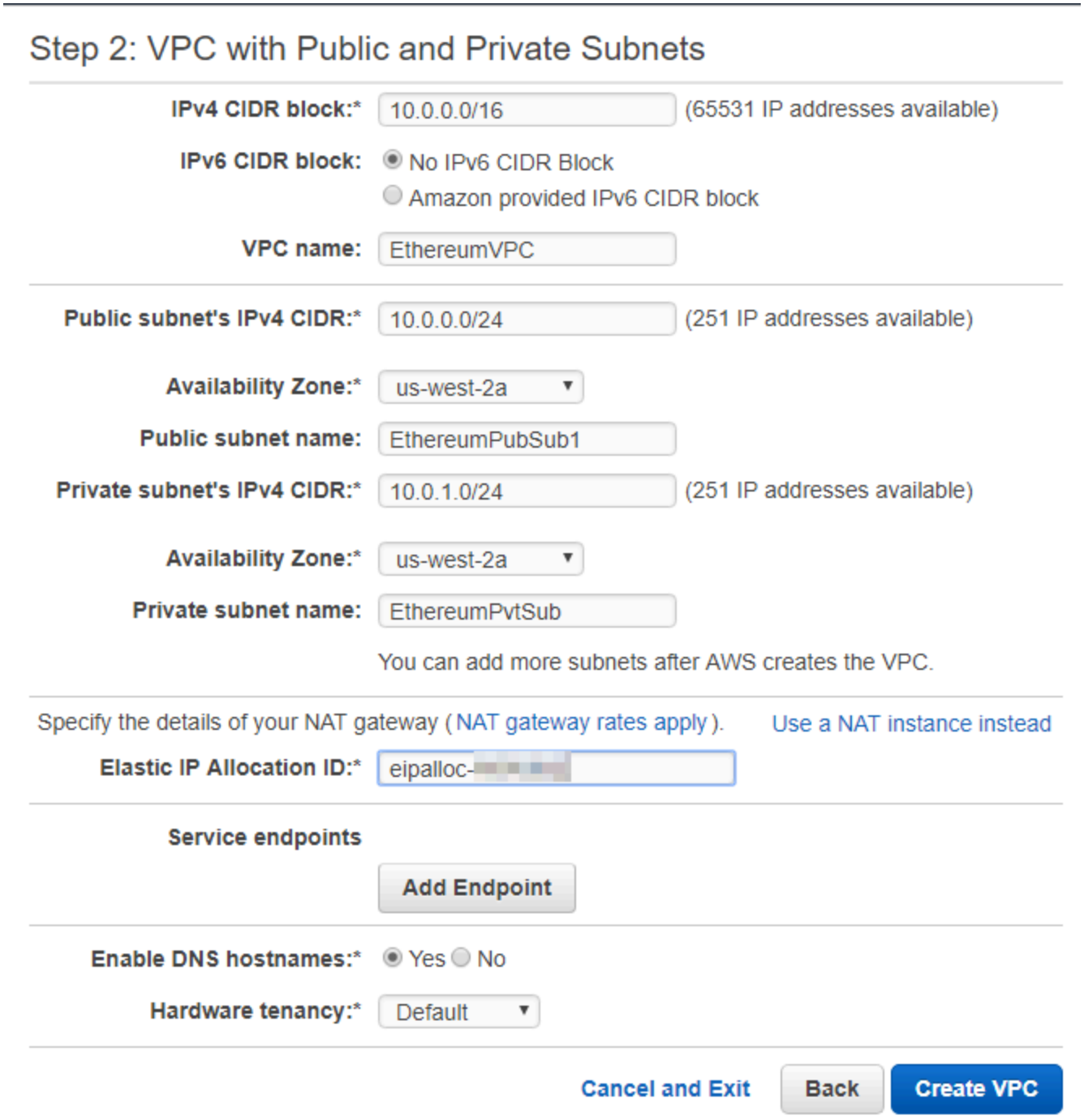
1. ナビゲーションバーで、VPC のリージョンを選択します。VPC はリージョンに固有であるため、キーペアを作成し、Ethereum スタックを起動するのと同じリージョンを選択します。詳細については、「キーペアの作成」を参照してください。
2. VPC ダッシュボードで、[Start VPC Wizard] (VPC ウィザードの起動) を選択します。
3. [ステップ 1: VPC 設定の選択] ページで [パブリックとプライベートサブネットを持つ VPC] を選択し、[選択] を選択します。
4. [ステップ 2: パブリックとプライベートサブネットを持つ VPC] ページで、[IPv4 CIDR ブロック] と [IPv6 CIDR ブロック] はデフォルト値のままにします。[VPC 名] に、わかりやすい名前を入力します。
5. [パブリックサブネットの IPv4 CIDR] は、デフォルト値のままにしておきます。[アベイラビリティーゾーン] でゾーンを選択します。[パブリックサブネット名] にわかりやすい名前を入力します。

   このサブネットを、テンプレートを使用するときに Application Load Balancer の 2 つのサブネットの最初の 1 つとして指定します。

   プライベートサブネットに同じアベイラビリティーゾーンを選択し、他のパブリックサブネットに別のアベイラビリティーゾーンを選択するため、このサブネットのアベイラビリティーゾーンを書き留めておきます。
6. [プライベートサブネットの IPv4 CIDR] は、デフォルト値のままにしておきます。[アベイラビリティーゾーン] で、前のステップと同じアベイラビリティーゾーンを選択します。[プライベートサブネット名] に、わかりやすい名前を入力します。
7. [Elastic IP 割り当て ID] で、前に作成した Elastic IP アドレスを選択します。

8. 他の設定はデフォルト値のままにします。
9. [Create VPC (VPC の作成)] を選択します。

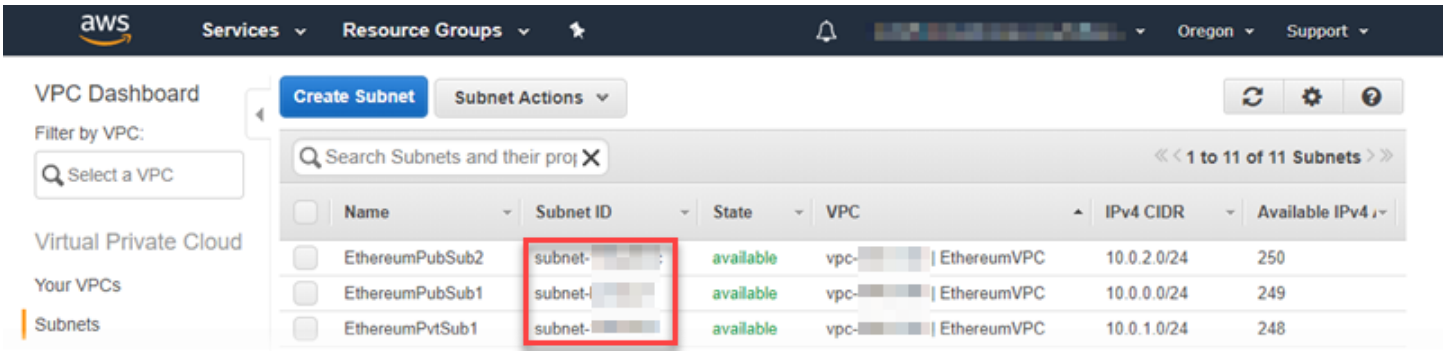
   以下の例に示しているのは、パブリックサブネット EthereumPubSub1 およびプライベートサブネット EthereumPvtSub1 を持つ VPC EthereumNetworkVPC です。パブリックサブネットでは、アベイラビリティーゾーン [us-west-2a] を使用します。



別のアベイラビリティーゾーンに 2 つ目のパブリックサブネットを作成するには

1. [サブネット] を選択し、前に作成したパブリックサブネットをリストから選択します。[ルートテーブル] タブを選択し、ルートテーブル ID を書き留めます。次の 2 番目のパブリックサブネットに、この同じルートテーブルを指定します。
2. [Create Subnet] を選択します。
3. [名前タグ] に、サブネットの名前を入力します。この名前は、後でこのネットワークに要塞ホストを作成するときに使用します。
4. [VPC] で、前に作成した VPC を選択します。
5. [アベイラビリティーゾーン] で、最初のパブリックサブネット用に選択したゾーンとは異なるゾーンを選択します。
6. [IPv4 CIDR ブロック] に「10.0.2.0/24」と入力します。
7. [はい、作成する] を選択します。サブネットがサブネットのリストに追加されます。
8. リストからサブネットを選択した状態で、[サブネットのアクション]、[自動割り当て IP 設定の変更] の順に選択します。[自動割り当て IP]、[保存]、[閉じる] の順に選択します。これにより、要塞ホストは、このサブネットで作成されたパブリック IP アドレスを取得できます。
9. [ルートテーブル] タブで [編集] を選択します。[変更先] で、前に書き留めたルートテーブル ID を選択し、[保存] を選択します。

これにより、先ほど作成した VPC のサブネットが 3 つ表示されます。各サブネットの名前と ID を書き留め、テンプレートの使用時に指定できるようにします。



セキュリティグループを作成する

セキュリティグループはファイアウォールとして機能し、リソースへ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コントロールします。テンプレートを使用して Amazon ECS クラスター上に Ethererum ネットワークを作成するときは、2 つのセキュリティグループを指定します。

- EC2 インスタンスのセキュリティグループは、クラスター内にある EC2 インスタンスに出入りするトラフィックをコントロールします。
- Application Load Balancer、EC2 インスタンス、および要塞ホスト間のトラフィックを制御する Application Load Balancer 用のセキュリティグループ。このセキュリティグループを要塞ホストにも関連付けます。

各セキュリティグループには、Application Load Balancer と EC2 インスタンス間の通信、およびその他の最小ルールを可能にするルールがあります。これには、セキュリティグループが互いに参照する必要があります。このため、最初にセキュリティグループを作成し、適切なルールを使用して更新します。

2 つのセキュリティグループを作成するには

1. Amazon EC2 コンソールの https://console.aws.amazon.com/ec2/ を開いてください。
2. ナビゲーションペインで [Security Groups] (セキュリティグループ) を選択して、[Create Security Group] (セキュリティグループの作成) を選択します。
3. [セキュリティグループ名] に、わかりやすいセキュリティグループ名を入力します。[EthereumEC2-SG] や [EthereumALB-SG] など、他のセキュリティグループと区別できる名前にします。これらの名前は後で使用します。[説明] に、簡単な概要を入力します。
4. [VPC] で、前に作成した VPC を選択します。
5. [作成] を選択します。
6. 上記の手順を繰り返して、他のセキュリティグループを作成します。

EC2 インスタンスのセキュリティグループにインバウンドルールを追加する

1. 先ほど作成した EC2 インスタンスのセキュリティグループを選択します。
2. [インバウンド] タブで、[編集] を選択します。
3. [タイプ] で、[すべてのトラフィック] を選択します。[ソース] で [カスタム] を選択したままにし、現在編集しているセキュリティグループをリストから選択します (例: EthereumEC2-SG)。これにより、セキュリティグループ内の EC2 インスタンスが相互に通信できるようになります。
4. [ルールの追加] を選択します。
5. [タイプ] で、[すべてのトラフィック] を選択します。[ソース] で [カスタム] を選択したままにし、Application Load Balancer のセキュリティグループをリストから選択します (例:

EthereumALB-SG)。これにより、セキュリティグループ内の EC2 インスタンスが Application Load Balancer と通信できるようになります。

6. [保存] を選択します。

Application Load Balancer のセキュリティグループのインバウンドルールとアウトバウンドルールを追加する

1. 先ほど作成した Application Load Balancer インスタンスのセキュリティグループを選択します。
2. [インバウンド] タブで [編集] を選択し、次のインバウンドのルールを追加します。
   a. [タイプ] で、[すべてのトラフィック] を選択します。[ソース] で [カスタム] を選択したままにし、現在編集しているセキュリティグループをリストから選択します (例: EthereumALB-SG)。これにより、Application Load Balancer はそれ自体および要塞ホストと通信できるようになります。
   b. [ルールの追加] を選択します。
   c. [タイプ] で、[すべてのトラフィック] を選択します。[ソース] で [カスタム] を選択したままにし、EC2 インスタンスのセキュリティグループをリストから選択します (例: EthereumEC2-SG)。これにより、セキュリティグループ内の EC2 インスタンスは Application Load Balancer および要塞ホストと通信できるようになります。
   d. [ルールの追加] を選択します。
   e. [タイプ] で SSH] を選択してください。[ソース] で [マイ IP] を選択します。これにより、コンピュータの IP CIDR が検出されて入力されます。

      > **Important**
      >
      > このルールにより、コンピュータからの SSH トラフィックを受け入れることを要塞ホストに許可します。また、要塞ホストを使用してウェブインターフェイスを表示したり、Ethereum ネットワークの EC2 インスタンスに接続したりすることをコンピュータに許可します。Ethereum ネットワークに接続することを他のコンピュータに許可するには、それらのコンピュータをソースとして、このルールに追加します。信頼されたソースへのインバウンドトラフィックのみを許可します。

   f. [保存] を選択します。
3. [アウトバウンド] タブで、[編集] を選択し、すべての IP アドレスへのアウトバウンドトラフィックを許可するように自動的に作成されたルールを削除します。

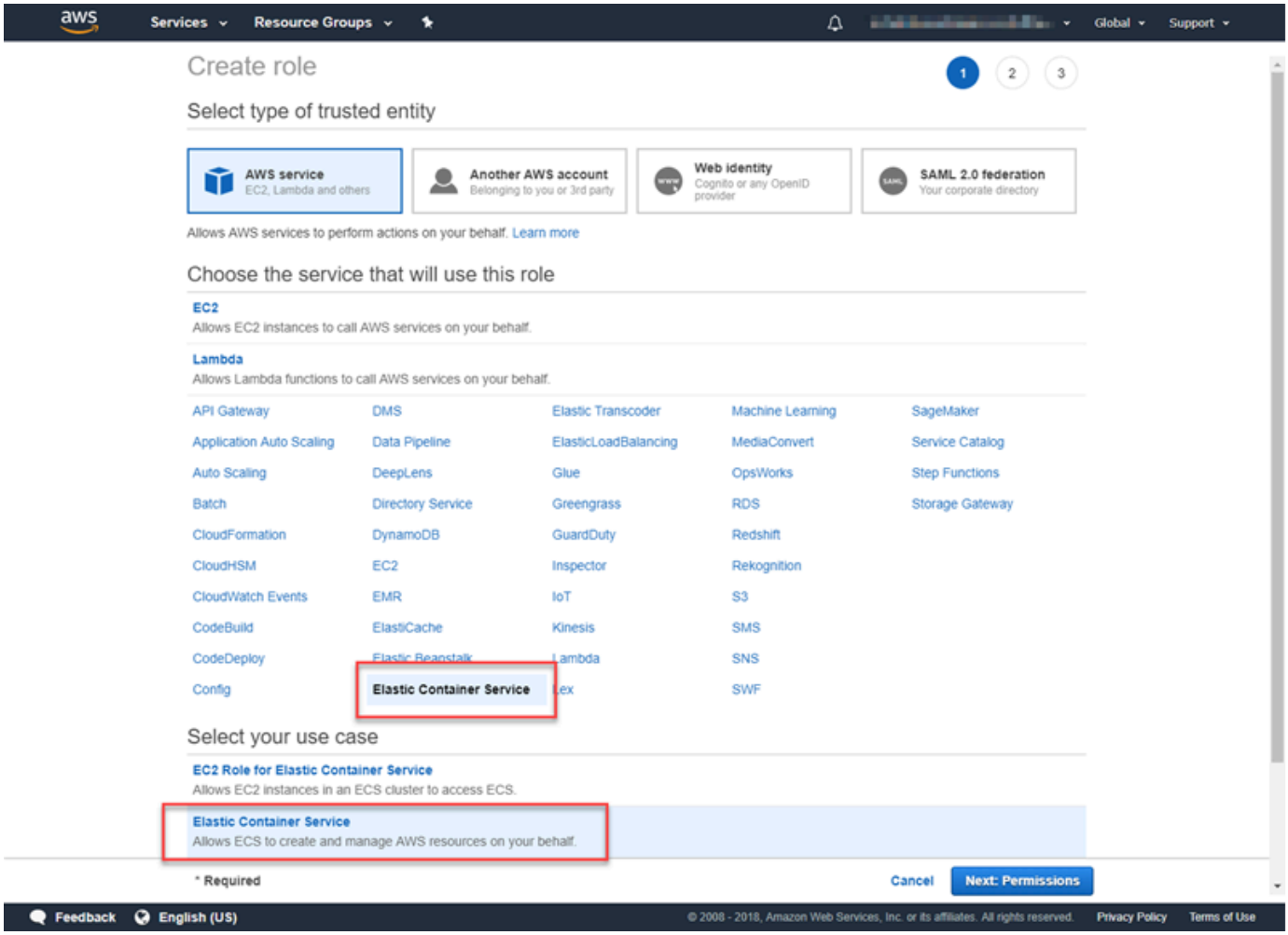
4. [ルールの追加] を選択します。
5. [タイプ] で、[すべてのトラフィック] を選択します。[送信先] で [カスタム] を選択したままにし、EC2 インスタンスのセキュリティグループをリストから選択します。これにより、Application Load Balancer および要塞ホストから Ethereum ネットワーク内の EC2 インスタンスへのアウトバウンド接続が可能になります。
6. [ルールの追加] を選択します。
7. [タイプ] で、[すべてのトラフィック] を選択します。[ソース] で [カスタム] を選択したままにし、現在編集しているセキュリティグループをリストから選択します (例: EthereumALB-SG)。これにより、Application Load Balancer はそれ自体および要塞ホストと通信できるようになります。
8. [保存] を選択します。

Amazon ECS および EC2 インスタンスプロファイルの IAM ロールを作成する

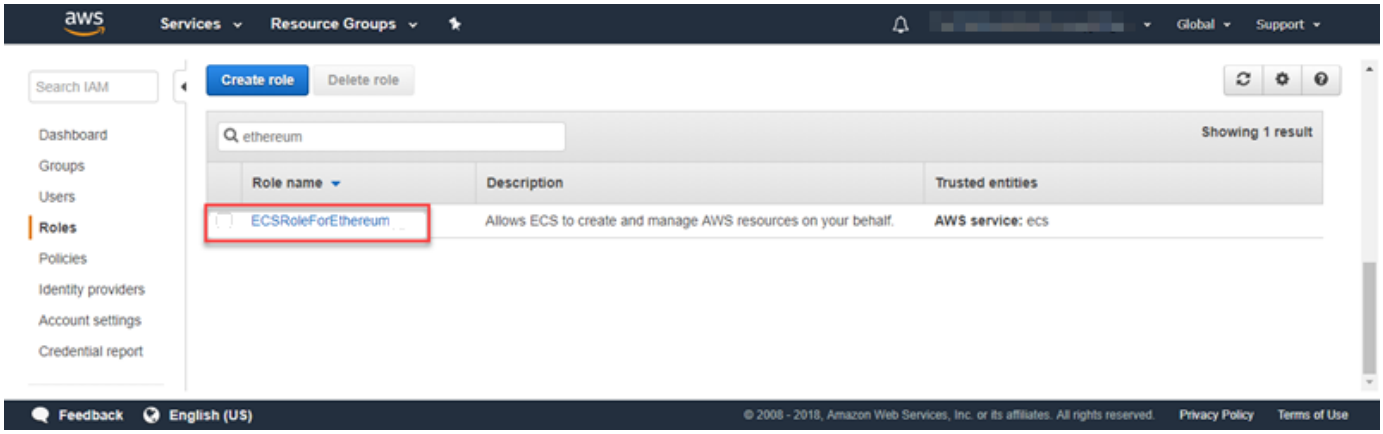
このテンプレートを使用するときは、Amazon ECS および EC2 インスタンスプロファイルの IAM ロールを指定します。これらのロールにアタッチされたアクセス権限ポリシーにより、クラスターの AWS リソースとインスタンスは、他の AWS リソースとやり取りすることができます。詳細については、IAM ユーザーガイドの IAM ロールを参照してください。IAM コンソール (https://console.aws.amazon.com/iam/) を使用して、Amazon ECS および EC2 インスタンスプロファイルの IAM ロールを設定します。

Amazon ECS の IAM ロールを作成するには

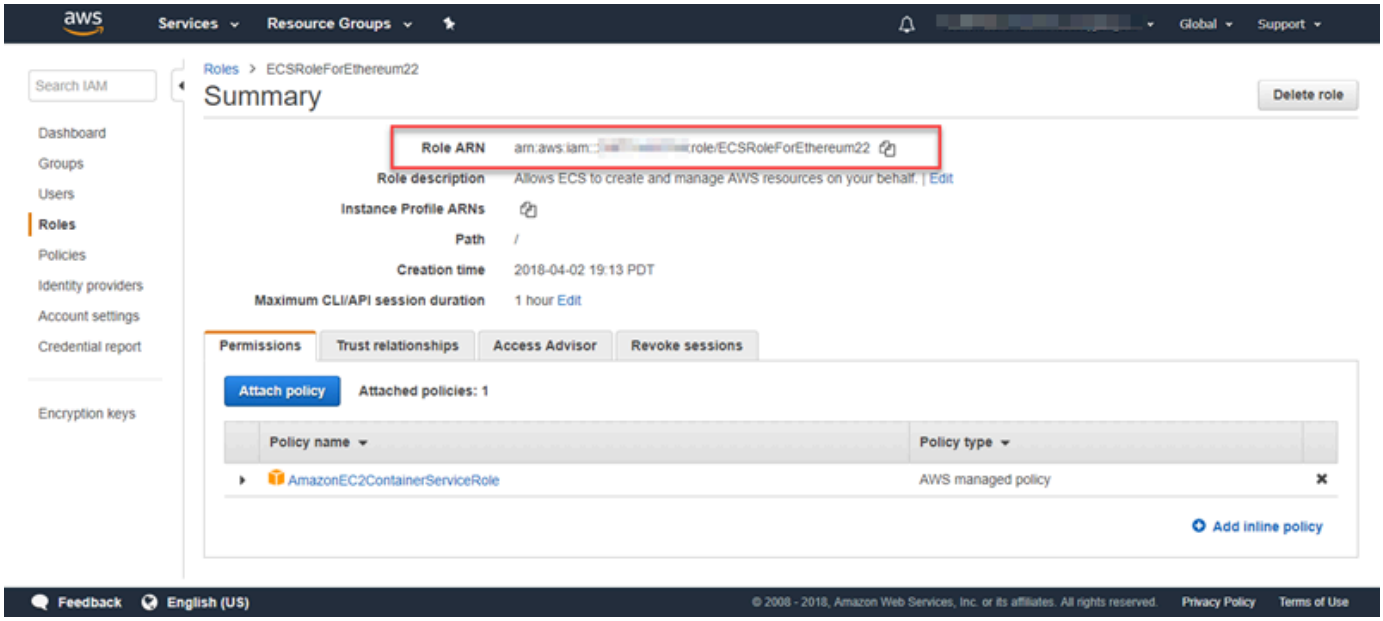
1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで [Roles (ロール)]、[Create Role (ロールの作成)] の順に選択します。
3. [信頼されたエンティティの種類を選択] で、[AWS のサービス] を選択します。
4. [Choose the service that will use this role] (このロールを使用するサービスを選択) で、[Elastic Container Service] (伸縮自在コンテナサービス) を選択します。
5. [ユースケースの選択] で [Elastic Container Service] を選択し、[Next: Permissions (次の手順: アクセス許可)] を選択します。



6. [Permissions policy (アクセス許可ポリシー)] で、デフォルトのアクセス許可ポリシー (AmazonEC2ContainerServiceRole) を選択したままにし、[Next:Review (次の手順: 確認)] を選択します。
7. [ロール名] に、ロールを識別するのに役立つ値を入力します (例: ECSRoleForEthereum)。[ロールの説明] に、簡単な要約を入力します。後で使用するため、ロール名を書き留めておきます。
8. [ロールの作成] を選択してください。
9. リストから、作成したロールを選択します。アカウントに多数のロールがある場合は、ロール名で検索できます。



10. [ロールの ARN] の値をコピーし、再度コピーできるように保存します。Ethereum ネットワークを作成するときに、この ARN が必要です。



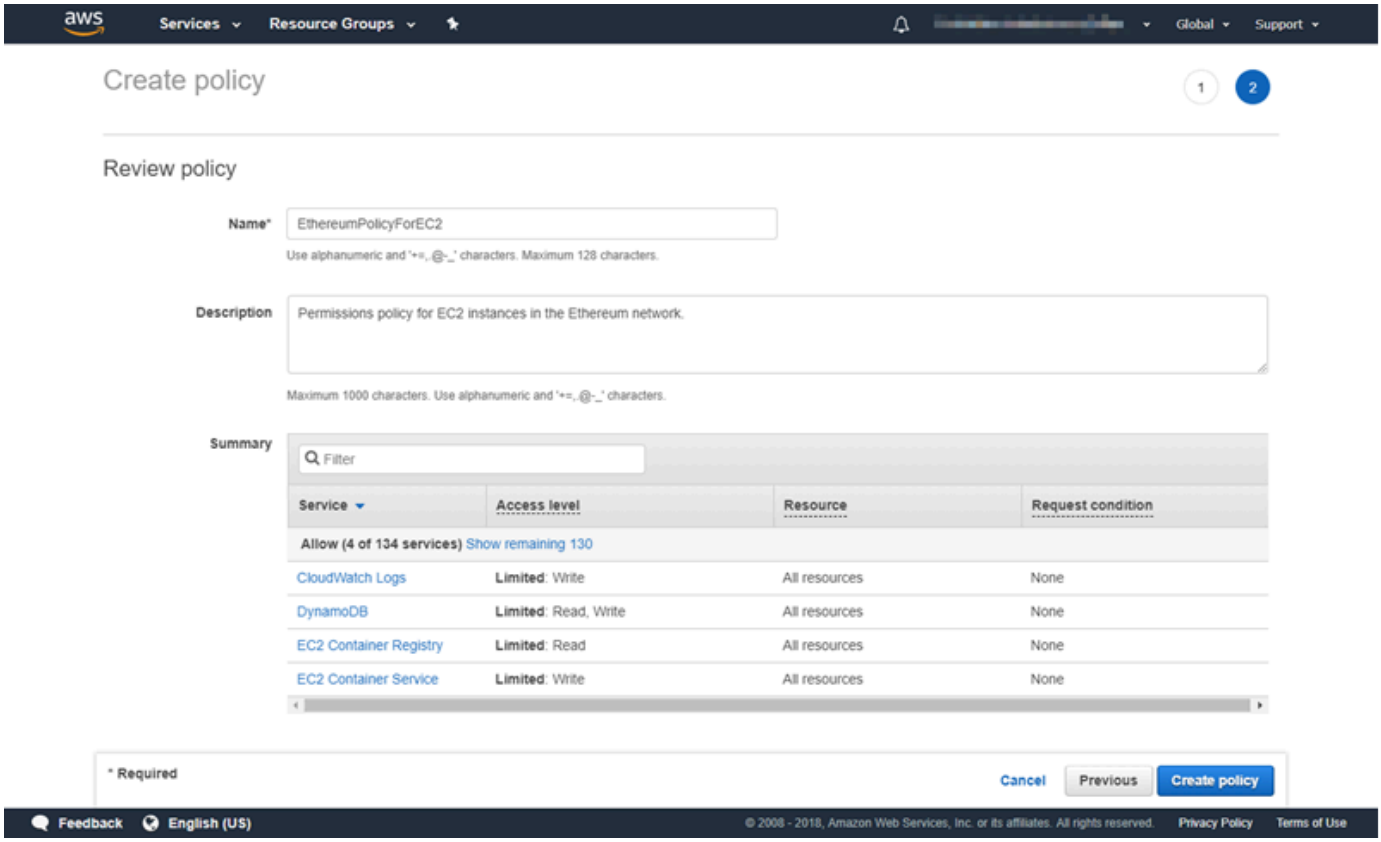
テンプレートで指定した EC2 インスタンスプロファイルは、他の AWS サービスとやり取りするために Ethereum ネットワークの EC2 インスタンスによって引き受けられます。ロールのアクセス許可ポリシーを作成し、ロールを作成します (同じ名前のインスタンスプロファイルが自動的に作成されます)。次に、これらのアクセス許可ポリシーをロールにアタッチします。

EC2 インスタンスプロファイルを作成するには

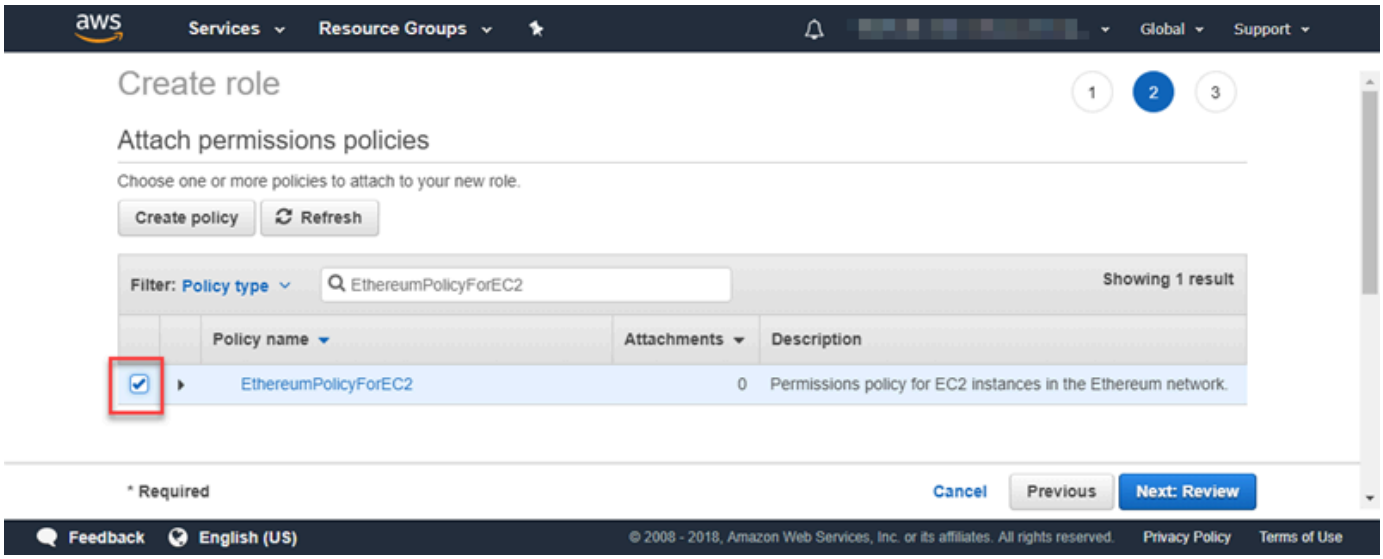
1. ナビゲーションペインで、[Policies]、[Create policy] の順に選択してください。
2. [JSON] を選択し、デフォルトのポリシーステートメントを次の JSON ポリシーに置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:DiscoverPollEndpoint",
                "ecs:Poll",
                "ecs:RegisterContainerInstance",
                "ecs:StartTelemetrySession",
                "ecs:Submit*",
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "dynamodb:BatchGetItem",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:Scan",
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": "*"
        }
    ]
}
```

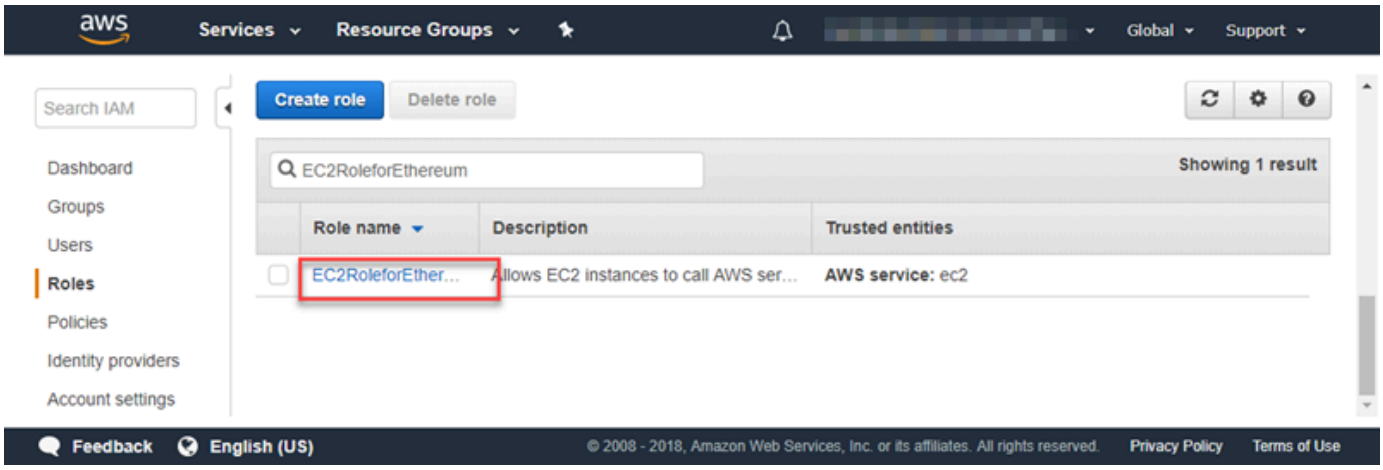
3. [ポリシーの確認] を選択します。
4. [名前] に、このアクセス許可ポリシーのわかりやすい名前を入力します (例: EthereumPolicyForEC2)。[説明] に、簡単な概要を入力します。[Create policy] (ポリシーの作成) を選択します。



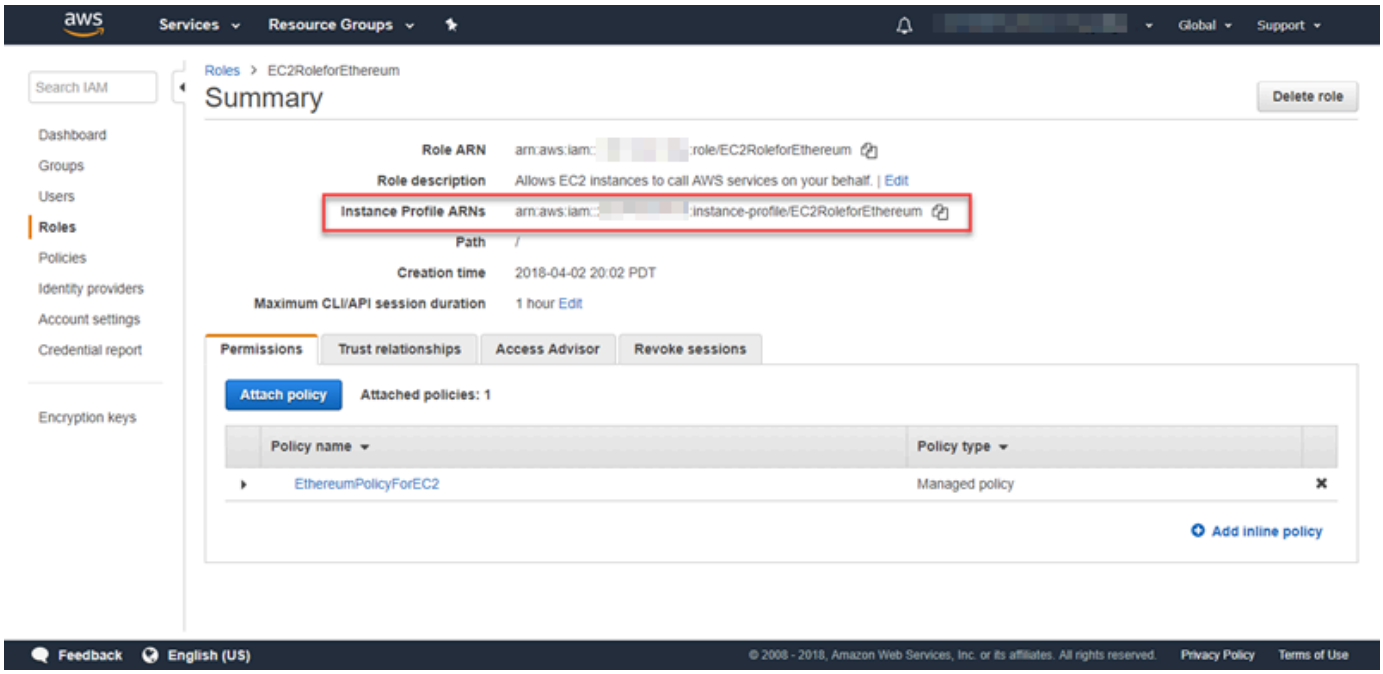
5. [ロール]、[ロールの作成] を選択します。
6. [EC2]、[次の手順: アクセス許可] の順に選択します。
7. [検索] フィールドに、先ほど作成したアクセス許可ポリシーの名前 (例: EthereumPolicyForEC2) を入力します。
8. 先ほど作成したポリシーのチェックマークを選択し、[次の手順: 確認] を選択します。



9. [ロール名] に、ロールのわかりやすい名前を入力します (例: EC2RoleForEthereum)。[ロールの説明] に簡単な説明を入力し、[ロールの作成] を選択します。
10. リストから、作成したロールを選択します。アカウントに多くのロールがある場合は、[検索] フィールドにロール名を入力できます。



11. [インスタンスプロファイルの ARN] の値をコピーし、再利用できるように保存しておきます。Ethereum ネットワークを作成するときに、この ARN が必要です。



要塞ホストの作成

このチュートリアルでは、要塞ホストを作成します。これは、Ethereum ネットワークのウェブインターフェイスとインスタンスに接続するために使用する EC2 インスタンスです。その唯一の目的

は、VPC 外の信頼されたクライアントからの SSH トラフィックを転送し、これらのクライアントから Ethereum ネットワークのリソースにアクセスできるようにすることです。

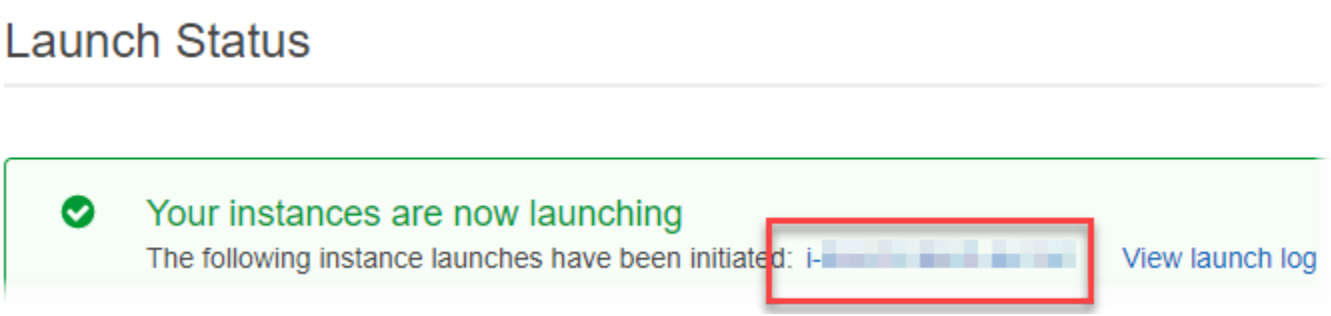
テンプレートで作成する Application Load Balancer は内部型である (内部 IP アドレスのみをルーティングする) ため、要塞ホストをセットアップします。要塞ホスト:

- 前に作成した 2 番目のパブリックサブネットで起動されるため、Application Load Balancer が認識する内部 IP アドレスを持ちます。
- VPC 外の信頼されたソースからアクセスできるパブリック IP アドレスがサブネットから割り当てられます。
- 前に作成した Application Load Balancer のセキュリティグループに関連付けられます。このセキュリティグループには、信頼されたクライアントからの SSH トラフィック (ポート 22) を許可するインバウンドルールがあります。

信頼されたクライアントから Ethereum ネットワークにアクセスするには、要塞ホストを介して接続するようにクライアントを設定する必要があります。詳細については、「要塞ホストを使用して EthStats および EthExplorer に接続する」を参照してください。要塞ホストは 1 つの方法です。信頼されたクライアントから VPC 内のプライベートリソースにアクセスできる方法であれば、どれでも使用できます。

要塞ホストを作成するには

1. Amazon Amazon EC2ユーザーガイドの最初の 5 つのステップに従って、インスタンスを起動します。
2. [インスタンスの詳細の編集] を選択します。[ネットワーク] で前に作成した VPC を選択し、[サブネット] で前に作成した 2 番目のパブリックサブネットを選択します。その他すべての設定はデフォルトのままにします。
3. 変更を確認するメッセージが表示されたら、[確認と作成] を選択します。
4. [セキュリティグループの編集] を選択します。[セキュリティグループの割り当て] で、[既存のセキュリティグループを選択する] を選択します。
5. セキュリティグループのリストから、前に作成した Application Load Balancer のセキュリティグループを選択し、[確認と作成] を選択します。
6. [Launch] (起動する) を選択します。
7. インスタンス ID を書き留めます。後で「要塞ホストを使用して EthStats および EthExplorer に接続する」際に必要になります。



Ethereum ネットワークを作成する

このトピックのテンプレートを使用して指定した Ethereum ネットワークは、Ethereum ネットワークの EC2 インスタンスの Amazon ECS クラスターを作成する CloudFormation スタックを起動します。テンプレートは、「前提条件の設定」で前に作成したリソースに依存します。

テンプレートを使用して CloudFormation スタックを起動すると、一部のタスク用にネストされたスタックが作成されます。作成の完了後、要塞ホストを介してネットワークの Application Load Balancer が提供するリソースに接続し、Ethereum ネットワークが実行中およびアクセス可能であることを確認できます。

Ethereum 用の AWS Blockchain Template を使用して Ethereum ネットワークを作成するには

1. 「AWS Blockchain Templates の開始方法」を参照し、AWS リージョンのクイックリンクを使用して CloudFormation コンソールで最新の AWS Blockchain Template for Ethereum を開きます。
2. 次のガイドラインに従って値を入力します。

   - [スタック名] に、わかりやすい名前を入力します。この名前は、スタックが作成するリソースの名前の中で使用されます。
   - [Ethereum ネットワークのパラメータ] と [プライベート Ethereum ネットワークのパラメータ] は、デフォルト設定のままにしておきます。

   > **Warning**
   >
   > デフォルトのアカウントおよび関連するニーモニックフレーズは、テスト目的でのみ使用します。ニーモニックフレーズにアクセスできるすべてのユーザーはアカウントから Ether にアクセスしたり、Ether を盗んだりできるため、デフォルトの一連のアカウントを使用して実際の Ether を送信しないでください。代わりに、本稼働用の

カスタムアカウントを指定します。デフォルトのアカウントに関連付けられているニーモニックフレーズは `outdoor father modify clever trophy abandon vital feel portion grit evolve twist` です。

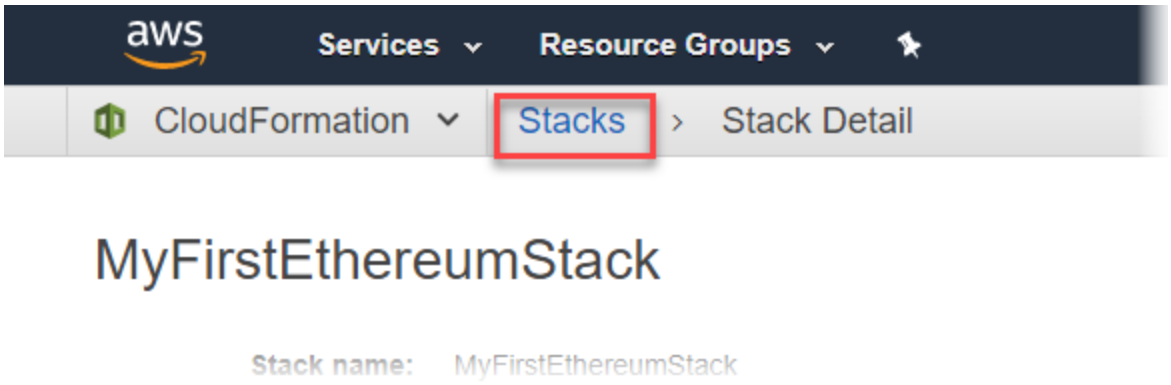
- [Platform configuration] (プラットフォーム設定) をデフォルト設定のままにしておくと、EC2 インスタンスの Amazon ECS クラスターが作成されます。または、[docker-local] で単一の EC2 インスタンスを使用して Ethereum ネットワークを作成します。
- [EC2 設定] で、次のガイドラインに従ってオプションを選択します。
  - [EC2 キーペア] でキーペアを選択します。キーペアの作成の詳細については、「キーペアの作成」を参照してください。
  - [EC2 セキュリティグループ] で、「セキュリティグループを作成する」で前に作成したセキュリティグループを選択します。
  - [EC2 インスタンスプロファイル ARN] に、「Amazon ECS および EC2 インスタンスプロファイルの IAM ロールを作成する」で前に作成したインスタンスプロファイルの ARN を入力します。
- [VPC ネットワーク設定] で、次のガイドラインに従ってオプションを選択します。
  - [VPC ID] で、「VPC とサブネットを作成する」で前に作成した VPC を選択します。
  - [Ethereum ネットワークサブネット ID] で、手順「To create the VPC」で先ほど作成した単一のプライベートサブネットを選択します。
- [ECS クラスター設定] は、デフォルトのままにします。これにより、3 つの EC2 インスタンスの ECS クラスターが作成されます。
- [Application Load Balancer 設定 (ECS のみ)] で、次のガイドラインに従ってオプションを選択します。
  - [Application Load Balancer サブネット ID] で、前に書き留めた list of subnets から 2 つのパブリックサブネットを選択します。
  - [Application Load Balancer セキュリティグループ] で、「セキュリティグループを作成する」で前に作成した Application Load Balancer のセキュリティグループを選択します。
  - [IAM Role] (IAM ロール) に、Amazon ECS および EC2 インスタンスプロファイルの IAM ロールを作成する で前に作成した ECS ロールの ARN を入力します。
- [EthStats] で、次のガイドラインに従ってオプションを選択します。
  - [Deploy EthStats] は、デフォルト設定 (true) のままにしておきます。
  - [EthStats 接続シークレット] に、6 文字以上の任意の値を入力します。
- [ethExplorer] で、[ethExplorer のデプロイ] をデフォルト設定 (true) のままにします。

- [その他のパラメータ] で、[Nested Template S3 URL Prefix] をデフォルト値のままにし、その値を書き留めておきます。ここで、ネストされたテンプレートを見つけることができます。

3. 他のすべての設定をデフォルトのままにし、確認のチェックボックスをオンにして、[作成] を選択します。

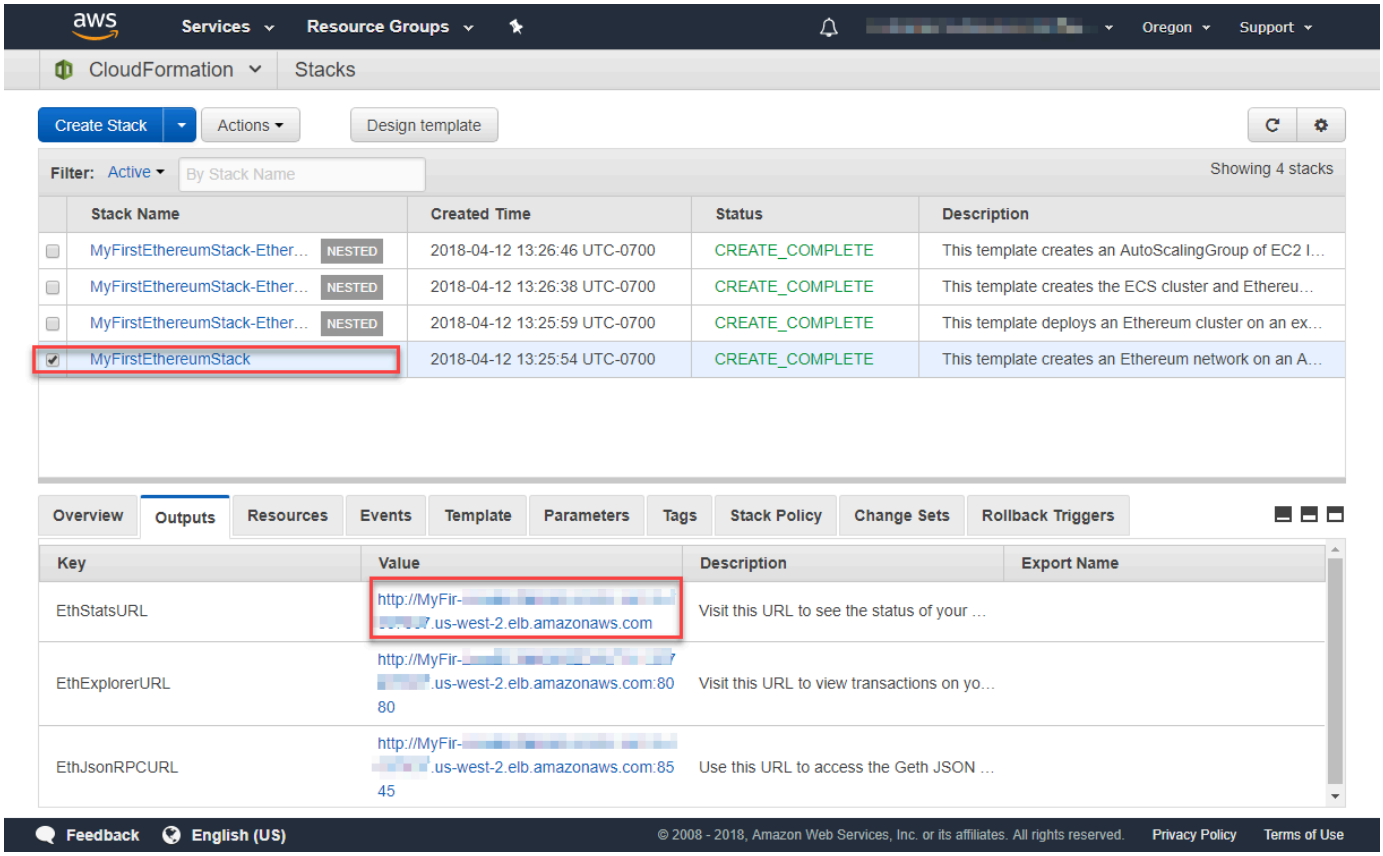
   CloudFormation 起動するルートスタックのスタック詳細ページが表示されます。

4. ルートスタックとネストされたスタックの進行状況をモニタリングするには、[スタック] を選択します。



5. すべてのスタックで [Status] (ステータス) に [CREATE_COMPLETE] が表示されたら、Ethereum のユーザーインターフェイスに接続して、ネットワークが動作し、アクセス可能であることを確認できます。ECS コンテナプラットフォームを使用する場合、Application Load Balancer を介してEthStats、EthExplorer、および EthJsonRPC に接続するための URLは、ルートスタックの [出力] タブで使用できます。

> **Important**
>
> クライアントコンピュータの要塞ホストを介してプロキシ接続を設定するまでは、これらの URL や SSH に直接接続することはできません。詳細については、「要塞ホストを使用して EthStats および EthExplorer に接続する」を参照してください。



要塞ホストを使用して EthStats および EthExplorer に接続する

このチュートリアルで Ethereum リソースに接続するには、要塞ホストを介して SSH ポート転送 (SSH トンネリング) を設定します。次の手順は、この設定を行うことにより、ブラウザを使用して EthStats と EthExplorer の URL に接続する方法を示しています。次の手順では、まずローカルポートに SOCKS プロキシを設定します。次に、ブラウザ拡張機能である FoxyProxy を使用し、この転送ポートを Ethereum ネットワークの URL で使用します。

Mac OS または Linux を使用している場合は、SSH クライアントを使用して要塞ホストへの SOCKS プロキシ接続を設定します。Windows ユーザーの場合は、PuTTY を使用します。接続する前に、使用しているクライアントコンピュータが、以前に設定した Application Load Balancer のセキュリティグループで、インバウンド SSH トラフィックの許可されたソースとして指定されていることを確認します。

SSH を使用して SSH ポート転送で要塞ホストに接続するには

- 「Amazon EC2 ユーザーガイド」の「SSH を使用した Linux インスタンスへの接続」の手順に従います。 Amazon EC2 Linux インスタンスへの接続手順のステップ 4 で、SSH コマンドに -D 9001 を追加し、AWS Blockchain Template 設定で指定したものと同じキーペアを指定して、踏み台ホストの DNS 名を指定します。

```
ssh -i /path/my-template-key-pair.pem ec2-user@bastion-host-dns -D 9001
```

PuTTY を使用して SSH ポート転送で要塞ホストに接続するには (Windows)

1. Amazon Amazon EC2 ユーザーガイドのPuTTY を使用して Windows から Linux インスタンスに接続する」の手順に従って、PuTTY セッションの開始」の手順のステップ 7 を実行し、AWS Blockchain Template for Ethereum 設定で指定したのと同じキーペアを使用します。
2. PuTTY の [Category] で、[Connection]、[SSH]、[Tunnels] の順に選択します。
3. [Port forwarding] で、[Local ports accept connections from other hosts] を選択します。
4. [Add new forwarded port] で、次の操作を行います。

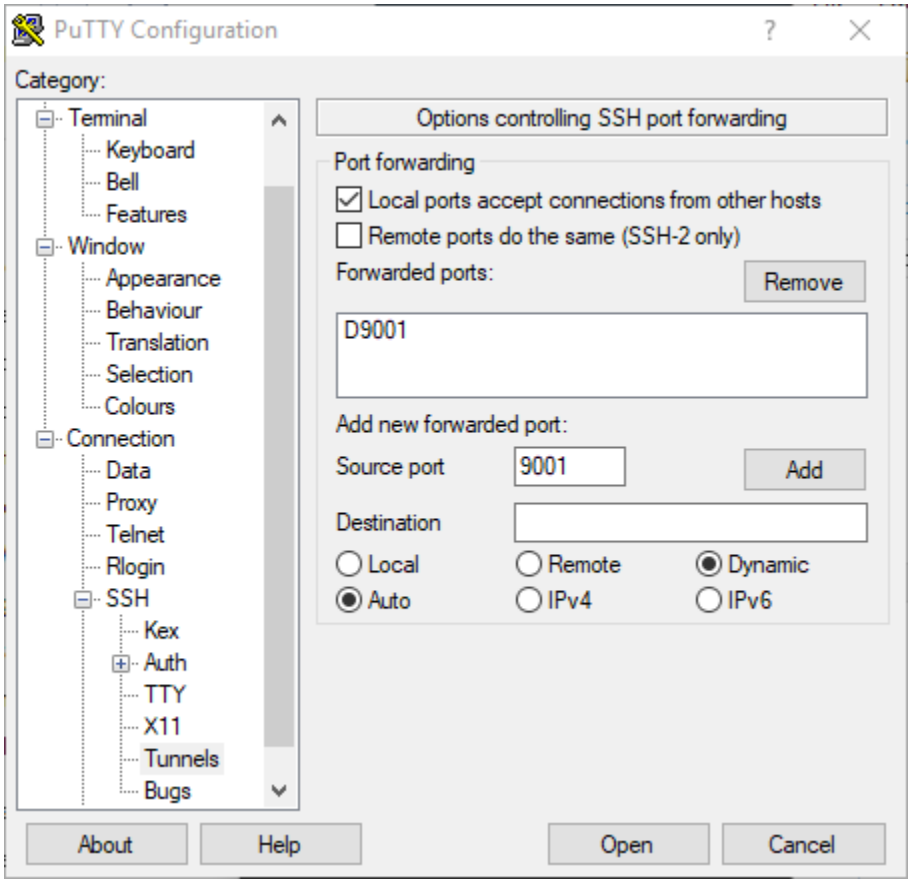
   a. [Source port] に「9001」と入力します。これは、任意に選択した未使用のポートであり、必要に応じて別のポートを選択できます。

   b. [Destination] を空白のままにします。

   c. [Dynamic] を選択します。

   d. [Add] (追加) を選択します。

   [Forwarded ports] の場合、[D9001] は次のように表示されます。



5. [Open] を選択し、キー設定の必要に応じて要塞ホストに対して認証します。接続を開いたままにします。

PuTTY 接続を開いた状態で、Ethereum ネットワーク URL の転送ポートを使用するようにシステムまたはブラウザ拡張機能を設定します。次の手順では、以前に転送ポートとして設定した EthStats および EthExplorer の URL パターンとポート 9001 に基づき、接続の転送方法として FoxyProxy Standard を使用していますが、任意の方法を使用できます。

Ethereum ネットワーク URL で SSH トンネルを使用するように FoxyProxy を設定するには

次の手順は Chrome に基づいて作成されています。別のブラウザを使用する場合は、そのブラウザの FoxyProxy バージョンに応じた設定およびシーケンスに変換してください。

1. FoxyProxy Standard のブラウザ拡張機能をダウンロードしてインストールし、ブラウザの指示に従って [Options] を開きます。
2. [Add New Proxy] を選択します。

3. [General] (全般) タブでプロキシが [Enabled] (有効) になっていることを確認し、このプロキシ構成を識別しやすいように、[Proxy Name] (プロキシ名) と [Proxy Notes] (プロキシノート) に入力します。
4. [Proxy Details] タブで、[Manual Proxy Configuration] を選択します。[Host or IP Address] (一部のバージョンでは [Server or IP Address]) に「localhost」と入力します。[Port] に[9001」と入力します。[SOCKS Proxy?] を選択します。
5. [URL Pattern] タブで、[Add New Pattern] を選択します。
6. [Pattern name] に、わかりやすい名前を入力します。[URL Pattern] に、テンプレートで作成したすべての Ethereum リソース URL と一致するパターン (http://internal-MyUser-LoadB-* など) を入力します。URL の表示については、「Ethereum URLs」を参照してください。
7. 他の設定はデフォルトのままにして、[Save] を選択します。

これで、テンプレートで作成したルートスタックの [Outputs] タブを使用して、CloudFormation コンソールで Ethereum URL に接続できるようになりました。

リソースのクリーンアップする

CloudFormation を使用すると、スタックが作成したリソースを簡単にクリーンアップできます。スタックを削除すると、スタックが作成したすべてのリソースが削除されます。

テンプレートが作成したリソースを削除するには

- CloudFormation コンソールを開き、前に作成したルートスタックを選択し、アクション、削除を選択します。

  前に作成したルートスタックおよび関連するネストされたスタックの [Status] が [DELETE_IN_PROGRESS] に更新されます。

Ethereum ネットワーク用に作成した前提条件を削除することもできます。

VPC を削除する

- Amazon VPC コンソールを開き、以前作成した VPC を選択して、[Actions] (アクション)、[Delete VPC] (VPC の削除) の順に選択します。これにより、VPC に関連付けられたサブネット、セキュリティグループ、および NAT ゲートウェイも削除されます。

IAM ロールと EC2 インスタンスプロファイルを削除する

- IAM コンソールを開き、[Roles] (ロール) を選択します。前に作成した ECS のロールと EC2 のロールを選択し、[Delete] を選択します。

要塞ホストの EC2 インスタンスを終了する

- Amazon EC2 ダッシュボードを開き、[Running instances] (実行中のインスタンス) を選択して、踏み台ホスト用に作成した EC2インスタンスを選択し、[Actions] (アクション)、[Instance State] (インスタンスの状態)、[Terminate] (終了) の順に選択します。

AWS Blockchain Templates と機能

このセクションでは、すぐにブロックチェーンネットワークを作成するためのリンクと、 AWSでネットワークを設定するための設定オプションと前提条件に関する情報について説明します。

次のテンプレートを使用できます。

- Ethereum 用の AWS Blockchain Template
- Hyperledger Fabric 用の AWS Blockchain Template

AWS Blockchain Templates は以下のリージョンで利用できます。

- 米国西部 (オレゴン) リージョン (us-west-2)
- 米国東部 (バージニア北部) リージョン (us-east-1)
- 米国東部 (オハイオ) リージョン (us-east-2)

> **Note**
>
> 上記に記載されていないリージョンでテンプレートを実行すると、米国東部 (バージニア北部) リージョン (us-east-1) でリソースが起動します。

Ethereum 用の AWS Blockchain Template の使用

Ethereum は、Ethereum 固有の言語である Solidity を使用してスマートコントラクトを実行するブロックチェーンフレームワークです。Homestead は Ethereum の最新のリリースです。詳細については、Ethereum Homestead のドキュメントと Solidity のドキュメントを参照してください。

起動へのリンク

Ethereum テンプレートを使用して特定のリージョンで起動するリンクについては、「AWS ブロックチェーンテンプレートの開始方法」を参照してください。 CloudFormation

Ethereum のオプション

テンプレートを使用して Ethereum ネットワークを設定する場合は、後続の要件を決定する選択を行います。

コンテナプラットフォームの選択

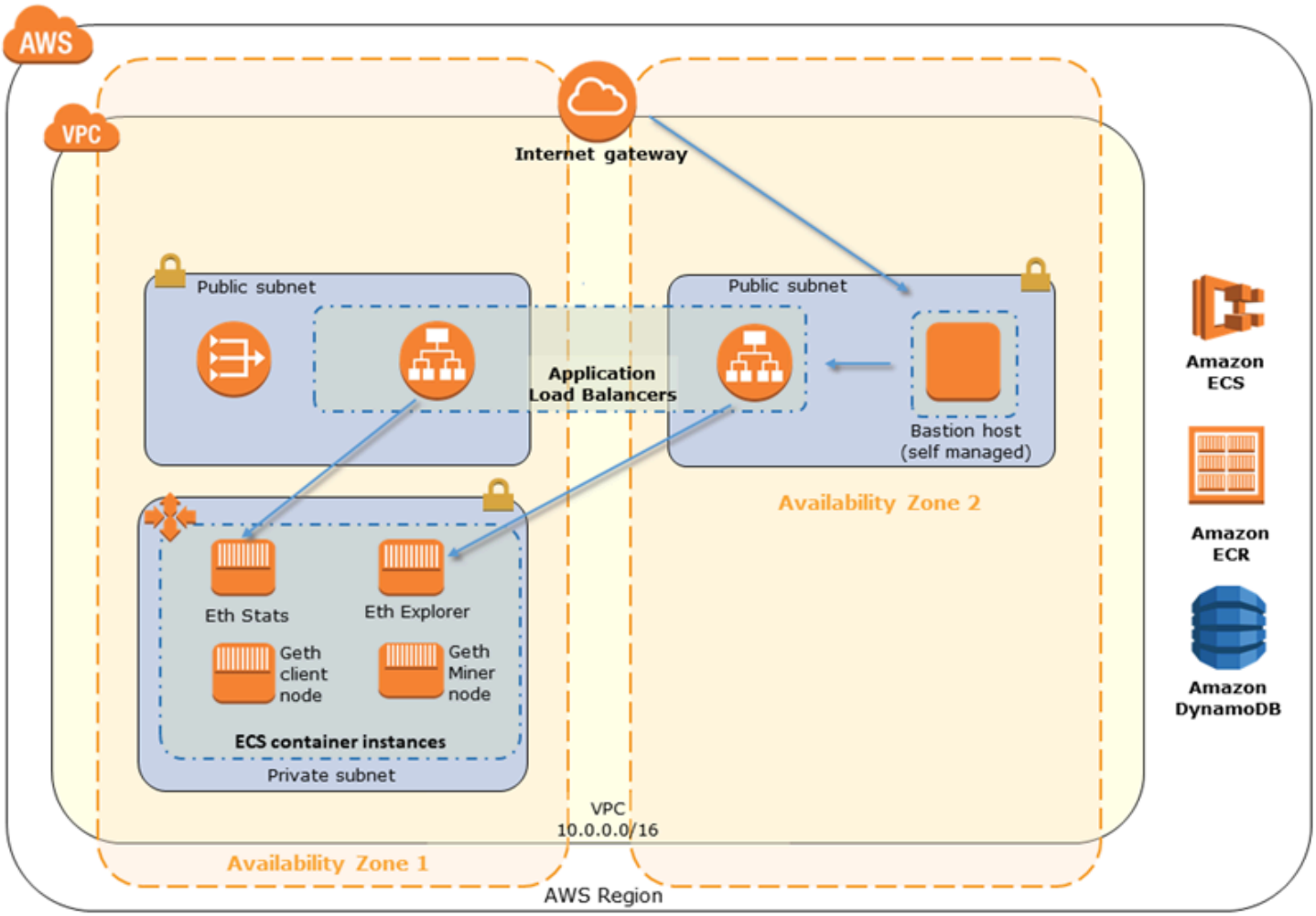
AWS Blockchain Templates は、Amazon ECR に保存されている Docker コンテナを使用してブロックチェーンソフトウェアをデプロイします。Ethereum 用の AWS Blockchain Template には、[Container Platform] (コンテナプラットフォーム) について 2 つの選択肢が用意されています。

- ecs — Ethereum が Amazon EC2 インスタンスの Amazon ECS クラスターで実行されるように指定します。
- docker-local — Ethereum が単一の EC2 インスタンスで実行されるように指定します。

Amazon ECS コンテナプラットフォームの使用

Amazon ECS で Application Load Balancer および関連リソースを使用し、複数の EC2 インスタンスで構成された ECS クラスターで Ethereum ネットワークを作成します。Amazon ECS 設定の使用の詳細については、AWS Blockchain Templates の開始方法チュートリアルを参照してください。

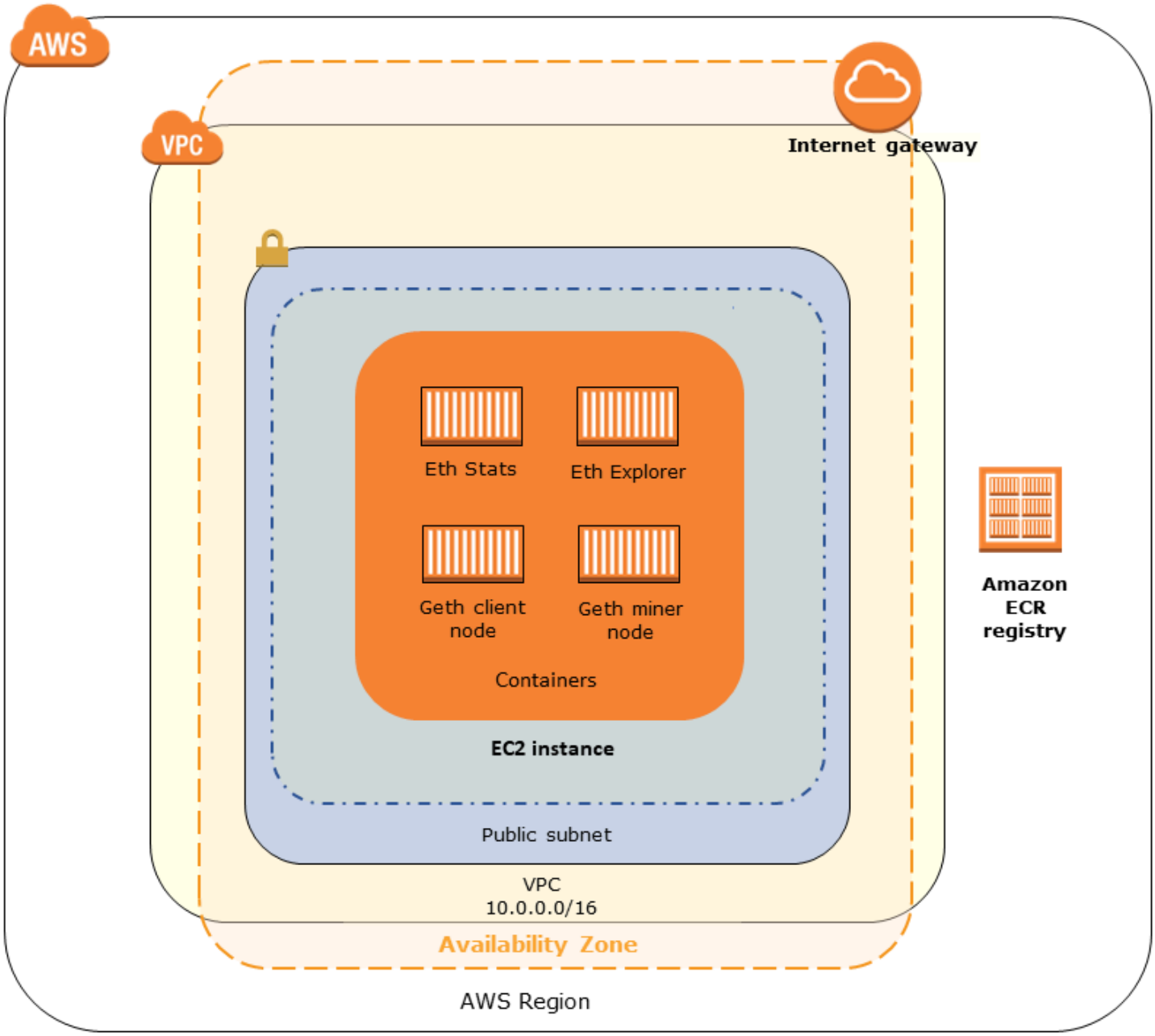
次の図は、テンプレートで ECS コンテナプラットフォームオプションを使用して作成した Ethereum ネットワークを示しています。



Docker-Local プラットフォームの使用

または、単一の Amazon EC2 インスタンス内で Ethereum コンテナを起動することができます。すべてのコンテナが単一の EC2 インスタンスで実行されます。これは簡略化されたセットアップです。

次の図は、テンプレートで docker-local コンテナプラットフォームオプションを使用して作成した Ethereum ネットワークを示しています。



プライベートまたはパブリック Ethereum ネットワークの選択

1 ~ 4 以外の [Ethereum Network ID (Ethereum ネットワーク ID)] 値を選択すると、指定したプライベートネットワークパラメーターを使用して、定義したネットワーク内で実行されるプライベート Ethereum ノードが作成されます。

[Ethereum Network ID] (Ethereum ネットワーク ID) 1 ~ 4 を選択すると、作成した Ethereum ノードは、公開されている Ethereum ネットワークに結合されます。プライベートネットワーク設定とそのデフォルト値は無視できます。Ethereum ノードをパブリック Ethereum ネットワークに参加させ

る場合は、ネットワーク内の適切なサービスがインターネットにアクセス可能であることを確認してください。

デフォルトのアカウントおよびニーモニックフレーズの変更

ニーモニックフレーズは、任意のネットワークにおいて、関連付けられたアカウントの Ethereum ウォレット (プライベート/パブリックキーペア) を生成するために使用できる一連のランダムな単語です。ニーモニックフレーズは、関連付けられたアカウントの Ether にアクセスするために使用できます。デフォルトのアカウントに関連付けられたデフォルトのニーモニックが Ethereum テンプレート用に作成済みです。

> **Warning**
>
> デフォルトのアカウントおよび関連するニーモニックフレーズは、テスト目的でのみ使用します。ニーモニックフレーズにアクセスできるすべてのユーザーはアカウントから Ether にアクセスしたり、Ether を盗んだりできるため、デフォルトの一連のアカウントを使用して実際の Ether を送信しないでください。代わりに、本稼働用のカスタムアカウントを指定します。デフォルトのアカウントに関連付けられているニーモニックフレーズは `outdoor father modify clever trophy abandon vital feel portion grit evolve twist` です。

前提条件

Ethereum 用の AWS Blockchain Template を使用して Ethereum ネットワークを設定する場合は、次に示す最小要件を満たす必要があります。テンプレートには、以下のカテゴリごとにリストされている AWS コンポーネントが必要です。

トピック

- Ethereum リソースにアクセスするための前提条件
- IAM の前提条件
- セキュリティグループの前提条件
- VPC 前提条件
- EC2 インスタンスプロファイルと ECS ロールの IAM アクセス許可の例

Ethereum リソースにアクセスするための前提条件

前提条件	ECS プラットフォームの場合	Docker-Local の場合
EC2 インスタンスへのアクセスに使用できる Amazon EC2 キーペア。このキーは、ECS クラスターおよび他のリソースと同じリージョンに存在する必要があります。	✓	✓
要塞ホストやインターネット接続ロードバランサーなど、Application Load Balancer へのトラフィックが許可された内部アドレスを持つインターネット接続コンポーネント。テンプレートはセキュリティ上の理由から内部ロードバランサーを作成するため、これは ECS プラットフォームに必須です。EC2 インスタンスがプライベートサブネットにある場合、これは docker-local プラットフォームに必須です (推奨)。要塞ホストの設定については、「要塞ホストの作成」を参照してください。	✓	✓ (プライベートサブネットを使用)

IAM の前提条件

前提条件	ECS プラットフォームの場合	Docker-Local の場合
関連するすべてのサービスを処理するアクセス許可を持つ	✓	✓

前提条件	ECS プラットフォームの場合	Docker-Local の場合
IAM の原則 (ユーザーまたはグループ)。		
EC2 インスタンスが他のサービスとやりとりするための適切なアクセス許可を持つ Amazon EC2 インスタンスプロファイル。詳細については、「To create an EC2 instance profile」を参照してください。	✓	✓
Amazon ECS が他のサービスとやりとりするアクセス許可を持つ IAM ロール。詳細については、「ECS ロールとアクセス許可の作成」を参照してください。	✓	

セキュリティグループの前提条件

前提条件	ECS プラットフォームの場合	Docker-Local の場合
EC2 インスタンスのセキュリティグループと、次の要件:	✓	✓
• 0.0.0.0/0 (デフォルト) へのトラフィックを許可するアウトバウンドルール。	✓	✓
• インバウンドルール自体からのすべてのトラフィックを許可するインバウンドルール (同じセキュリティグループ)。	✓	✓

前提条件	ECS プラットフォームの場合	Docker-Local の場合
• Application Load Balancer のセキュリティグループからのすべてのトラフィックを許可するインバウンドルール。	✓	
• クライアントコンピュータの IP CIDR など、信頼された外部ソースからの HTTP (ポート 80)、EthStats (ポート 8080 で提供)、JSON RPC over HTTP (ポート 8545)、および SSH (ポート 22) を許可するインバウンドルール。		✓

前提条件	ECS プラットフォームの場合	Docker-Local の場合
Application Load Balancer のセキュリティグループと、以下の要件: • インバウンドルール自体からのすべてのトラフィックを許可するインバウンドルール (同じセキュリティグループ)。 • EC2 インスタンスの同じセキュリティグループからのすべてのトラフィックを許可するインバウンドルール。 • EC2 インスタンスのセキュリティグループへのすべてのトラフィックを許可するアウトバウンドルール。詳細については、「セキュリティグループを作成する」を参照してください。 • この同じセキュリティグループを要塞ホストに関連付ける場合は、信頼されたソースからの SSH (ポート 22) トラフィックを許可するインバウンドルール。 • 要塞ホストや他のインターネット接続コンポーネントが別のセキュリティグループにある場合は、そのコンポーネントからのトラフィックを許可するインバウンドルール。	✓	

VPC 前提条件

前提条件	ECS プラットフォームの場合	Docker-Local の場合
Elastic IP アドレス。Ethereum サービスへのアクセスに使用されます。	✓	✓
EC2 インスタンスを実行するサブネット。プライベートサブネットを強くお勧めします。	✓	✓
パブリックにアクセス可能な 2 つのサブネット。各サブネットは、別個のアベイラビリティーゾーンに配置し、それは EC2 インスタンスのサブネットと同じアベイラビリティーゾーンであることが必要です。	✓	

EC2 インスタンスプロファイルと ECS ロールの IAM アクセス許可の例

テンプレートを使用するときは、EC2 インスタンスプロファイル ARN をパラメータの 1 つとして指定します。ECS コンテナプラットフォームを使用する場合は、ECS ロールの ARN も指定します。これらのロールにアタッチされたアクセス権限ポリシーにより、クラスターの AWS リソースとインスタンスは、他の AWS リソースとやり取りすることができます。詳細については、IAM ユーザーガイドの IAM ロールを参照してください。アクセス許可を作成するための出発点として、以下のポリシーステートメントと手順を使用してください。

EC2 インスタンスプロファイルのアクセス許可ポリシーの例

次のアクセス許可ポリシーは、ECS コンテナプラットフォームを選択したときに、EC2 インスタンスプロファイルに対して許可されるアクションを示しています。同じポリシーステートメントは、ドッカーローカルコンテナプラットフォームで使用可能で、アクセスを制限するために `ecs` コンテキストキーが削除されています。

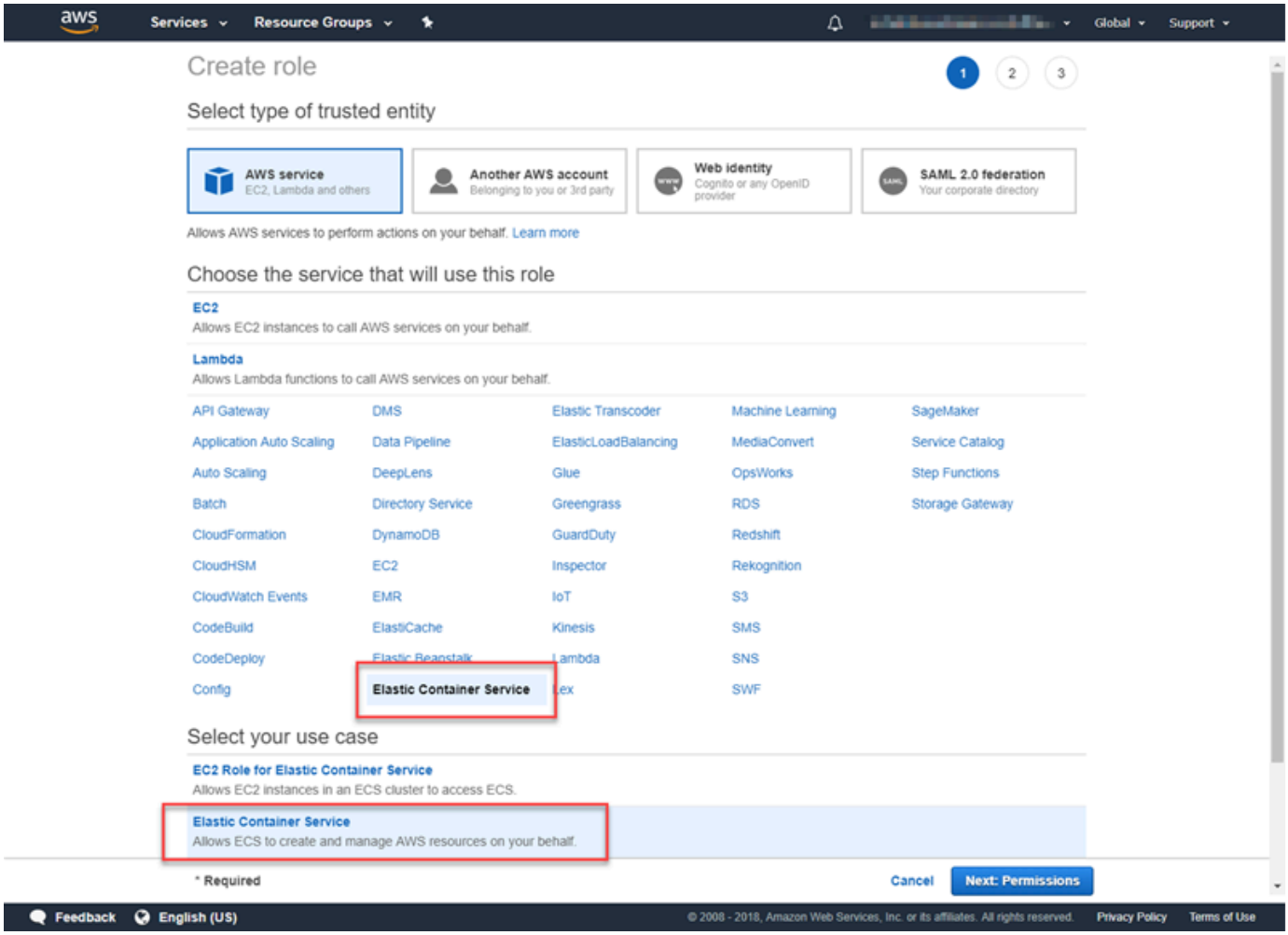
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:DiscoverPollEndpoint",
                "ecs:Poll",
                "ecs:RegisterContainerInstance",
                "ecs:StartTelemetrySession",
                "ecs:Submit*",
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "dynamodb:BatchGetItem",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:Scan",
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": "*"
        }
    ]
}
```

ECS ロールとアクセス許可の作成

ECS ロールにアタッチするアクセス許可については、[AmazonEC2ContainerServiceRole] アクセス許可ポリシーから開始することをお勧めします。ロールを作成し、このアクセス許可ポリシーをアタッチするには、次の手順を実行します。IAM コンソールを使用して、このポリシーの最新のアクセス許可を表示します。

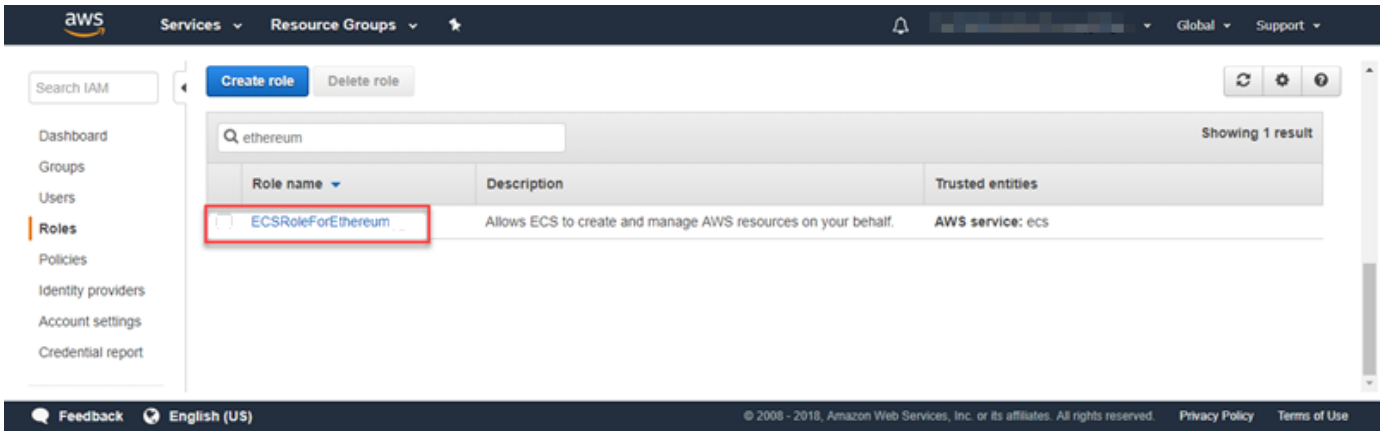
Amazon ECS の IAM ロールを作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで [Roles (ロール)]、[Create Role (ロールの作成)] の順に選択します。
3. [信頼されたエンティティの種類を選択] で、[AWS のサービス] を選択します。
4. [Choose the service that will use this role] (このロールを使用するサービスを選択) で、[Elastic Container Service] (伸縮自在コンテナサービス) を選択します。
5. [ユースケースの選択] で [Elastic Container Service] を選択し、[Next: Permissions (次の手順: アクセス許可)] を選択します。

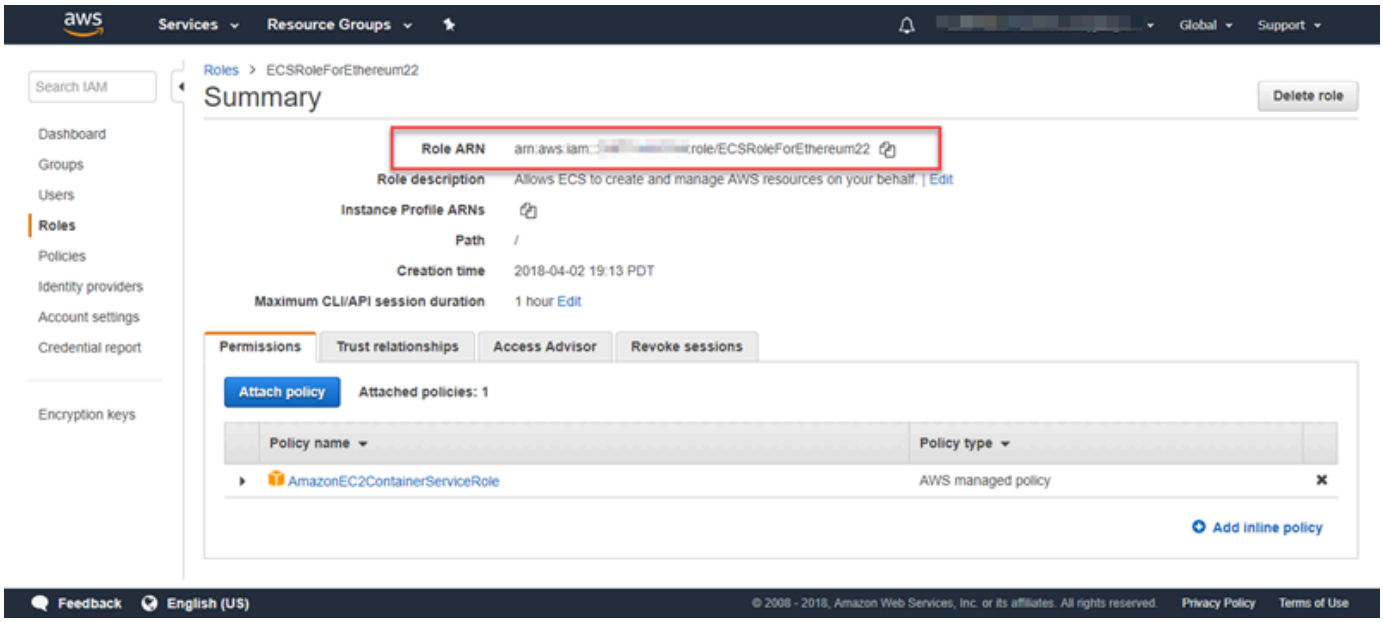


6. [Permissions policy (アクセス許可ポリシー)] で、デフォルトのアクセス許可ポリシー (AmazonEC2ContainerServiceRole) を選択したままにし、[Next:Review (次の手順: 確認)] を選択します。
7. [ロール名] に、ロールを識別するのに役立つ値を入力します (例: ECSRoleForEthereum)。[ロールの説明] に、簡単な要約を入力します。後で使用するため、ロール名を書き留めておきます。

8. [ロールの作成] を選択してください。
9. リストから、作成したロールを選択します。アカウントに多数のロールがある場合は、ロール名で検索できます。



10. [ロールの ARN] の値をコピーし、再度コピーできるように保存します。Ethereum ネットワークを作成するときに、この ARN が必要です。



Ethereum リソースへの接続

テンプレートで作成したルートスタックに CREATE_COMPLETE が表示されたら、 CloudFormation コンソールを使用して Ethereum リソースに接続できます。接続する方法は、選択した ECS または docker-local のコンテナプラットフォームによって異なります。

- ECS — ルートスタックの [Output] (出力) タブは、Application Load Balancer で実行されているサービスへのリンクを提供します。セキュリティ上の理由から、これらの URL に直接アクセスす

ることはできません。接続するには、要塞ホストを設定し、このホストを使用してプロキシ接続を行います。詳細については、以下の 踏み台ホストを使用したプロキシ接続 を参照してください。

- docker-local — 以下に示す Ethereum サービスをホストする EC2 インスタンスの IP アドレスを使用して接続します。テンプレートで作成したインスタンスの *ec2-IP-address* を見つけるには、EC2 コンソールを使用します。
  - EthStats — http://*ec2-IP-address* を使用
  - EthExplorer — http://*ec2-IP-address*:8080 を使用
  - EthJsonRpc — http://*ec2-IP-address*:8545 を使用

  [Ethereum Network Subnet ID] (テンプレート内で使用する VPC サブネットのリスト) でパブリックサブネットを指定した場合は、直接接続できます。クライアントは、SSH (ポート 22) のインバウンドトラフィックの信頼できる送信元である必要があります。これは Ethereum 用の AWS Blockchain Template で指定した [EC2 Security Group] (EC2 セキュリティグループ) によって決まります。

  プライベートサブネットを指定した場合は、要塞ホストを設定し、このホストを通じてこれらのアドレスへのプロキシ接続を行うことができます。詳細については、以下の 踏み台ホストを使用したプロキシ接続 を参照してください。

踏み台ホストを使用したプロキシ接続

一部の構成では、Ethereum サービスが一般公開されない場合があります。このような場合は、踏み台ホストを介して Ethereum リソースに接続できます。踏み台ホストの詳細については、Linux 踏み台ホストクイックスタートガイドの Linux 踏み台ホストアーキテクチャを参照してください。

踏み台ホストは EC2 インスタンスです。以下の要件が満たされていることを確認してください。

- 踏み台ホストの EC2 インスタンスが、[Auto-assign Public IP] (自動割り当てパブリック IP) が有効な状態でインターネットゲートウェイを持つパブリックサブネット内にある。
- 踏み台ホストに、ssh 接続を許可するキーペアがある。
- 接続するクライアントからのインバウンド SSH トラフィックを許可するセキュリティグループに踏み台ホストが関連付けられている。
- Ethereum ホストに割り当てられたセキュリティグループ (例えば、ECS がコンテナプラットフォームの場合は Application Load Balancer、docker-local がコンテナプラットフォームの場合はホスト EC2 インスタンス) は、VPC 内のソースからのすべてのポートでのインバウンドトラフィックを許可します。

踏み台ホストを設定したら、接続するクライアントが踏み台ホストをプロキシとして使用していることを確認します。次の例では、Mac OS を使用してプロキシ接続を設定しています。*BastionIP* を踏み台ホストの EC2 インスタンスの IP アドレスと置き換え、*MySshKey.pem* を踏み台ホストにコピーしたキーペアファイルと置き換えます。

コマンドラインで、以下のように入力します。

```
ssh -i mySshKey.pem  ec2-user@BastionIP -D 9001
```

これにより、ローカルマシン上のポート 9001 の踏み台ホストへのポート転送が設定されます。

次に、`localhost:9001` の SOCKS プロキシを使用するようにブラウザまたはシステムを設定します。たとえば、Mac OS を使用して、[システム環境設定]、[ネットワーク]、[詳細]、[SOCKS プロキシ] の順に選択し、「localhost:9001」と入力します。

Chrome で FoxyProxy Standard を使用して、[その他のツール]、[拡張機能] の順に選択します。[FoxyProxy Standard] で、[詳細]、[拡張機能のオプション]、[プロキシを新規追加] の順に選択します。[手動プロキシ設定] を選択します。[ホストまたは IP アドレス] に「localhost」と入力し、[ポート] に「9001」と入力します。[SOCKS Proxy?]、[保存] を選択します。

これで、テンプレート出力で示した Ethereum ホストアドレスに接続できるようになります。

Hyperledger Fabric 用の AWS Blockchain Template の使用

Hyperledger Fabric は、チェーンコードというスマートコントラクトを実行するブロックチェーンフレームワークであり、Go で記述されます。Hyperledger Fabric を使用してプライベートネットワークを作成し、ネットワークに接続して参加できるピアを制限することができます。Hyperledger Fabric の詳細については、Hyperledger Fabric のドキュメントを参照してください。チェーンコードの詳細については、Hyperledger Fabric のドキュメントの Chaincode for Developers (開発者向けのチェーンコード) トピックを参照してください。

Hyperledger Fabric 用の AWS Blockchain Template では docker-local コンテナプラットフォームのみがサポートされているため、Hyperledger Fabric コンテナは単一の EC2 インスタンスにデプロイされます。

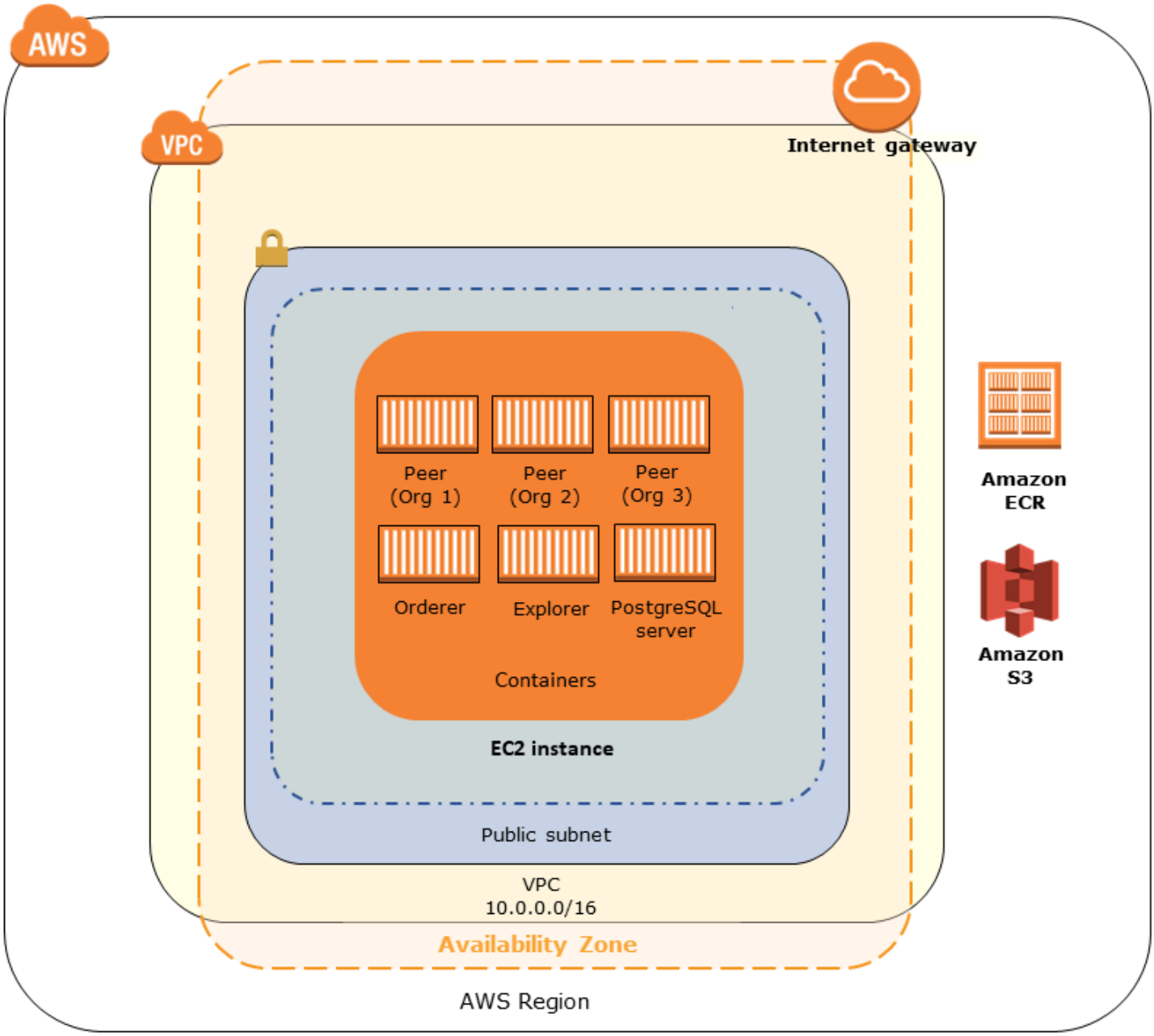
起動へのリンク

Hyperledger Fabric テンプレートを使用して特定のリージョンで起動するリンクについては、「AWS ブロックチェーンテンプレートの開始方法」を参照してください。 CloudFormation

Hyperledger Fabric 用の AWS Blockchain Template のコンポーネント

Hyperledger Fabric 用の AWS Blockchain Template は Docker で EC2 インスタンスを作成し、そのインスタンスのコンテナを使用して Hyperledger Fabric ネットワークを起動します。ネットワークには 1 つの注文サービスと 3 つの組織が含まれており、それぞれが 1 つのピアサービスを持っています。テンプレートはまた、ブロックチェーンデータを参照するための Hyperledger Explorer コンテナも起動します。PostgreSQL サーバーコンテナが起動されて Hyperledger Explorer をサポートします。

次の図は、テンプレートを使用して作成された Hyperledger Fabric ネットワークを示しています。



前提条件

テンプレートを使用して Hyperledger Fabric ネットワークを起動する前に、以下の要件が満たされていることを確認します。

- 使用する IAM の原則 (ユーザーまたはグループ) には、関連するすべてのサービスを処理するためのアクセス許可が必要です。
- EC2 インスタンスにアクセスするために使用できるキーペア (たとえば、SSH を使用) にアクセスできる必要があります。キーは、インスタンスと同じリージョンに存在する必要があります。
- コンテナをプルするための Amazon S3 へのアクセスと Amazon Elastic Container Registry (Amazon ECR) へのアクセスを許可するアクセス許可ポリシーがアタッチされた EC2 インスタンスプロファイルが必要です。アクセス許可ポリシーの例については、「EC2 インスタンスプロファイルの IAM アクセス許可の例」を参照してください。
- Amazon S3、および Amazon ECR にアクセスできるようにするには CloudFormation、パブリックサブネットを持つ Amazon VPC ネットワーク、または NAT ゲートウェイと Elastic IP アドレスを持つプライベートサブネットが必要です。
- SSH を使用してインスタンスに接続する必要がある IP アドレスからの SSH トラフィック (ポート 22) を許可するインバウンドルールが EC2 セキュリティグループに必要です。Hyperledger Explorer (ポート 8080) に接続する必要があるクライアントでも同様です。

EC2 インスタンスプロファイルの IAM アクセス許可の例

Hyperledger Fabric 用の AWS Blockchain Template を使用するときは、EC2 インスタンスプロファイル ARN をパラメータの 1 つとして指定します。EC2 のロールとインスタンスプロファイルにアタッチされているアクセス許可ポリシーの開始点として、次のポリシーステートメントを使用します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken",
```

```
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetRepositoryPolicy",
                "ecr:DescribeRepositories",
                "ecr:ListImages",
                "ecr:DescribeImages",
                "ecr:BatchGetImage",
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

Hyperledger Fabric のリソースへの接続

テンプレートで作成したルートスタックのステータスが [CREATE_COMPLETE] になったら、EC2 インスタンスの Hyperledger Fabric リソースに接続できます。パブリックサブネットを指定した場合は、他の EC2 インスタンスと同じように EC2 インスタンスに接続できます。詳細については、「Amazon EC2 ユーザーガイド」の「SSH を使用した Linux インスタンスへの接続」を参照してください。

プライベートサブネットを指定した場合は、踏み台ホストをセットアップして、Hyperledger Fabric リソースへのプロキシ接続に使用できます。詳細については、以下の 踏み台ホストを使用したプロキシ接続 を参照してください。

> **Note**
>
> テンプレートは、パブリック IP アドレスを Hyperledger Fabric サービスをホスティングする EC2 インスタンスに割り当てますが、指定したプライベートサブネット内のルーティングポリシーは、この IP アドレスとパブリックソースとの間のトラフィックを許可しないため、この IP アドレスはパブリックにアクセスすることはできません。

踏み台ホストを使用したプロキシ接続

一部の構成では、Hyperledger Fabric サービスが一般公開されない場合があります。このような場合は、踏み台ホストを介して Hyperledger Fabric リソースに接続できます。踏み台ホストの詳細につ

いては、Linux 踏み台ホストクイックスタートガイドの Linux 踏み台ホストアーキテクチャを参照してください。

踏み台ホストは EC2 インスタンスです。以下の要件が満たされていることを確認してください。

- 踏み台ホストの EC2 インスタンスが、[Auto-assign Public IP] (自動割り当てパブリック IP) が有効な状態でインターネットゲートウェイを持つパブリックサブネット内にある。
- 踏み台ホストに、ssh 接続を許可するキーペアがある。
- 接続するクライアントからのインバウンド SSH トラフィックを許可するセキュリティグループに踏み台ホストが関連付けられている。
- Hyperledger Fabric ホストに割り当てられたセキュリティグループ (例えば、ECS がコンテナプラットフォームの場合は Application Load Balancer、docker-local がコンテナプラットフォームの場合はホスト EC2 インスタンス) は、VPC 内のソースからのすべてのポートでのインバウンドトラフィックを許可します。

踏み台ホストを設定したら、接続するクライアントが踏み台ホストをプロキシとして使用していることを確認します。次の例では、Mac OS を使用してプロキシ接続を設定しています。*BastionIP* を踏み台ホストの EC2 インスタンスの IP アドレスと置き換え、*MySshKey.pem* を踏み台ホストにコピーしたキーペアファイルと置き換えます。

コマンドラインで、以下のように入力します。

```
ssh -i mySshKey.pem  ec2-user@BastionIP -D 9001
```

これにより、ローカルマシン上のポート 9001 の踏み台ホストへのポート転送が設定されます。

次に、`localhost:9001` の SOCKS プロキシを使用するようにブラウザまたはシステムを設定します。たとえば、Mac OS を使用して、[システム環境設定]、[ネットワーク]、[詳細]、[SOCKS プロキシ] の順に選択し、「localhost:9001」と入力します。

Chrome で FoxyProxy Standard を使用して、[その他のツール]、[拡張機能] の順に選択します。[FoxyProxy Standard] で、[詳細]、[拡張機能のオプション]、[プロキシを新規追加] の順に選択します。[手動プロキシ設定] を選択します。[ホストまたは IP アドレス] に「localhost」と入力し、[ポート] に「9001」と入力します。[SOCKS Proxy?]、[保存] を選択します。

これで、テンプレート出力で示した Hyperledger Fabric ホストアドレスに接続できるようになります。

ドキュメント履歴

以下の表は、このガイドのドキュメントの変更点をまとめたものです。

ドキュメントの最終更新日: 2019 年 5 月 1 日

変更	説明	日付
AWS Blockchain Templates の廃止	AWS Blockchain Templates は 2019 年 4 月 30 日に廃止されました。このサービスやサポートドキュメントは今後更新されません。で最高の Managed Blockchain エクスペリエンスを得るには AWS、Amazon Managed Blockchain (AMB) を使用することをお勧めします。	2019 年 5 月 1 日
要塞ホストの更新。	要塞ホストの追加に関する入門チュートリアルと Ethereum の前提条件の要件を変更しました。これにより、内部ロードバランサー (ECS プラットフォームを使用した場合) と EC2 インスタンス (docker-local を使用した場合) を介して提供されるウェブリソースにアクセスできます。	2018 年 5 月 3 日
ガイドを作成。	AWS Blockchain Templates の初期リリースをサポートする新しい開発者ガイド。	2018 年 4 月 19 日

AWS 用語集

最新の AWS 用語については、 AWS の用語集 リファレンスのAWS 用語集を参照してください。